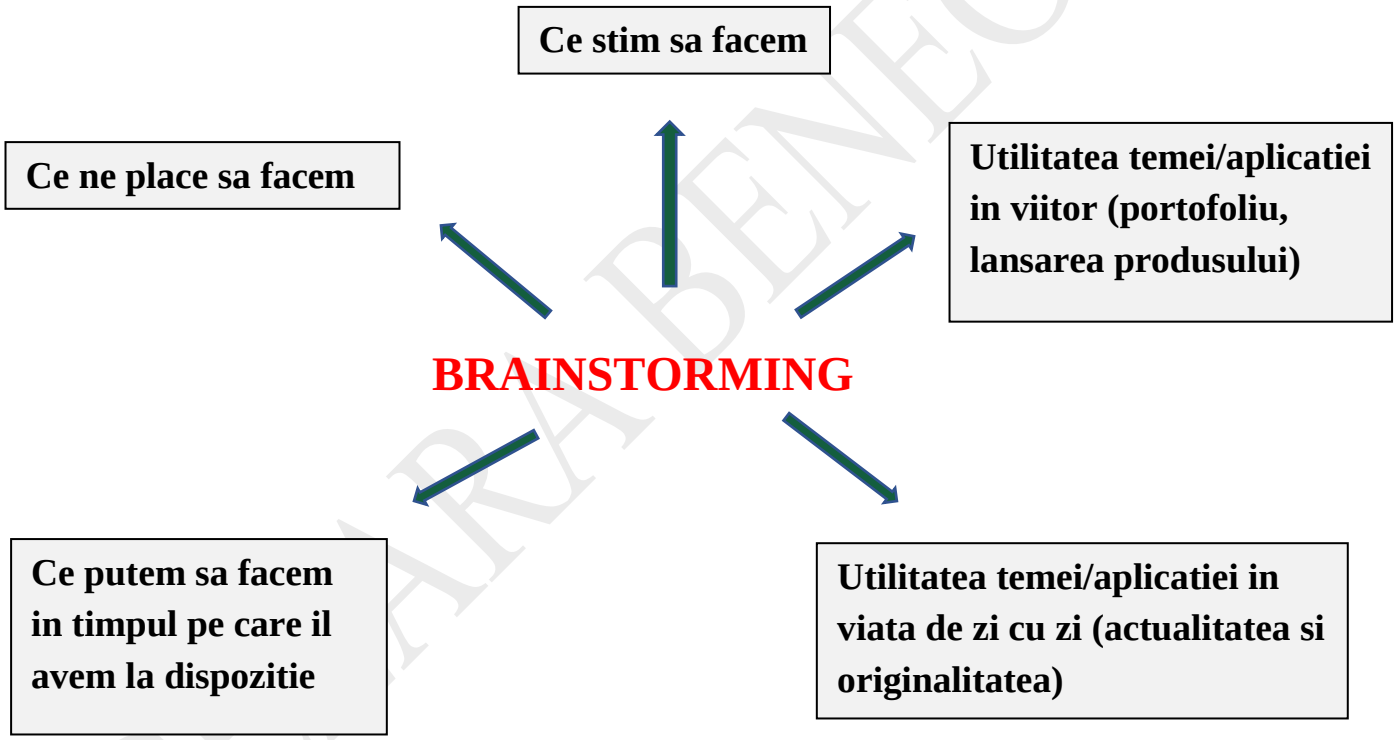
**Realizarea Lucrarii de Licenta - 2023**
**Aplicatie / Documentatie**

---

# Alegerea temei

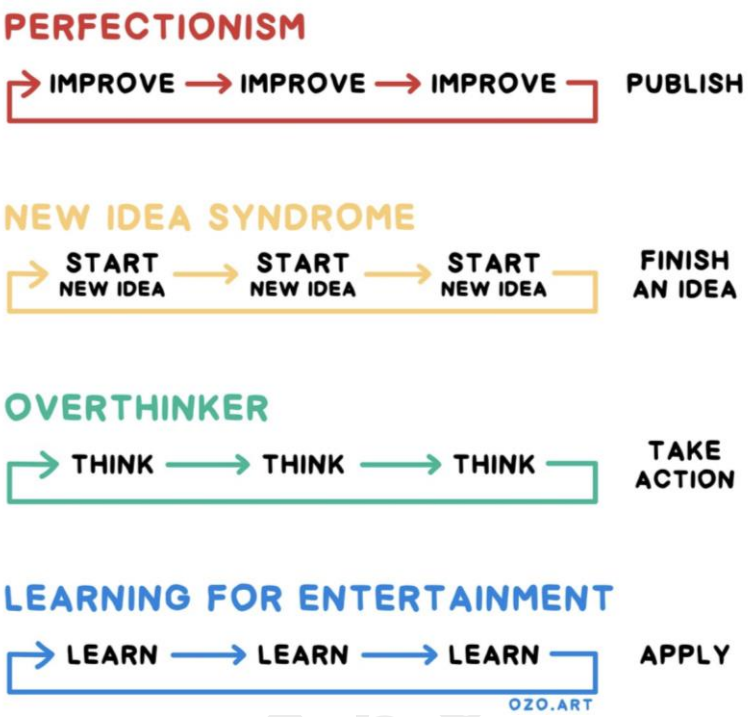


In momentul in care au fost alese 2-3 teme posibile, se poate realiza o Diagrama **SWOT** (**Strength, Weakness, Opportunity, Threat**) pentru a face cea mai buna alegere.

In tabel exista doar cateva idei pentru fiecare categorie a diagramei. Se poate extinde analiza in functie de specificul aplicatiei si de ideea de dezvoltare.

| | |
|---|---|
| **Strength – Puncte tari**<br><br>➢ Toate avantajele pe care le aduce aplicatia (ce probleme rezolva)<br>➢ Utilitatea unei astfel de aplicatii | **Opportunity – Oportunitati**<br><br>➢ Avantajele pe care aplicatia le aduce utilizatorilor finali |
| **Weakness – Puncte slabe**<br><br>➢ Aplicatia poate fi utila, dar ideea sa existe deja implementata (competitia pe piata) | **Threat – Amenintari**<br><br>➢ Competitie – alte aplicatii asemanatoare sau identice<br>➢ Timpul pe care il avem la dispozitie pentru dezvoltarea aplicatiei<br>➢ Cat de bine cunoastem limbajul/limbajele de programare / framework-urile utilizate |

**Ce trebuie sa evitam atunci cand demaram un astfel de proiect?**




# Originalitatea / Contributia proprie

- Aplicatia web nu trebuie sa fie doar o integrare de API-uri din care in final sa rezulte o aplicatie comuna. Aplicatie Web **NU INSEAMNA** site web (acesta fiind doar un site de prezentare)

- Aplicatia web trebuie sa aiba sistem de autentificare si implicit utilizatori care pot avea diferite roluri in cadrul aplicatiei

- In primul rand, aplicatia trebuie sa aiba o **CONRIBUTIE ORIGINALA** si un **SCOP BINE DEFINIT**

- Aplicatia trebuie **sa rezolve o problema**, sa fie **de actualitate** si sa fie **utila** pentru utilizatorii finali

- Trebuie sa aiba integrari de API-uri, dar sa existe si contributie originala

## Alegerea tuturor componentelor necesare implementarii

### Baza de date:

- Alegeti un sistem de gestiune a bazelor de date (DBMS) potrivit, cum ar fi MySQL, SQL Server, PostgreSQL, MongoDB, etc, in functie de aplicatia dezvoltata, dar si de experienta personala

### Backend:

- Alegeti un limbaj de programare si un framework potrivit pentru dezvoltarea backend-ului aplicatiei (cum ar fi Python cu Django, PHP cu Laravel, C# cu ASP.NET)

### Frontend:

- Utilizati tehnologii precum HTML, CSS si JavaScript (se poate utiliza si integra Bootstrap) pentru a crea interfata grafica a aplicatiei. De asemenea, framework-urile populare includ Angular, React, Vue.js, etc.

## Design responsive:

- Asigurati-va ca aplicatia functioneaza pe diferite dispozitive si dimensiuni de ecran, adaptandu-se in mod corespunzator

## Securitatea:

- Implementati masuri de securitate, cum ar fi criptarea datelor, protejarea impotriva atacurilor XSS si SQL Injection si folosirea unui certificat SSL pentru a asigura comunicarea securizata intre server si utilizator

## API-uri si servicii externe:

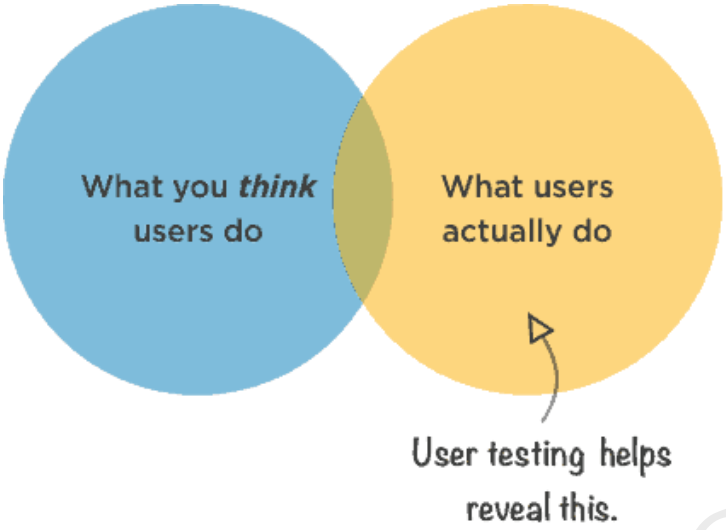
- Integrati API-uri si servicii externe (3-rd party) (ex: Google Maps, servicii de plati online, sisteme de autentificare, etc)

## Versionarea și controlul codului sursa:

- Folositi un sistem de versionare a codului, cum ar fi Git, pentru a va asigura ca aveti mai multe versiuni ale codului sursa si pentru a putea face modificari facile, revenind oricand la una din versiunile anterioare

## UI/UX Design:

- Acordati atentie **design-ului interfetei** raportandu-va intotdeauna la **experienta utilizatorilor finali**, folosind principii de design centrate pe utilizator si testand aplicatia cu **utilizatori reali** pentru a identifica si imbunatati eventualele probleme




- **Simplitatea** – prea multa informatie intr-o pagina distrage atentia utilizatorului deoarece acesta trebuie sa citeasca si sa parcurga mult prea multa informatie pana la informatia de care are nevoie.

- **Culorile sunt foarte importante** – atunci cand alegeti culorile folositi o paleta consistenta de culori. De asemenea, sa va asigurati ca nu exista nuante foarte apropiate care nu pot fi deosebite si mai ales ca exista un contrast puternic intre text si fundal.
  **(Tehnica 60-30-10)**
  https://flatuicolors.com/

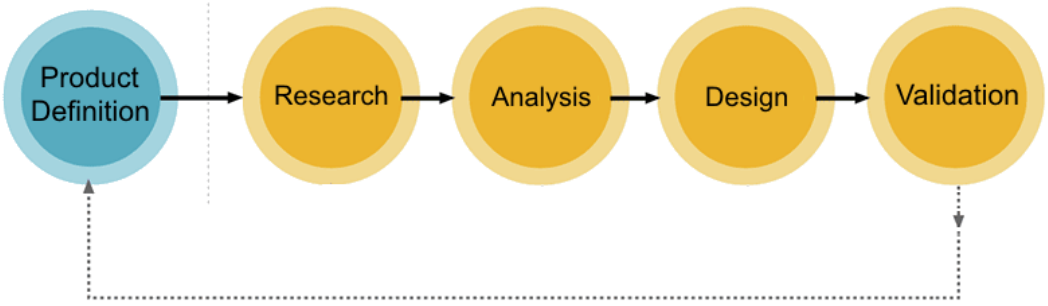
- **Navigarea ar trebui sa fie intuitiva** – un utilizator **NU** trebuie sa caute ceea ce doreste sa acceseze. Paginile trebuie sa fie bine organizate cu un design de tipul top-down, utilizatorii navigand usor printre diferitele sectiuni existente intr-o pagina

- **Consistenta este extrem de importanta** – utilizatorii nu ar trebui sa aiba sentimentul ca viziteaza un alt site sau o alta aplicatie web de fiecare data cand acceseaza o alta pagina a aplicatiei.

- **Utilizarea unui continut real in momentul dezvoltarii design-ului** – orice aplicatie este bazata pe continut si se dezvolta in jurul continutului

- **Fontul** – alegeti fonturi fara finisaje decorative si usor de citit. Dimensiunea fontului pentru citirea cu usurinta este de 16px

- **UX depinde de fiecare proiect in parte** – nu exista un UX general care poate fi aplicat oricarui proiect. De exemplu, atunci cand dezvoltati un nou proiect este necesara alocarea unui timp suplimentar pentru a cerceta ce tipuri de utilizatori vor accesa aplicatia, dar si care sunt specificatiile aplicatiei




- **Validari** si **mesaje de informare si eroare** pentru utilizatorii finali

# Gestionarea eficienta a taskurilor si a timpului

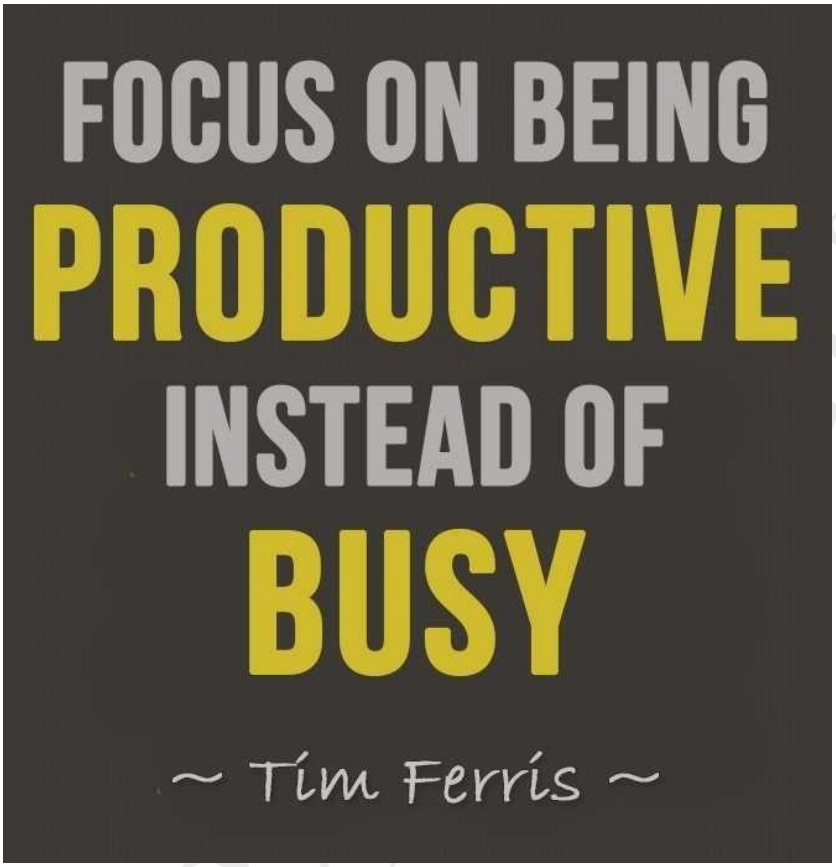
- Pentru **gestionarea eficienta a task-urilor** se utilizeaza https://trello.com/ sau https://www.atlassian.com/software/jira

- Se impart task-urile in functie de ordinea logica a implementarii si se estimeaza timpul pentru fiecare

- Se aloca un timp saptamanal. Se pot stabili sprinturi de 2 saptamani pentru care se aloca un numar de task-uri si se estimeaza timpul alocat implementarii fiecarui task

## Ce trebuie sa evitam?

## Ce trebuie sa facem?



# Redactarea documentatiei

**(*) Informatiile prezentate nu fac parte dintr-un regulament. Sunt informatii generale la care se adauga si cerintele fiecarui profesor coordonator in parte.**

**La inceputul lucrarii** o sa existe si un rezumat al lucrarii, atat in limba romana, cat si in engleza, de aproximativ o pagina unde se prezinta pe scurt tema, importanta ei, contributia originala, aspectele importante, metodele utilizate (cele mai importante – algoritmi, implementarile importante, rezultatele obtinute).

## Cuprinsul

## Introducerea – Capitolul 1

- **Motivatia** – de ce ati ales sa dezvoltati aplicatia respectiva (scopul)
- **Prezentarea pe scurt a temei** punand accentul pe beneficiile pe care le aduce platforma (ce probleme rezolva)
- **Contributia originala** consta in descrierea pe scurt a functionalitatii/functionalitatilor originale (contributia proprie)
- **Structura lucrarii** care sa contina descrierea succinta a fiecărui capitol. Se pot enumera tehnologiile folosite, dupa care se descriu pe scurt capitolele care urmeaza, realizandu-se astfel o introducere pentru ceea ce urmeaza sa fie prezentat

## Concepte teoretice – Capitolul 2

Prezentarea tehnologiilor, arhitecturilor folosite, API-urilor, etc (notiunile stiintifice si tehnologiile care stau la baza temei):

- Despre limbajul folosit, avantajele limbajului utilizat (securitate, flexibilitate, limbaj inteligibil, etc)
- Descrierea conceptelor OOP si a design pattern-urilor (ex: arhitectura MVC)
- Descrierea librariilor si a framework-urilor folosite

Descrierea tuturor componentelor (protocoale, limbaje, arhitecturi, framework-uri, servicii externe, retele neuronale, computer vision, procesare de imagini, clasificatori de machine learning, etc)

**!** Pot fi chiar si doua capitole diferite (**Concepte teoretice** si **Tehnologii folosite**).

## Prezentarea aplicatiei - Capitolul 3

- **Prezentarea problemei**
  - scopul aplicatiei – ce probleme rezolva
  - de ce este un avantaj utilizarea ei
  - de ce este necesara o astfel de platforma si ce imbunatatiri adduce
- **Prezentarea in detaliu a aplicatiei**
  - functionalitatea
  - componentele

- **Cazuri de utilizare** (rolurile din aplicatie)
  - din perspectiva utilizatorului anonym
  - din perspectiva utilizatorilor inregistrati
  - din perspectiva administratorului

- **Scenarii de Utilizare** (nu toate – cele mai interesante)
  - cum se utilizeaza si cum functioneaza componentele existente in aplicatie

- **Structura interna a aplicatiei**
  - se detaliaza arhitectura aplicatiei, modelele din baza de date (Modelul User, Admin, Profile…)

- **Librariile folosite (componentele externe)**
  - ele sunt componente 3rd party – adica acele componente care pot fi integrate (de exemplu: sisteme de notificare prin e-mail, sisteme e-commerce, sisteme de geolocare, de incarcare video, etc)

## Concluzii - Capitolul 4

- concluzii generale
- imbunatatiri si propuneri viitoare

## Bibliografie – nu este capitol

- contine toate referintele folosite in elaborarea lucrarii
- referintele se ordoneaza alfabetic dupa numele primului autor

Referintele bibliografice trebuie numerotate cu [1], [2], …

Surselor bibliografice tiparite (carti) se scriu inaintea surselor preluate de pe internet (pagini web). Nu sunt acceptate referinte fara autor.

De aceea, referintele de pe Internet trebuie sa fie **referinte stiintifice** (documentatiile limbajelor/framework-urilor, **google scholar** – aici se gasesc lucrari stiintifice de unde puteti prelua notiuni teoretice).

**Pentru sursele tiparite se poate utiliza urmatorul stil de citare:**

[1] Nume Autor, Titlul Cartii, Numar de Pagini, Editura, Anul aparitiei

Ex: [1] Michael Hartl, The ruby on rails tutorial 3rd edition Learn web development with Rails, 744 pagini, Addison-Wesley Professional, 2015

**Pentru sursele web:**

[2] Wikipedia [23 martie 2023], scurta descriere (ex: definitie retea neuronala). Preluata din sursa: http//Wikipedia…/

## Dupa Bibliografie se pot adauga Anexe:

De exemplu, Anexa 1 poate contine diagrama bazei de date, secvente de cod (cele care au o dimensiune semnificativa), algoritmi, etc. Fiecare anexa trebuie mentionata cel putin o data in lucrare.

Paginile se numeroteaza incepand cu pagina de titlu, pana la ultima pagina, dar numarul paginii trebuie sa apara doar incepand cu Introducerea.

Lucrarea se redacteaza utilizand Times New Roman, font de 12 pct, spatiere 1.5, margini de 2.5 cm.

Folosirea diacriticelor este obligatorie.

Fiecare capitol trebuie sa inceapa pe o pagina noua (capitol **NU** subcapitol).

# Pregatirea demo-ului aplicatiei

# Pregatirea prezentarii

- Aplicatia in sine impreuna cu prezentarea ppt se prezinta intr-un interval de 15-20 de minute
- Prima data se prezinta pe scurt ppt-ul
- Urmand prezentarea propriu-zisa a aplicatiei
- In final sunt alocate 5-7 minute pentru intrebari
- Puneti accent pe lucrurile cu adevarat importante
- Scoateti in evidenta contributia originala

# Incadrarea in timp in momentul sustinerii

- Alocati un interval de timp fiecarei prezentari (ppt + demo)
- Repetati prezentarea de cateva ori, urmarind si imbunatatind incadrarea in timp