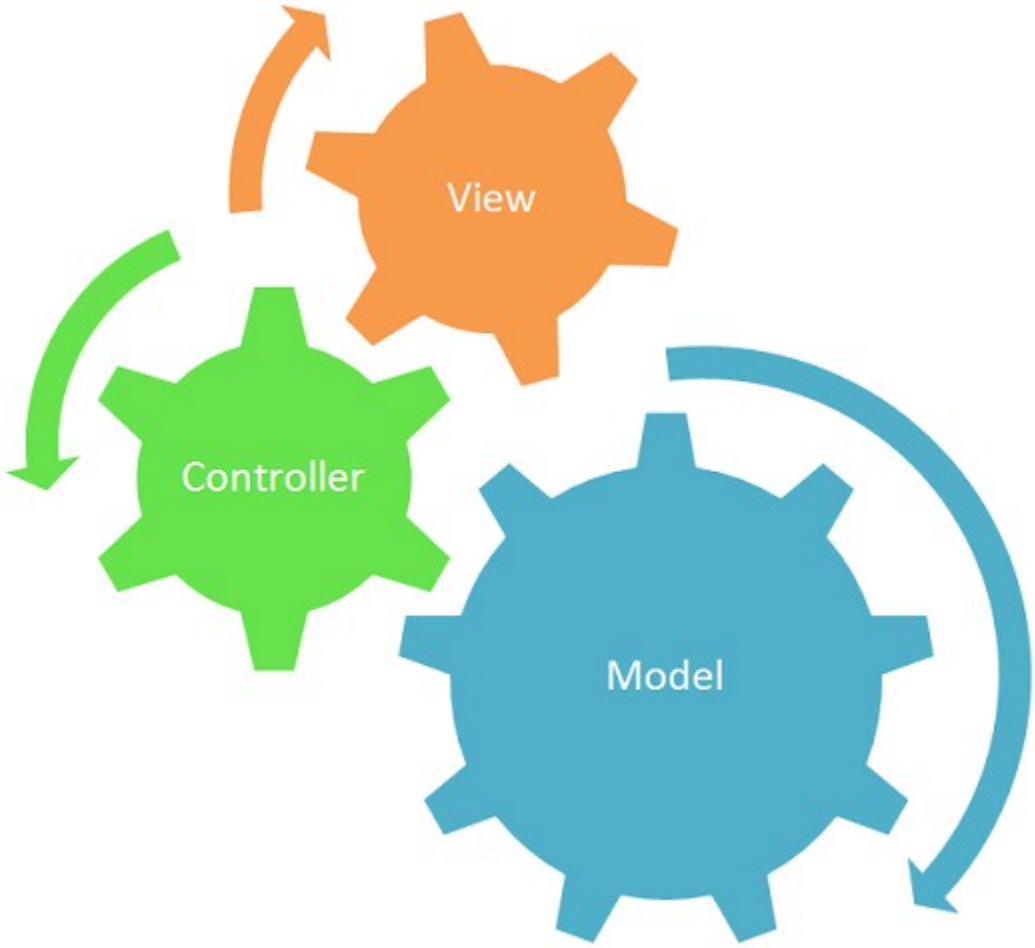
MVC

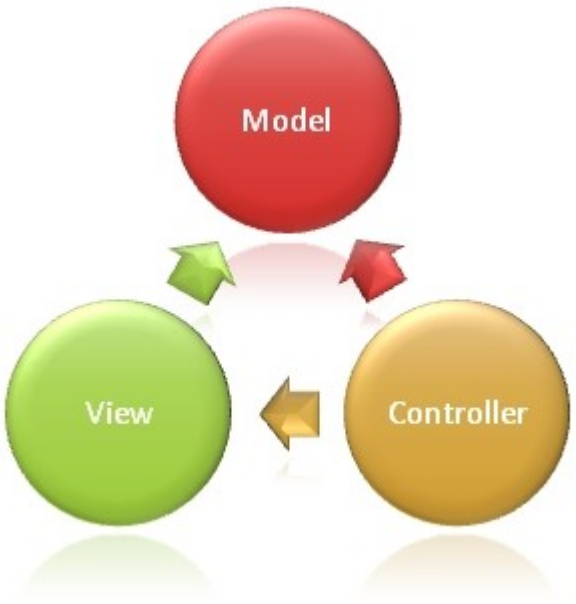


Model – View – Controller

Ce este MVC?

- MVC (Model – View - Controller) este un model de arhitectură utilizat în ingineria software.
- Succesul modelului se datorează izolării părții logice (business) de consideratele interfețe cu utilizatorul, rezultând o aplicație unde aspectul vizual sau/și nivelele inferioare ale regulilor de business sunt mai ușor de modificat, fără a afecta alte nivele.
- MVC este un model de arhitectură software care separă reprezentarea informațiilor din interacțiunea cu utilizatorul cu informațiile în sine.

Arhitectura MVC

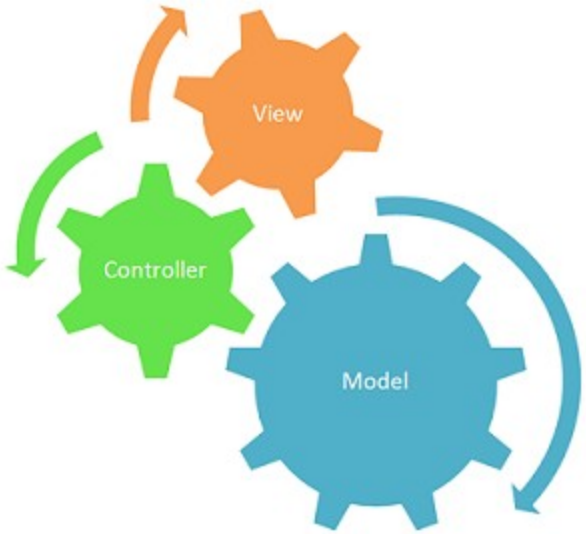


Modelul MVC definește aplicații web cu 3 straturi:

- Stratul business (logic) - Model
- Afișarea – View
- Control de intrare – Controller

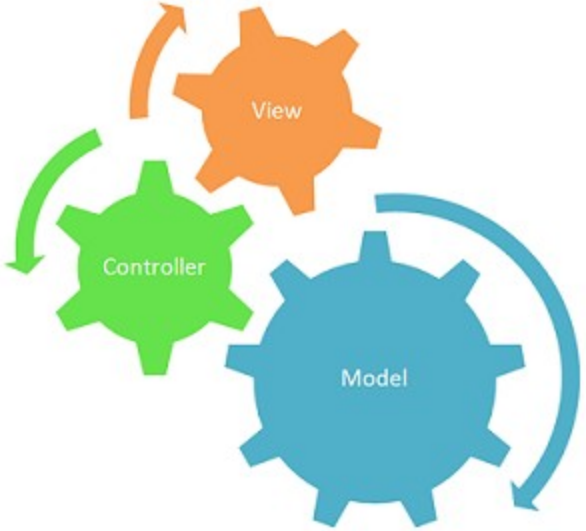
Model

- Modelul este responsabil cu gestionarea datelor din aplicație.
- Răspunde la cereri care vin din View, deasemenea și intrucțiunilor din Controller.
- Este cel mai jos nivel care se ocupă cu menținerea datelor.
- Modelul reprezintă nucleul aplicației – aici se face și legătura cu baza de date.
- Mai este numit și stratul de domeniu.



View

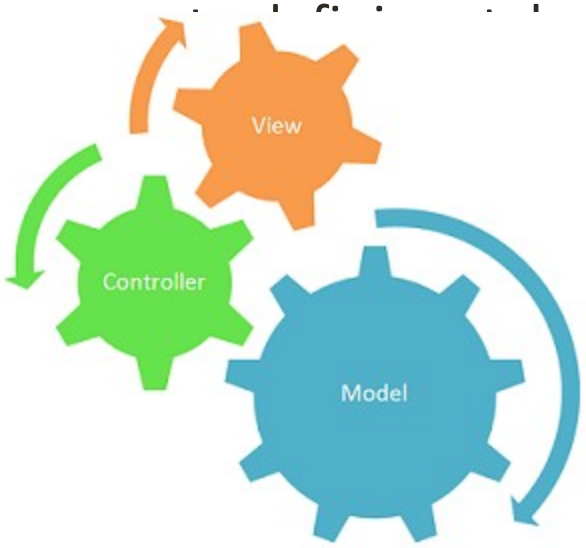
- View-ul este partea în care se afișează datele (înregistrările din baza de date).
- View-ul conține de cele mai multe ori limbaj HTML și este partea văzută de către utilizator.
- De asemena în view se pot face și operații javascrpit/jquery (este de preferat ca javascriptul/jquery să fie pus într-un fișier separat și să se facă referire la acel fișier în view-ul dorit)



Controller

- Controller-ul este partea aplicației care se ocupă de interacțiunea cu utilizatorul.
- În controller se citesc datele introduse se utilizator, se trimit către model, se execută operațile, după care se trimite răspunsul către view.
- Fiecare controller conție așa numitele **Action**-uri (niște funcții). Cu ajutrul structurii controllerului {controller}/{action}/{id} (de ex:

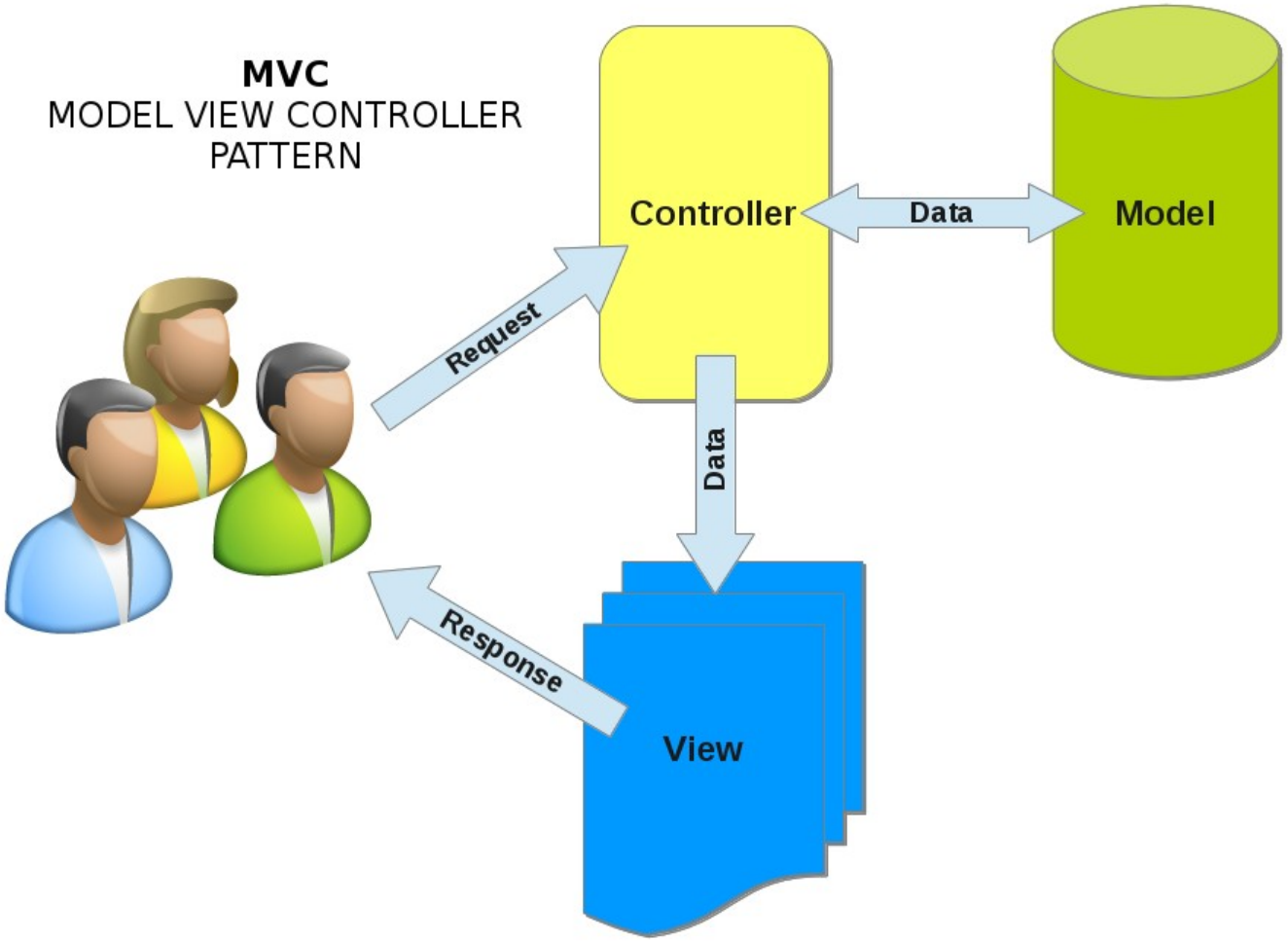
  Home/Index/4).



Cum funcționează MVC?

Fluxul de lucru în MVC este următorul:

1. Utilizatorul interacționează cu interfața de utilizator într-un fel (de exemplu, apasă un buton);
2. Un controller preia datele adăugate de către utilizator.
3. Controllerul crează modelul, eventual actualizează acesta într-un mod adecvat pentru acțiunea utilizatorului (de exemplu, coș de cumpărături actualizări regulatorului utilizator) și este trimis înapoi la view.
7. View-ul utilizează modelul pentru a genera o interfață pentru utilizator adecvată cerințelor (de ex. listează conținutul coșului de cumpărături). View-ul primește datele din model. Modelul nu are cunoștințe directe despre view.
8. Interfața pentru utiliztor așteptă o nouă intercțiune cu utilizatorul,


MVC
MODEL VIEW CONTROLLER
PATTERN
Controller
Data
Model
Request
Data
Response
View

Structura unei aplicații MVC

- **Informații despre aplicație**

  Properties

  References

- **Fișierele aplicației**

  App_Data

  App_Start

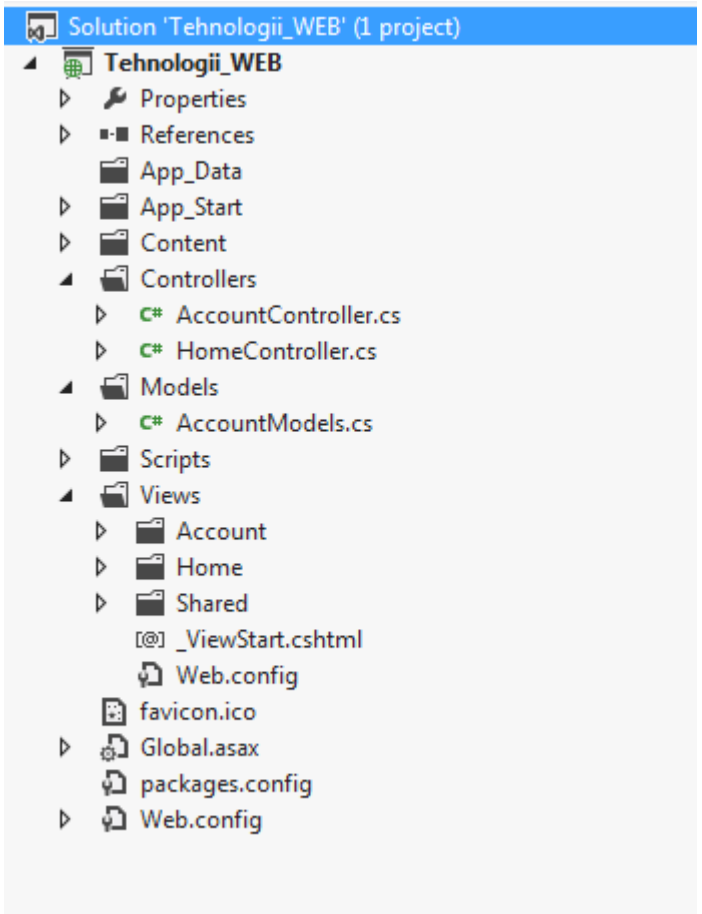
  Content

  Controllers

  Models

  Scripts

  Views



De ce să folosești MVC?

- MVC este unul din acele concepte care au sens doar atunci când ai de-a face cu aplicații mai mare și/sau atunci când lucrează mai mulți dezvoltatori pe aceeași cerințe.
- Dacă sunt 3 cerințe de făcut până la o anumită dată, nu se simte nevoia de a avea o organizare a sarcinilor, dar dacă vor fi zeci de sarcini diferite atunci se începe utilizarea așa numitelor liste TODO. Dar atunci când trebuie să lucreze o echipă de oameni pe același proiect și trebuie să acorde prioritate cerințelor nu mai este suficientă această listă. În astfel de situații devin mai evidente beneficiile MVC.
- În cazult aplicațior pentru mobile, 90% din codul aplicației web va fi același pentru mobile, dar în loc de a salva datele utiliazatorului la un obiect comun sau printr-un serviciu web, se va folosi un DB local (în cazul aplicaților offline). Fără MVC, șansele de a modifica o mulțime de clase sunt destul de mari.

Avantajele utilizării MVC

- Organizare
- Dezvoltare rapidă
- Reutilizarea codului
- Dezvoltare paralelă
- Flexibilitate
- Permite standarde diferite de interfață

Dezavantajele utilizării MVC

- Complezitatea este mare la dezvoltarea aplicațiilor folosind acest model fără învățare corespunzătoare.
- Nu este chiar adecvat utilizarea acestul model pentru aplicații extra-mici.
- Procesul acesta de izolare a interfețelor de nivelul logic și de controllere poate duce la o întârziere în dezvolarea aplicației.

Vă mulțumesc pentru atenție!