**REPETITION 1**

**MVC:**Bygger på att webbservern och webbläsaren kommunicerar med varandra, via http-protokollet

Kommunikationen styrs genom http-protokollet

**Request och respons:**Request är fråga från webbläsaren till webbservern och responsen är svaret. MVC konceptet:

**Vad är MVC?**En webbapplikation består av saker, för att göra det enklare och tydligare, avgränsa varje del.

**Vilka är de olika delarna?**Controller: Sköter kommunikationen mellan fram och baksidan. Det som binder ihop M och V. Binder ihop allt. När man skickar anrop, controllen tar emot requesten och

Sambandscentral. När man skcikar request från klient hamnar den i controllern. Speciellt för MVC, INTE fysisk fil, det är en METOD som finns i en klass. Skiljer sig från andra tektniker på det sättet. Controllern är själva som tar emot ett anrop, sambandscentralen. När man skickar en request, hamnar den i en controller, HUR styr man VILKEN controller och actionmetod, vart ska request hamna? STYRS genom routing och URL

URL styr var man hamnar i en MVC lösning.

**VYN:**Reponsen är vyn. Vyn genererar upp responsen. Genererar upp htmlkod och kollar vilka länkar behöver den (cssfiler bilder etc ) för att kunna göra det den ska, vad behöver de. Den behöver html-kodn. Bakar in detta och skickar tillbaka till webbläsaren. Vyn hanterr responsen, men VYN ÄR INTE SJÄLVA RESPONSEN.

Vad kan man göra i vyn?  
Får inte statisk html kod, sidan ska vara dynamisk. Kan visa ren presentation av data som finns sparad någonstans. Får inte statisk html-kod, sidan ska vara dynamisk, det som finns i databasen.

Hur styr man i vyn vad som ska synas via db?

Lägger ut razorkod som anpassas utifrån omstädnigheterna. Vyn generear htmlkod, kan vara helt statisk, men oftast vill man den ska vara dynamisk.

* Ex senaste blogginlägg
* Vilka sökningar har en användare gjort
* Senaste ann

Vy är inte färdig html-kod, utan den generar kod. Vyn generaar upp det färdiga slutresultatet, generar upp responsen.

**MODELLEN:**

**Vad är en modell?**

**2 huvudsyftern: 1 hantera info som ska ut på sidan**Är en mappning mot vad som finns i db och det man vill visa på sidan. Hanterar info man vill visa på sidorna. Hantera den info som ska ut på hemsidan. Görs genom att man binder modellen till vyn, strongly typed view – FÖRDEL: binda modell till vy; Man får möjlighet att validera, om man gör litet fel dyker det upp snabbt, ser snabbt kopplingen, för formulär fördel: Man får värderna från modellen i input fil, det skickas in värden via modellen som man lätt kan skicka ut. Kan enkelt ta emot värden i modellklass.

Innan klasser kom skrev man mycket kod för att mappa

Strongly styped view: Kopplar en vy till en modell, kopplar data man ska visa på hemsidan kopplar men till vyn

Modellen representerar datan/info man vill visa på sidan som oftaa stämmer överens i datan. Db och modell liknar ofta, men kan vara från en tjänst man beställer också. Ex att man gör stora slagningar mot Adresser i sverige, via externt anrop

**View model:**Är en separat modell som är till för att representera data på ett visst sätt, om datan inte kan mappa direkt till en db.

Modellklasser ska mappa ex databaser, men ex på websidan innehålls info från flera tabeller, räcker inte med vanlig modellklass, UTVIDGAD modell som passar ju den här webbsidan, som byggs för just den. Innehåller ALL info som ska finnas på sidan.

Har man vyer som är mer komplexa o en enda klass inte räcker så gör man en viewmodel, enskilda parametrar, lista av objekt

ÄR EN UTÖKAD MODELL FÖR EN SPECIELL SIDA

**MODELLER**

**Andra VIKTIGT SYFTE FÖRUTOM ATT VISA DATA PÅ HEMSIDAN:** Datat vill man ofta uppdatera, när man sparar in datat är det viktigt attha VALIDERING, vill ha datat i rätt format. Man vill itne att det ska gå ner ändå till db för att se om ngt va felaktigt format. SäkerthetsdeleN: man vill fågnga upp saker så tidigrt som möjligt om man får in konstig data. Validerinig STYRS FRÅN Modellen

Mappa data som oftast finns i en db - Modelln

Regler för hur datat ska matas in – Modellen

PÅ klienten och på servern genom attgöra valiregler i modellen – vali kan då göras på åda ställena. Varför viktigt på båda ställena? Om man bara har på klienten kan det ändå bli javascrip, en egen post som inte går via webbsidan.

Man lämnar webbserveer helt öppen om man inte har vali på servern kan man skcika in ett anrop så finns det inget som stoppar, förutom när man kommer till db. Går att gå runt valideriing på klienten. Varför ha vali på klienten? Annars för mkt trafik mellan server o klient, dettar upp trafik så inga onödiga request går iväg. SÅ lite trafik som möjligt. Kan man göra ngt i webbläsaren gör man gärna det. Om man matar in i ett stort formulär o trycker submit, sidan hoppar lite o fryser..Ni har matat in dethär fältet fel, istället för att direkt påfå felmeddelandet när man trycker o bara meddelandet – Bättre användarupplevelse.

Kan man köra i webbläsaren ska man det.

Validerinh bde på klient o server.

**FORMULÄR**

**Strongly typed view:**Vyn är bunden till en modell 🡪 Man kan binda element i vyn till modellens properties

**HttpPost:**- Ingenting syns i <URL:en>  
- <URL:en> påverkas inte  
- Värden bakas in i sidan  
- Renare/snyggare URL  
- Värden kan döljas för användaren

**Postback:**För att ta emot en postback av ett formulär med httppost så måste man skapa en overload metod av den ursprungliga actionmetoden

**Bunden vy:**Vyn är bunden till en modell. Ex vid en postback, om vyn är bunden kommer en inparameter som objekt av datatyp modellklassen

* **Skicka värden vid overload**
* **HttpPOST:** Objekt kommer
* **HttpGET:** Värden tas emot som enskilda parametrar

**Skicka vidare till ny controller:**Ex vid ett inloggningsformulär

* Om användaren lyckats logga in med rätt inmatning skickas man vidare till en ny webbsida och även till en annan controller

***RedirectToAction(”NamnPaActionMetod”)***

* … INTE lyckats logga in så stannar man i samma controller

**Tömma ett formulär:**Om man ex är i ett inmatningsformulär och man efter att ha matat in värden vill tömma formuläret för att kunna mata in nya värden används  
***ModelState.Clear();***

**Binda vy till lista av modellobjekt**

**Dropdown och listbox**

**Radiobuttons**

**Modelbinding: Binda modellobjekt till en vy i form av lista**

* Man gör det genom att ändra @model till en lista
* Listan läggs ut i vyn som en HTML-lista

**Dropdown och listbox helpers**Tag helpers genererar dessa. Följande html-element skapas

* <select>-lista med en eller flera <option></option>
* Man måste skapa en List<selectListItem> i controllern
* Statiska värden kan fyllas i direkt i vyn, annars kan värdena skickas in till vyn och kopplas till listan
* FÖR LISTBOK LÄGGS **multiple=”multiple”** TILL

<select asp-for=”ID” **multiple=”multiple”** [asp-items=@Model.BanTyper></select](mailto:asp-items=@Model.BanTyper%3e%3c/select)>

<select id=”bantyper” **multiple=”multiple”** name=”bantyper”>  
 <option value=”1”>Squash</option>  
 <option value=”1”>Squash</option>

</select>

**Radiobuttons:**Används i formulär när det är något som har minst 2 alternativ och man endast får välja ett. Fördelen är att alla alternativ syns samtidigt på skärmen (jämför med listbox då det INTE syns)

* Då skickas INTE en List<selectListItem> med utan en lista av de objekt som ska synas

**Instuderingsfrågor MVC**

**Hur kommunicerar en webbläsare med en webbserver?**Klienten skickar en request till servern. Controllern tar emot requesten. Controllern och metod, hämtar datan från severn som skickas vidare till vyn. I vym är det inte färdig html-kod, utan vyn generar kod, det färdiga slutresultatet och generas upp som responsen.

**Vad är HTTP?**Grunden för kommunikationen webbserver-webbläsare. Det sätt som en webbläsare och webbserver kommunicerar.

**Vad är en Request och Response?**När en webbklient gör en förfrågan till en webbserver skickas en request över http. När servern svarar skickas ett svar tillbaka, en s.k. respons.

**Hur fungerar html formulär? Har du koll på form, input, label, textarea, fieldset, select, option, button-taggarna och deras attribut?**

**• Vad gör type, id, name och value-attributet på input taggar?**

**• Hur gör du radio-knappar respektive checkboxar? Hur använder du labeltaggen tillsammans med radio-knappar och checkboxar?**

**• Hur gör du en rullgardinsmenyer (drop down list) eller listboxar och hur fyller du dom med data som skickas med från controllern?**

**• Vad är skillnaden på GET och POST på ett formulär? Varför ska man välja POST istället för GET på en del formulär?**

**• Vad är URL? Vad är Query Parametrar?**

**• Vad är IIS?**

**• Vad är HttpContext objektets uppgift i ASP.NET?**Håller informationen om den aktuella http-requesten.

**• Vad är dependency injection, vad är det som kan injectas och hur kan det göras?**