Ministerul educației al Republicii Moldova

Universitatea de Stat din Moldova

Facultatea de Matematică și Informatică

**DAREA DE SEAMĂ**

Lucrare independentă la ” HTML, CSS, JAVASCRIPT”

A efectuat: Țurcanu Cristian (IA 1901)

A verificat: Tatiana Pașa

**CHIȘINĂU, 2020**

**Introducere**

În primul rând am ales tipul siteului pentru prezentarea unui produs. Ca produs am ales o armă din seria de jocuri video Fallout, care se caracterizează cu un stil retrofuturistic, bazat pe stilul anilor 45-50 din SUA. Această serie se începe cu o catastrofă nucleară: întreaga lume este acoperită de focul bombelor atomice, iar singurii supraviețuitori sunt cei ce au reușit să ajungă în bunkere, vândute ca produse de rând, care se vor deschide cu sute de ani după catastrofă. Datorită acestui fapt, această serie combină umorul corporațiilor despre catastrofa iminentă și mesajul anti-război a populației de rând.

Propriu-zis arma este o modificație pentru jocul video Fallout 4, care introduce o reconstruire a armei classice Wattz 2000 din Fallout 1 și 2. Această armă era folosită în cadrul forțelor armate SUA, și datorită rezistenței acesteia era considerată ca un AK a armelor cu laser, fiind folosită și secole după catastrofă.

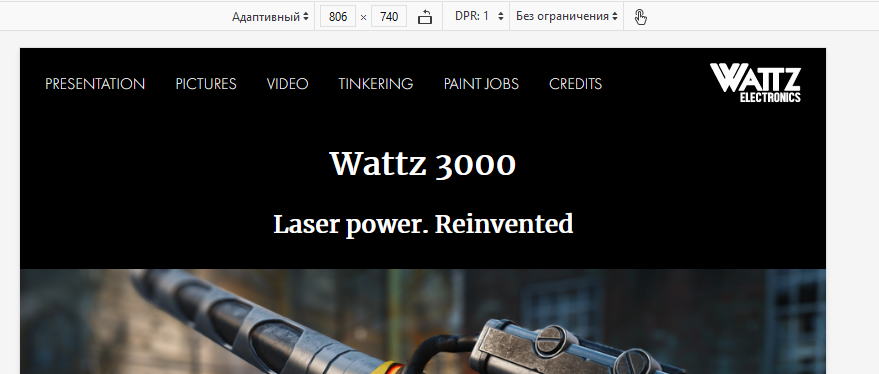
Mai multe detalii despre seria Fallout:  
 [Fallout 4 - Intro](https://youtu.be/4NkHQs7ann4)

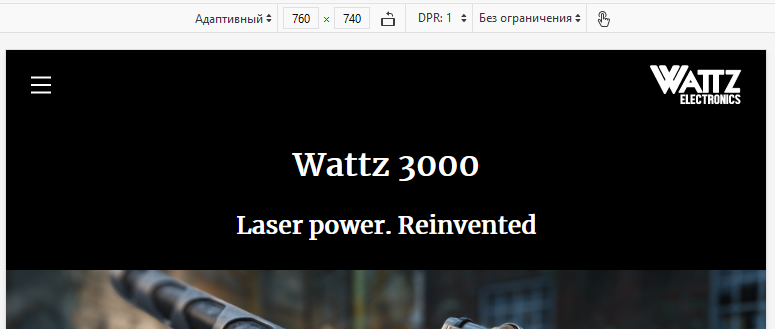
[Fallout 1 - Intro](https://youtu.be/geLiEiAiQJA?t=15)

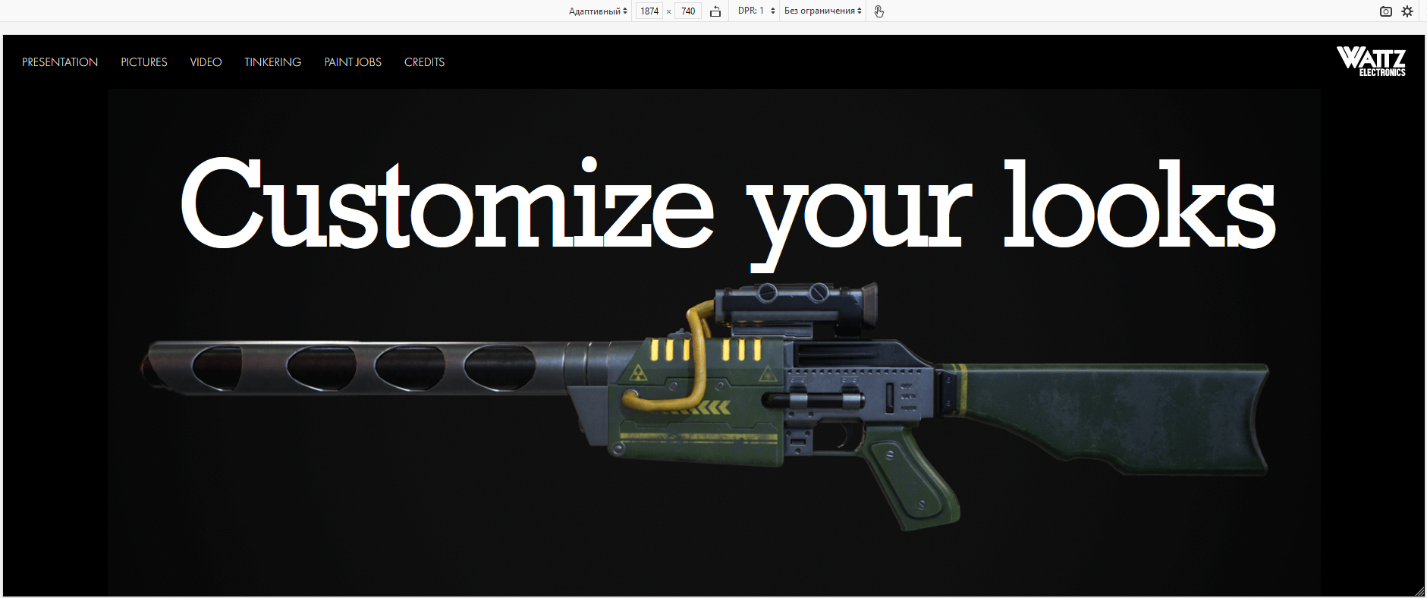
**Comentarii**

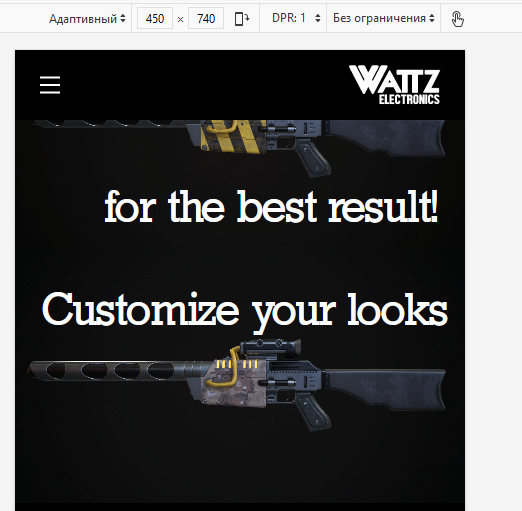
Ca scop personal am prioritizat folosirea cât mai redusă a scripturilor JS și înlocuirea acestora cu proprietăți CSS. Mai mult ca o chestie personală aș vrea ca site-ul să fie în mare parte funcțional și fără scripturi sau cu acestea fiind blocate (de exemplu – prin modificația NoScript a browserilor). De asemenea acest fapt ar minimiza resursele CPU folosite de site, care ar permite vizitarea acestuia de majoritatea device-urilor.

Ca website-ul dat să fie adaptiv a fost necesară schimbarea unor tactici de lucru. În primul rând, header-ul își schimbă conținutul în dependență de rezoluție. În cazul în care viewportul are un width mai mare sau egal cu 768px, acesta va ține linkurile ca de obicei. În cazul în care acesta este mai mic de 768px, linkurile se ascund sub un hamburger menu, care se deschide/închide în baza unui script JS. Scriptul pur și simplu schimbă proprietatea elementelor în display: none sau display: flex.



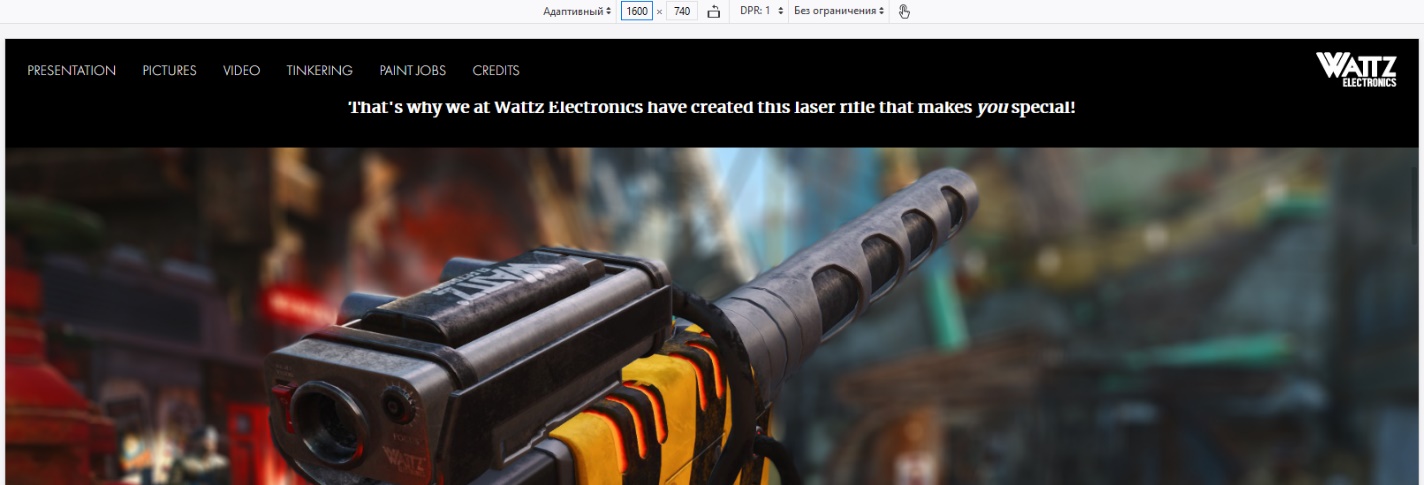


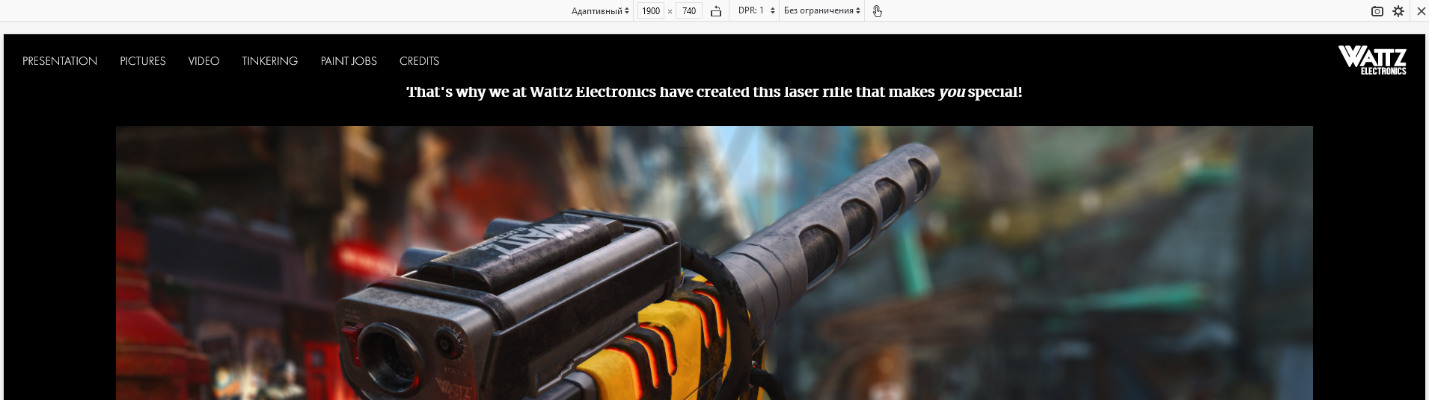
În continuare, majoritatea elementelor își au mărimile sau pozițiile setate prin procente sau alte unități ce depind de rezoluție, astfel este asigurată amplasarea identică a elementelor la rezoluții diferite. Spre exemplu, gif-urile din sfârșitul paginii sunt incluse ca background-image a div-ului container, iar textul este setat în interiorul div-ului prin position: relative; și setarea parametrilor top și left.  




Desigur, o metodă mai simplă ar fi includerea textului direct în gif-ul dat prin editoare de imagini, însă aici am vrut să încerc anume situarea adaptivă și manuală a textului.

O altă problemă legată de adaptivitate am întâlnit în cazul ecranelor cu un aspect ratio mare. La început, toate imaginile aveau un width de 100%, fapt care permite ocuparea cu folos a întregului ecran și privirea imaginilor în detalii. De asemenea, toate imaginile au un aspect ratio de 16:9, identic standartelor HD, FullHD, 1440p și 4K a ecranelor desktop, adică totul trebuie să încapă perfect. Problema a apărut prin faptul că ecranul nu este întotdeauna liber, spre exemplu, jos este prezent taskbar-ul în sistemele Windows și MacOS, iar sus este prezent header-ul browserului. Acest fapt micșorează înălțimea viewportului, lăsând lățimea neschimbată. Din cauza acestui fapt, orice imagine întinsă la 100% nu va putea fi văzută complet pe ecranele cu un aspect ratio de 16:9 (nemaivorbind de standartele ultrawide). Pentru a rezolva acest fapt, am inclus tot content-ul paginii într-un container, max-width-ul căreia a fost setat ca 1600px. Acest fapt măcar va asigura un layout comod în cazul ecranelor cu rezoluția FullHD. Un caz ideal ar fi să se limiteze max-width-ul contentului în dependență de înălțimea viewportului. Spre exemplu, păstrarea unui aspect ratio minim a contentului de 4:3 va permite o vizualizare comodă a paginii. Desigur vor fi prezente coloanele negre fără content, însă acesta ar fi un compromis împotriva imaginilor ce nu încap. De asemenea, păstrarea unui aspect ratio ar fi logică doar în cazul ecranelor calculatoarelor, și spre exemplu nu ar trebui să se aplice în cazul modului orizontal a telefoanelor și tabletelor.





Mai departe am folosit un slideshow dintr-o lucrare de laborator. Am modificat script-ul ca event handler-urile sa fie atribuite în cadrul inițierii paginii, și nu manual în documentul html. De asemenea am folosit și event-ul onClick pe lângă onMouseEnter, ca scriptul să lucreze și pe telefoane. Am mai folosit și proprietatea overflow-x: scroll în flex container ca să apară un scroll sub container și să fie posibil scroll-ul pe telefoane. Această proprietate se activează doar când lățimea viewportului este mai mică decât lățimea containerului, în caz contrar este folosit overflow-x: auto. Această schimbare permite browserului Chrome să ascundă scrollbar-ul când acesta nu este necesar, iar browserul Firefox va arăta scrollbarul sub container și nu dedesubtul paginii.

De asemenea am folosit fonturi specializate, care se descarcă împreună cu pagina la încărcare. Am mai împărțit toate scripturile CSS în mai multe documente în bază de categorii și elemente, ca mai ușor să fac debuggingul stilurilor. În continuare am încercat folosirea imaginilor vectoriale de format SVG în cadrul paginii (logotipul și hamburger menu). Probleme de compatibilitate am găsit doar într-un default browser vechi pentru Android, în care textul “ELECTRONICS” din logotip nu era arătat cu fontul corect.

**Concluzii**

Personal, dacă aș mai avea timp, aș mai încerca încă o dată să reconstitui modelul 3d în blender și să fac renderuri realistice personale. Prima dată când am vrut să reconstruiesc modelul am întâlnit o diferență considerabilă dintre imaginile autorului și ale mele. De asemenea, modelul reutilizează texturi din cadrul jocului pentru a economisi spațiu, însă eu nu am putut să găsesc texturile date. Exemplu:

Renderul meu personal



Renderul autorului

În planul contentului aș mai vrea să adaug materiale propriu-zis filmate în cadrul jocului, însă din cauza videocartei vechi, pot porni jocul doar cu setarea unui nivel jos a graficii, care ar face contentul personal de o calitate mai rea ca imaginile sau videourile autorilor.

De asemenea, aș introduce limitarea content-ului în bază de aspect ratio. Acest proces poate lua mai mult timp din cauza refacerii div-urilor cu imagini animate ca background. De asemenea, detectarea dacă site-ul este deschis pe calculator, telefon sau tabletă fără folosirea comparării rezoluțiilor (în zilele de azi, ecranul unui telefon mai nou poate avea o rezoluție mai mare ca ecranul unui laptop/calculator) poate fi făcută doar prin intermediul scripturilor JS, care pot lua mai mult timp pentru creare și debugging.