



UNIVERSITATEA TEHNICĂ
DIN CLUJ-NAPOCA

DOCUMENTAȚIE

Proiectare Software

Final Assignment

Photography Album Site Application

Boloș Andrei Nicolae
GRUPA: 30236

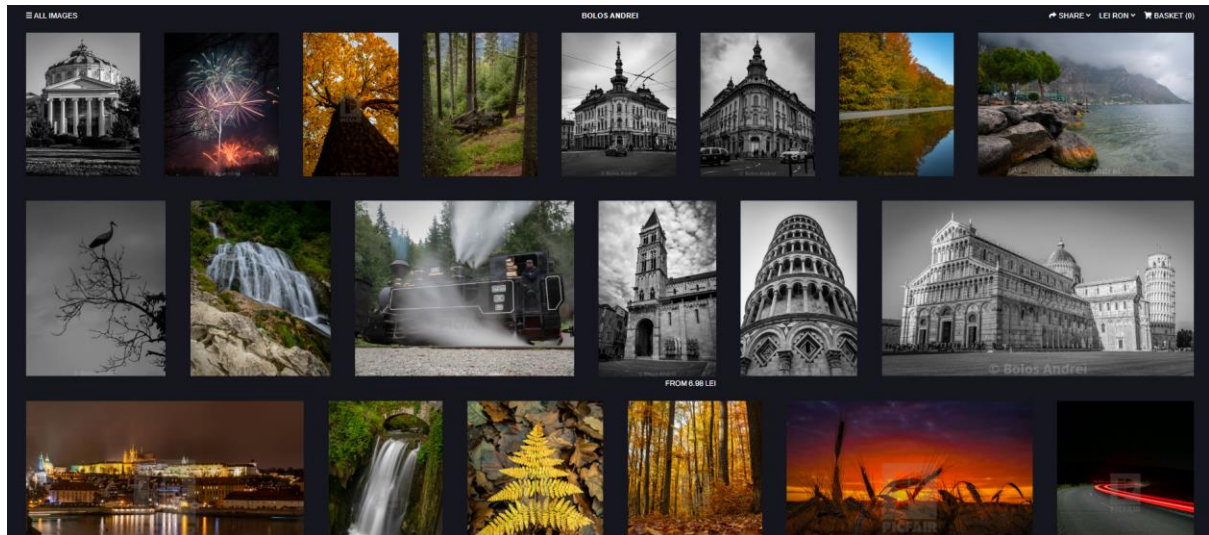
CUPRINS

DOCUMENTAȚIE.....	1
CUPRINS	2
1. <i>Project Specification</i>	3
2. <i>Functional Requirements</i>	5
3. <i>Use Case Model</i>	5
4. <i>Supplementary Specification</i>	7
5. <i>Bibliografie</i>	Error! Bookmark not defined.

1. Project Specification

Obiectivul principal al aplicației reprezintă proiectarea și implementarea unui site pentru Afișarea unui album de imagini ale unor utilizatori, după ce aceștia s-au conectat cu success, într-o aplicație Web care se bazează pe Java.

<Imagine de inspirație: PicFair <https://bolosh.picfair.com>>



Rezultat:

Login Register Create_New_Photo Photo_List Sorted_Photo_List Filtered_Photo_List LogOff

Login

Tell a Thought

Email

Submit

Password

Submit

Register

Login Register Create_New_Photo Photo_List Sorted_Photo_List Filtered_Photo_List LogOff

Register

Name

Email

Password

☐ I'm not a robot

Submit

RECAPTCHA

Login Register Create_New_Photo Photo_List Sorted_Photo_List Filtered_Photo_List LogOff

User List

Id	Name	Email	Password	Admin?	IsLoggedIn?	
1	admin	admin@admin.com	◆◆◆◆e◆◆◆◆□◆◆◆^◆pg◆	true	true	Update ?Admin
2	User1	Email1@email1.com	◆◆□□◆i6'◆◆?P◆◆◆	true	false	Update ?Admin
3	User2	Email2@email2.com	DR07◆◆5◆□□◆l0y◆	false	false	Update ?Admin

All Rights reserved 2023 @Bolosh Andrei

3

[Login](#)
[Register](#)
[Create_New_Photo](#)
[Photo_List](#)
[Sorted_Photo_List](#)
[Filtered_Photo_List](#)
[LogOff](#)

Photo List


Generate PDF

Id	Name	Path	Description	User ID	Actions
1	Photo1	/assets/BOL_4929-16x9.jpg	First Photo	2	<div>Display</div> <div>Update</div> <div>Delete</div>
2	Photo2	/assets/BOL_7158.jpg	Second Photo	3	<div>Display</div> <div>Update</div> <div>Delete</div>
3	Photo3	/assets/BOL_8065-2.jpg	Third Photo	2	<div>Display</div> <div>Update</div> <div>Delete</div>
4	Photo2	/assets/BOL_7158.jpg	Second Photo	3	<div>Display</div> <div>Update</div> <div>Delete</div>
5	Photo2	/assets/BOL_7158.jpg	Second Photo	3	<div>Display</div> <div>Update</div> <div>Delete</div>
6	Photo2	/assets/BOL_7158.jpg	Second Photo	3	<div>Display</div> <div>Update</div> <div>Delete</div>
7	Photo2	/assets/BOL_7158.jpg	Second Photo	3	<div>Display</div> <div>Update</div> <div>Delete</div>
8	Photo2	/assets/BOL_7158.jpg	Second Photo	3	<div>Display</div> <div>Update</div> <div>Delete</div>

[Login](#)
[Register](#)
[Create_New_Photo](#)
[Photo_List](#)
[Sorted_Photo_List](#)
[Filtered_Photo_List](#)
[LogOff](#)

Generate PDF

Photo Id: 1
Name: Photo1
Path: /assets/BOL_4929-16x9.jpg
Description: First Photo
User Id: 2



All Rights reserved 2023 @Bolosh Andrei

[Login](#)
[Register](#)
[Create_New_Photo](#)
[Photo_List](#)
[Sorted_Photo_List](#)
[Filtered_Photo_List](#)
[LogOff](#)

Add new Photo

Name

Path

Description

User Id

Photo

Choose File

No file chosen

upload image

No file chosen

upload

Get Image

Submit

All Rights reserved 2023 @Bolosh Andrei

[Login](#)
[Register](#)
[Create_New_Photo](#)
[Photo_List](#)
[Sorted_Photo_List](#)
[Filtered_Photo_List](#)
[LogOff](#)

Update Photo

Name

Photo1

Update Name

Description

First Photo

Update Description

Path

/assets/BOL_4929-16x9.jpg

Update Path

User Id

2

Update User

All Rights reserved 2023 @Bolosh Andrei

2. Functional Requirements

În funcție de *user*, atributul de admin se setează din back-end pe *true* doar la persoanele autorizate cu access la codul sursă.

Utilizatorul normal are disponibile următoarele operații:

- ➔ Operații pe baza de date pentru *User* – CRUD;
- ➔ Operații pe baza de date pentru *Photo* – CRUD;
- ➔ Acțiunea de a cumpăra o poză;
- ➔ Trimitere mail utilizatorului a cărui poză se intenționează a fi cumpărată;
- ➔ Trimitere mail de confirmare a comenzii utilizatorului a care are intenția de a cumpăra o poză;
- ➔ Ștergerea unui *user* normal de către un *admin*.

3. Use Case Model

Use-Case:

Level:

Primary Actor:

Main success scenario:

Extensions:

Use-Case1: Register

Level: User-Goal

Primary Actor: User

Main success scenario: Successful register

Extensions: Format ilegal de date

Use-Case2: LogIn

Level: Subfunction

Primary Actor: User

Main success scenario: Successful login

Use-Case3: LogOut

Level: Subfunction

Primary Actor: User

Main success scenario: Successful logout

Use-Case4: Delete/ Add a Photo

Level: User

Primary Actor: User

Main success scenario: Userul apasă butoane care adaugă poze sau le sterge din baza de date. Se selectează poza din File Explorer

Use-Case5: Buy a Photo

Level: Client-Goal

Primary Actor: User

Main success scenario: Un user se uita în lista de poze ale altui user, dă hover, se afișează prețul și opțiunea de a Cumpara

Extensions: clientul nu are suficient monetar în cont, rezultă în mesaj de eroare.

Use-Case6: Delete a user

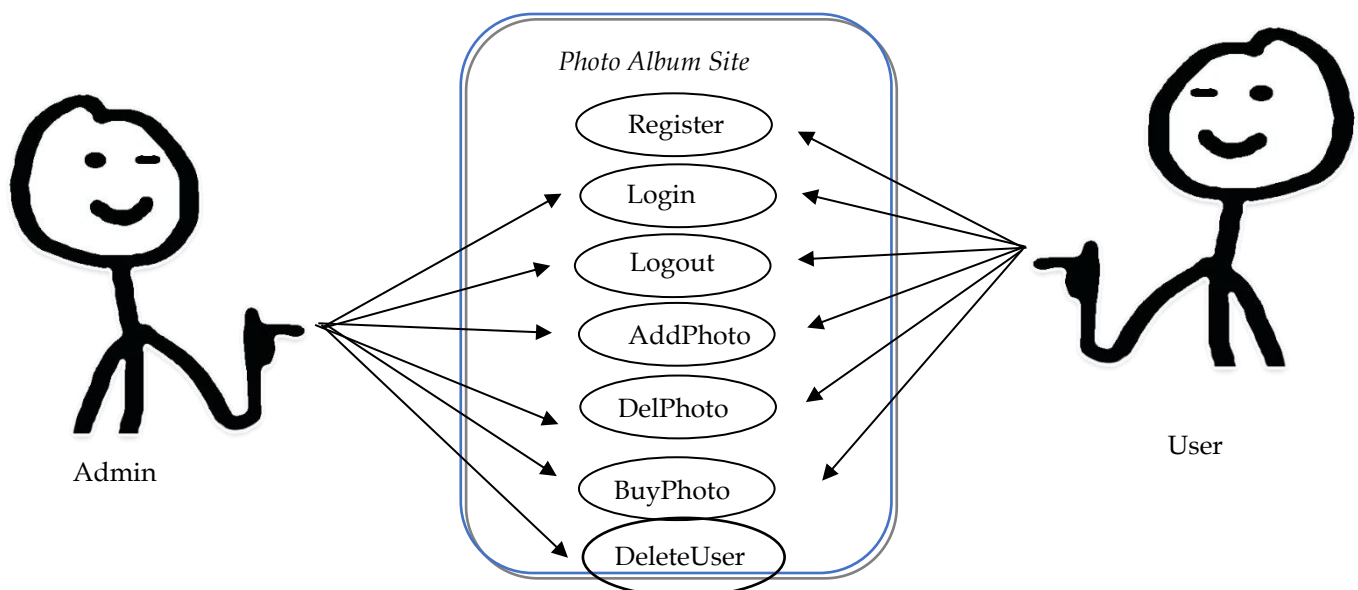
Level: admin

Primary Actor: admin

Main success scenario: Adminul poate șterge din baza de date. Se selectează userul dintr-o listă de useri.

Diagrama UML (Unified Modeling Language) reprezintă structura logică a proiectului. Specifică împărțirea în pachete, clase și relații. Ea poate conține și atributele, constructorii și metodele claselor.

Diagrama Usecase:



4. *Supplementary Specification*

Non-Functional Requirements:

- Portabilitate – poate functiona pe diferite device-uri, cu diferite rezolutii;
- Usability: user friendly interface easily navigable
- Security: password encryption pt a face dificil furtul de date;
- Internationalizare si localizare: website ul poate fi tradus in mai multe limbi;

Design Constraints:

Partea de backend a proiectului o sa fie conceputa cu ajutorul limbajului Java si Spring, parte de frontend o sa fie facuta in React iar Baza de date o sa fie implementata si stocata cu ajutorul aplicatiei MySQL, simplificarea codului va fi realizata cu ajutorul Lombok, maparea intre baza de date si java se face cu JPA.

IDE-ul folosit ieste IntelliJ. In cadrul proiectului o sa fie folosita o arhitectura stratificata care o sa respecte principiile SOLID.

O sa fie cinci mari module: Model, Repo, UI, Controller si Service, care vor avea fiecare multiple clase.

5. *Domain Model*

Modelele principale ale proiectului sunt *User*-ul și *Photo*-ul.

Un *User* este văzut ca o persoană care poate deține niște poze, iar poza este obiectul principal al acestei expoziții.

Această afirmație se poate traduce în implementarea aplicației ca fiind necesitatea a două entități: ***Photo***, ***User***.

O ***Poză*** poate avea următoarele attribute:

- *id* – un identificator unic al pozei;
- *name* – un nume sugestiv;
- *description* – o descriere sugestivă;
- *path* – calea de referință către zona de memorie unde este stocată poza;
- *user* – poate fi atribuit si un deținător al pozei.

Un ***User*** poate avea următoarele attribute:

- *id* – un identificator unic al userului;
- *name* – un nume sugestiv;
- *email* – un email cu care se loghează;
- *password* – o parolă cu care se loghează;

- admin – determină dacă acest user are drepturi de *admin* sau nu;
- photoList – o listă cu pozele pe care acest user le deține.



6. Architectural Desing

- *Conceptual Architecture*

Arhitectura are la bază structura pe nivele – Layered Architecture.

a. Stratul de prezentare (**Presentation Layer**)

Acest strat reprezintă partea vizibilă de către utilizator și are rolul de a prelua datele de intrare din interfața grafică și de a le trimite mai departe către următorul nivel.

Pentru acest nivel s-au folosit tehnologii precum HTML, CSS, Angular/React.

Această aplicație poate să ruleze pe desktop sau pe web, dar este implementată doar rularea într-un browser.

b. Stratul de acces la date (**Data Access Layer**)

Acest strat are rolul de a prelua datele din interfața grafică și le stochează în baza de date și le trimite mai departe pentru prelucrare ulterioară. În baza de date sunt stocate modelele aplicației.

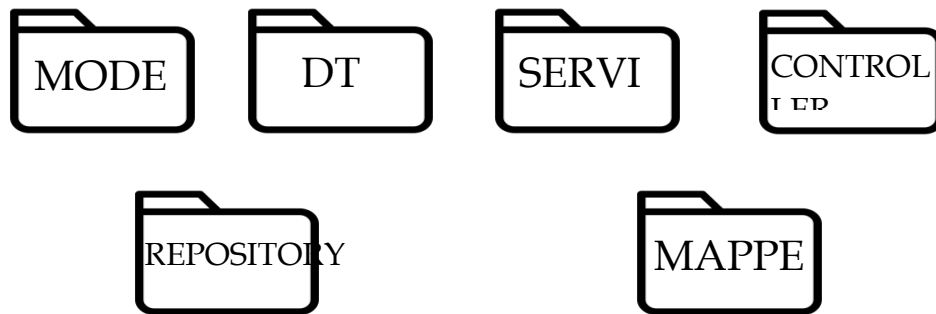
c. Stratul de logică (**Business Layer**)

Acest strat se ocupă cu prelucrarea datelor ce sunt stocate în baza de date și trimiterea lor spre afișare.

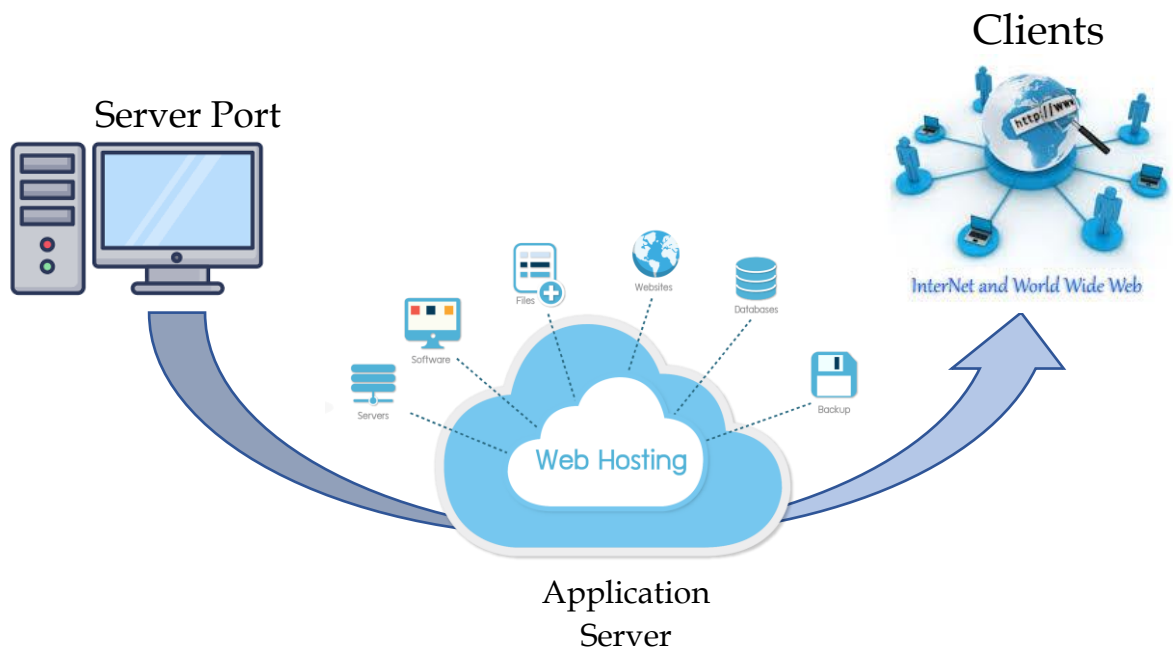
Aici se află detaliile legate de implementare, precum implementarea funcțiilor de acces la baza de date, de modificare a proprietăților fiecărui obiect și diferite sarcini.

Arhitectura se poate adapta după cerințe proaspăt primite sau actualizate.

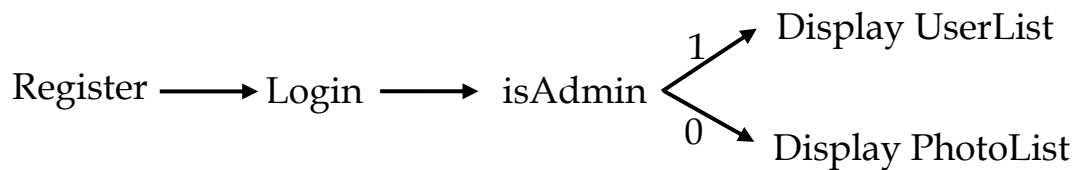
- *Package Design*



- *Component & Deployment Diagrams*



7. Design Model



- **Scenario 1:**

Dacă un user nu este înregistrat în baza de date, atunci se poate înregistra cu o adresă de email, un nume și o parolă. Parola este criptată.

Odată înregistrat în baza de date, acest user poate să dea Login folosind email-ul și parola introduse la înregistrare.

Dacă acest user are statutul de Admin, atunci el poate vedea o listă cu toți utilizatorii existenți în baza de date.

- **Scenario 2:**

Dacă un user nu este înregistrat în baza de date, atunci se poate înregistra cu o adresă de email, un nume și o parolă. Parola este criptată.

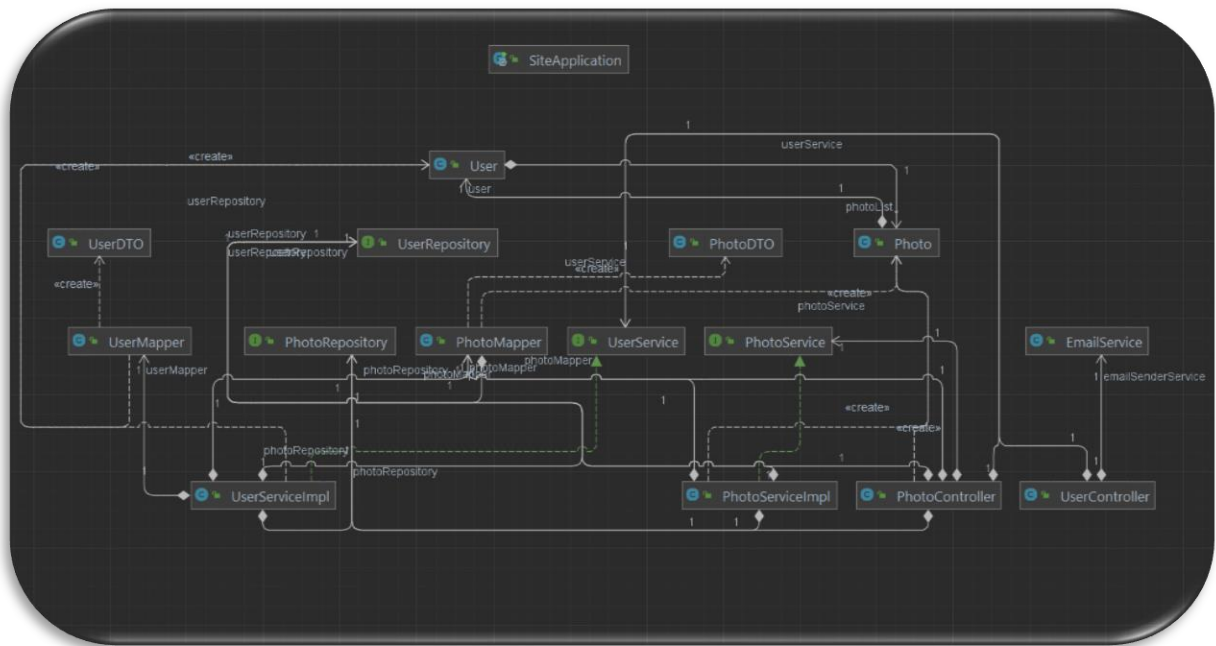
Odată înregistrat în baza de date, acest user poate să dea Login folosind email-ul și parola introduse la înregistrare.

Dacă acest user nu are statutul de Admin, atunci el poate vedea o listă cu pozele stocate în baza de date.

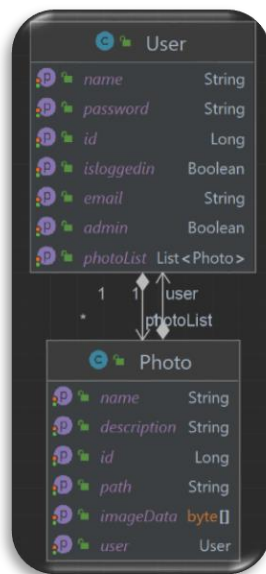
Aceste poze se pot modifica, șterge sau se pot afișa detaliat.

Lista se poate sorta crescător sau descrescător în funcție de ID, sau se poate filtra pentru a permite doar afișarea pozelor ce aparțin unui anumit user.

8. Class Diagram: UML



9. Data Model



Data model-ul aplicatiei acestea este unul relational. Se foloseste o baza de date ca sa se stocheze entitatile principale: User si Photo.

Relatia dintre User si Photo este OneToMany, deoarece un user poate avea o lista de poze.

10. System Testing

Partea de testare reprezinta validarea corectitudinii functionalitatilor si a implementarilor acestora.

Exista testare automata a sistemului, fiind implementate teste unitare ce verifica functionalitatea operatiilor pe baza de date: Create, Read, Update, Delete.

Majoritatea testării s-a făcut verificând comportamentul aplicației live, adică la fiecare actualizare s-a verificat aspectul vizual, corectitudinea rezultatelor și soluționarea erorilor care apăreau mai mult sau mai puțin neprevăzut.

Testarea manuală a reprezentat baza verificării faptului că aplicația merge sau nu cum a fost intenționat să funcționeze.

11. *Future Improvements*

Pentru ca aplicația să poată fi folosită la următorul nivel, eventual acesta ar putea fi chiar un nivel de producție, aceasta trebuie să fie pusă la punct din mai multe puncte de vedere:

- Securitate;
- Accesibilitate & Ușurință de folosire;
- Aspect vizual;
- Utilitate;
- Noi funcționalități, precum posibilitatea de tranzacții între utilizatori.

12. *Conclusion*

Acest proiect a avut un rol important în dezvoltarea mea pe plan profesional, deoarece am impresia că acest tip de proiecte face parte din probabil cea mai utilizată ramură din domeniul IT.

Aș fi dorit să depun mai mult timp acestui proiect, încât să poată fi utilizat mai mult decât cu scop educativ.

13. *Bibliografie*

Link Git:

Link Resurse:

<https://www.javaguides.net/2019/07/spring-data-jpa-save-findbyid-findall-deletebyid-example.html>

<https://www.youtube.com/watch?v=tLBX9fq813c&list=PLGRDMO4rOGcNzi3CpBWsCdQSzbjWWy-f&index=1>

inspirație efecte: <https://www.soaringcloudsrecords.com>

<https://www.javainuse.com/fullstack/imageupload>

<https://medium.com/shoutloudz/spring-boot-upload-and-download-images-using-jpa-b1c9ef174dc0>

tutorial how to upload an image:

<https://www.javainuse.com/fullstack/imageupload>

page up scroll button:

https://www.w3schools.com/howto/howto_js_scroll_to_top.asp

websocket:

<https://www.youtube.com/watch?v=A3zBohNgHKE>

Export HTML table to PDF

<https://www.youtube.com/watch?v=Kik1SvebqTg>

Export XML

https://www.youtube.com/watch?v=koYZ_xS8mf0

Encriptare parola

<https://www.youtube.com/watch?v=a3ZdcDvUwdY>