|  |  |
| --- | --- |
| Obrázok, na ktorom je stôl, znak, kreslenie  Automaticky generovaný popis | SOŠ StrojNÍCKA Partizánska cesta 76  957 01 Bánovce nad Bebravou |
|  |  |

Náučná Kyberbezpečnostná Webstránka

projekt

|  |  |
| --- | --- |
| Bánovce nad Bebravou | Patrik Pirťan |
| 2024 | Ročník štúdia: štvrtý |

|  |  |
| --- | --- |
| Obrázok, na ktorom je stôl, znak, kreslenie  Automaticky generovaný popis | SOŠ StrojNÍCKA Partizánska cesta 76  957 01 Bánovce nad Bebravou |
|  |  |

Náučná kyberbezpečnostná Webstránka

projekt

|  |  |
| --- | --- |
| Riešiteľ: | Patrik Pirťan |
| Študijný odbor: | 3918 M Technické lýceum |
| Katalógové číslo: |  |
| Ročník štúdia: | štvrtý |
| Konzultant: | Ing. Pavel Hazucha |
|  |  |
| Bánovce nad Bebravou | |
| 2024 | |

**Čestné vyhlásenie**

Vyhlasujem, že som prácu vypracoval samostatne a použil som len odbornú literatúru, ktorú uvádzam v zozname použitej literatúry. Moja dokumentácia neobsahuje chránené údaje podniku a/alebo zákazníka a neporušuje autorské práva.

|  |  |
| --- | --- |
| V Bánovciach nad Bebravou, dňa 09.02.2024 | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  Vlastnoručný podpis |

Obsah

[0 Úvod 5](#_Toc158199009)

[1 Problematika a prehľad literatúry 6](#_Toc158199010)

[1.1 Figma - Dizajn a prototypovanie 6](#_Toc158199011)

[1.2 HTML - Štruktúra webových stránok 6](#_Toc158199012)

[1.3 CSS - Estetika a rozloženie webových stránok 7](#_Toc158199013)

[1.4 JavaScript - Interaktivita a dynamika webových stránok 7](#_Toc158199014)

[1.5 Visual Studio Code - Efektívne Vývojové prostredie 8](#_Toc158199015)

[1.5.1 Extensions -Efektívnejšia práca 8](#_Toc158199016)

[1.6 Git – Správa verzií 9](#_Toc158199017)

[1.7 GitHub 9](#_Toc158199018)

[2 Ciele práce 10](#_Toc158199019)

[3 Materiál a metodika 11](#_Toc158199020)

[3.1 Inštalácia a konfigurácia Gitu a Githubu 11](#_Toc158199021)

[3.2 Dizajnovanie vo Figme 12](#_Toc158199022)

[3.2.1 Dizajn domovskej stránky (Index) 13](#_Toc158199023)

[3.2.2 Dizajn podstrániek prihlasovania a registrácie 14](#_Toc158199024)

[3.2.3 Posledné úpravy a dokončenie dizajnu 14](#_Toc158199025)

[3.3 Programovanie a štýlovanie webovej stránky 15](#_Toc158199026)

[3.3.1 Vzorove programovanie v HTML 16](#_Toc158199027)

[3.3.2 Vzorové programovanie v CSS 18](#_Toc158199028)

[4 Závery práce a zhrnutie 20](#_Toc158199029)

[5 Zoznam použitej literatúry 21](#_Toc158199030)

[6 Prílohy 22](#_Toc158199031)

# Úvod

V súčasnom, digitálnom veku, kľúčovú úlohu zohráva kyberbezpečnosť. Rozvojom digitálnej technológie a internetu je nevyhnutné, aby sme sa po internete pohybovali bezpečne, ale aj, aby sme rozumeli potenciálnym technikám a hrozbám, ktoré ohrozujú našu digitálnu bezpečnosť. V práci sa budeme zaoberať dvoma kľúčovými aspektmi kyberbezpečnosti – prevenciou a pochopením techník používanými hackermi. Náš každodenný život je prepojený s internetom napríklad ako komunikáciu, prácu, nákupy alebo zábavu. Zvyšuje sa neustále riziko kybernetických hrozieb napríklad: útoky tretích strán na osobne údaje, heslá a citlivé informácie. Táto práca má za cieľ poskytnúť čitateľovi ucelené informácie, aby sa s maximálnou bezpečnosťou mohol pohybovať po internete, ochrániť svoje digitálne stopy a vyhnúť sa potenciálnym nebezpečenstvám.

Okrem prevencie sa budeme venovať aj rôznym technikám používaných v oblasti hackovania. Je dôležité poznať tieto techniky, aby sme zlepšili prevenciu a lepšie pochopili, ako bezpečne používať technológie. Pár technik, o ktorých budeme hovoriť sú napríklad: „Cross-site-scripting“ (XSS), „Packet Sniffing,“ „Privilige Escalation“ a ďalšie. Cieľom práce je vybaviť čitateľov nástrojmi a znalosťami, ktoré umožnia bezpečnejšie využívanie internetu a lepšie chápanie kyberbezpečnosti.

# Problematika a prehľad literatúry

Táto kapitola slúži na to, aby sme sa oboznámili s programami alebo technológiami, s ktorými sme pri práci na projekte pracovali, alebo ich inak používali k zlepšeniu, alebo uľahčeniu práce. Súčasťou tejto kapitoly bude vysvetlenie dôležitých pojmov alebo skratiek, s ktorými sme sa počas vyhotovenia práce stretli.

## Figma - Dizajn a prototypovanie

Figma je webová aplikácia na návrh dizajnu alebo softvéru. Je to webová aplikácia a cloudová platforma, ktorú môžeme používať v tíme znamená to, že môžeme pracovať v reálnom čase viacerí na projekte. Figme používame na tvorbu grafických návrhov, užívateľských rozhraní (UI) a na prototypovanie.

Je to populárny nástroj, ktorí každodenne používajú dizajnéri na celom svete. Ponúka veľkú flexibilitu a zefektívňuje prácu medzi členmi tímu, ktorí pracujú na rôznych digitálnych projektoch.

## HTML - Štruktúra webových stránok

Hypert Text Markup Language alebo v skratke HTML je hypertextový značkovací jazyk, vďaka ktorému sme schopní vytvárať webové stránky. Je zložený z tagov inak povedané značiek. Tieto značky určujú, ako sa zobrazujú jednotlivé elementy na stránke. Tagy alebo, teda značky sa píšu v lomených zátvorkách <> a rozlišujeme ich na párové a nepárové. Rozdiel je ten, že párové tagy majú úvodnú a aj koncovú značku. Vyzerá to takto - <tag>obsah</tag>. Naopak nepárové sa nemusia nijak ukončiť. To znamená, že majú len ten začiatok <tag>

HTML slúži na zobrazovanie, to znamená, že stránka nie je interaktívna. Nedokážeme pomocou HTML vytvoriť napríklad hru alebo nejakú prezentáciu. Pre to, aby stránka bola interaktívna “dynamická” musíme pridať kód z iného jazyka. Napríklad, ako sme použili aj v práci JavaScript.

## CSS - Estetika a rozloženie webových stránok

CSS alebo Cascading Style Sheets po slovenský kaskádové štýly používame na to, aby sme naším tagom, ktoré sme použili dali nejaký grafický výzor. CSS majú osobitný dokument, ktorí potom prepájame s HTML v hlavičke. CSS ma tak ako HTML jednoduchú konštrukciu. Pre niektorých by mohla byť nevýhoda to, že práve, keď robíme túto konštrukciu tak musíme používať anglické výrazy pre určenie vlastnosti elementu, ktorí chceme zmeniť. Štruktúra vyzerá nejak takto:

**h1{**

**color: red;**

**font-size: 25px;**

**}**

Na príklade vyššie by sme vykonali zmeny v h1 tagu (tento tag sa používa na nadpisy), a to také, že by sme zmenili farbu písma na červenú a veľkosť písma na 25 pixelov.

V CSS môžeme aj trochu animovať. Môžeme na to použiť 2 spôsoby, a to cez transitions alebo animations. Pomocou transitions vytvárame jednoduché efekty napríklad “:hover” tento použijeme vtedy, keď chceme napríklad, aby sa nám zmenila farba pozadia, keď prejdeme myškou cez obrázok. Pomocou animations môžeme vytvárať zložitejšie animácie na toto používame “keyframes”. Tieto animácie sa môžu priblížiť animáciám, ktoré sú vytvorené v programoch, ako je napríklad Adobe Flash alebo After Effect

## JavaScript - Interaktivita a dynamika webových stránok

JavaScript je skriptovací programovací jazyk. Kompletizuje nám svätú trojicu, ktorá sa používa pri vytváraní webových stránok. Má veľa využití, ale používa sa hlavne pri webových stránkach, aby sme ich dokázali urobiť interaktívne. Môžeme vytvoriť napríklad dotazník alebo nejaký test na webovej stránke. Môžeme vďaka nemu vytvoriť takzvanú webovú aplikáciu. Pomocou javascriptu dokážeme aj vytvárať animácie alebo zlepšovať už vytvorené animácie v CSS. Javascript bol prioritne určený na zhotovovanie klientskych častí aplikácií, ale s príchodom nových Node.js a podobných technológií dokáže figurovať aj na strane servera.

Javascript môžeme písať v vlastnom súbore s koncovkou “.js” ale aj v súbore kde je HTML.V prípade, že píšeme javascript v súbore s html musíme vždy písať do tagov určených pre javascript. Tieto sa musia vždy nachádzať na úplnom konci HTML súboru. Napríklad v tomto kóde môžeme vidieť syntax javascriptu a tag, ktorý sa používa na vyhradenie miesta pre javascript.

**<script>**

**function myFunction() {**

**document.getElementById("demo").innerHTML = "Ahoj!";**

**}**

**</script>**

Je dôležité povedať, že JavaScript a Java sú úplne rozdielne programovacie jazyky.

## Visual Studio Code - Efektívne Vývojové prostredie

Softvér, ktorý slúži na editovanie zdrojového kódu. Prostredie tvorí editor je to hlavná časť celého programu. Ovládanie softvéru je ľahké ma rôzne klávesové skratky, ktoré nám uľahčujú prácu. Prácu si môžeme uľahčiť aj rôznymi rozšíreniami (extensions). Môžu pridávať ďalšiu funkcie. Ďalšiu výhodou je, že podporuje systém git. VS code alebo Visual Studio Code má zabudovanú podporu niektorých jazykov ako napríklad JavaScript.

### Extensions -Efektívnejšia práca

Existuje veľké množstvo rozšírení, ktoré vylepšujú funkčnosť alebo prispôsobujú editor tak aby sa nám lepšie pracovalo. Rozšírenia sú rôzne napríklad:

* **vzhľadové** – menia ako editor vypadá, menia font alebo pridávajú ikony pre súbory.
* **kód** – tieto nám pomáhajú písať kód napríklad namiesto toho aby sme písali celí názov funkcie tak nám ju hneď po pár písmenkách ponúkne alebo sa nám automaticky generujú aj koncove tagy.
* **Formátovanie** – takéto rozšírenia nám zase môžu uľahčovať prehľadnosť kódu. Môžu meniť farbu napríklad tagov alebo nejakých blokov.
* **Ostatné** – je mnoho rôznych rozšírení, ktoré robia veľmi rôzne veci. Napríklad sú také ktoré zvýrazňujú komentáre alebo nám púšťajú server.

V práci sme používali najviac LiveServer rozšírenie. Toto rozšírenie nám slúžilo na to aby sme vydelí zmeny na stránke hneď po uložený súboru to znamená, že nemusíme znovu načítať stránku stačí nám len uložiť zmeny.

## Git – Správa verzií

Git je distribuovaný systém pre správu verzií. Používame ho na sledovanie zmien v kóde a správu verzií projektov. Slúži nám na to aby sme efektívne mohli spolupracovať na projektoch. Vďaka gitu môžeme udržiavať prehľad o zmenách v kóde. Podporuje vetvenie (branching), zlučovanie (merging) a sledovanie histórie zmien. Pri gite musíme poznať tieto základne pojmy.:

* **Repozitár (Repository)** – Je to miesto, kde sa ukladá história projektu. Ukladajú sa všetky zmeny, vetvy, značky a súbory. Môže byť lokálny alebo vzdialený. V prípade vzdialeného môže pracovať na projekte viacero ľudí.
* **Komity (Commits)** – Základná jednotka zmien v git repozitári. Reprezentuje sadu zmien v súboroch a má jedinečný identifikátor. Pri každom commite sa vytvára správa o zmene a taktiež odkaz na predchádzajúci commit. Vytvárame tak stromovú štruktúru zmien v repozitári.
* **Vetvy (Branches)** – Vďaka vetvám môžeme pracovať na rôznych verziách kódu nezávisle od tej hlavnej vetvy. Môžeme experimentovať s kódom bez toho aby sme akokoľvek zasahovali do hlavného kódu teda hlavnej vetvy.
* **Zlučovanie (Merging)** – Môžeme kombinovať zmeny v jednej vetve do inej vetvy. Toto robíme vtedy keď máme vetvy dostatočne otestované a pripravené na to aby boli jedna z druhou zlúčene.

## GitHub

GitHub a Git sú dve odlišné veci. GitHub je poskytovateľ internetové hostingu, je určený na vývoj softvéru a správu verzií. Ponúka distribuovanú správu zdrojového kódu prostredníctvom systému git.

GitHub je najobľúbenejší hlavne pre projekty, ktoré sú open-source. To sú projekty na ktorých môžu pracovať všetci, ktorý majú na to znalosti. GitHub je taktiež taká sociálna sieť pre programátorov. Môžeme tam ukladať naše projekty, ktoré môžeme následne ukazovať aj v prípade keď sa uchádzame o prácu. Ale hlavne je to sieť kde sa nemusíme báť ukázať náš kód pretože si tam programátori navzájom dokážu ľahko pomáhať bez toho aby priamo zasahovali do nejakého kódu.

# Ciele práce

Náš hlavný cieľ je vytvoriť webovú stránku, ktorá sa bude zameriavať na kyberbezpečnosť a, ktorá vybaví čitateľov znalosťami, na ktoré sa budú môcť spoľahnúť pri práci s internetom alebo inými technológiami. Snahou je aj urobiť curriculum, ktoré bude prehľadné. S tím súvisí aj dizajn webovej stránky. Dizajn bude mať schopnosť nadchnúť a udržať čitateľa na webovej stránke. Implementáciou obsahu o kyberbezpečnosti chceme zdieľať aj rôzne interaktívne úlohy tretích strán, ktoré majú schopnosť nie len prehĺbiť čitateľove znalosti ale aj ich otestovať.

# Materiál a metodika

V tejto kapitole rozoberieme ako sme pracovali na tvorbe webovej stránky. Ako sme ju dizajnovali, ako sme ju naprogramovali a čo sme použili pre dosiahnutie našich cieľov.

## Inštalácia a konfigurácia Gitu a Githubu

Aby sme vykročili správnym krokom musíme stiahnuť Git z oficiálnej webovej stránky. Následne ho nainštalujeme. Po inštalácií sa nám otvorí špeciálne CLI (command line interface) dedikované práve len pre Git. Toto okno môžeme zavrieť. Nájdeme si alebo vytvoríme adresár, v ktorom budeme pracovať. Klikneme pravým a otvoríme Git Bash. Pre to aby sme spustili projekt v danom adresári musíme použiť príkaz **git init**. Tento príkaz slúži na založenie nového projektu v adresári. Pri tomto príkazy sa vytvorí adresár .git, do ktorého Git ukladá všetky informácie o repozitári. Teraz potrebujeme nastaviť email a meno. Je to potrebné v prípade, že pracujeme v tíme, aby ostatný vedeli kto commitol (nahral) zmeny. Meno a email nastavíme pomocou dvoch príkazov.

**git config --global user.name ‘Naše meno’**

**git config --global user.email ‘Náš email’**

Po nastavení emailu a mena sa musíme zaregistrovať/prihlásiť na GitHub. Po tom ako sa dostaneme do GitHubu klikneme v pravo hore na plusko a následne na Nový repozitár. Na tejto stránke nastavíme repozitár. Pri zadávaní mena je dobré zadať meno adresáta, ktorý sme vytvorili pre projekt. V prípade že chceme môžeme pridať nejaký komentár alebo zmeniť repozitár na súkromný repozitár. Ak chceme vytvoríme README file. Po tom, ako všetko nastavíme, ak chceme klikneme na vytvoriť repozitár. Aby sme mohli posielať súbory do repozitáru potrebujeme personálny vstupný token. Tento token vytvoríme v nastaveniach profilu. Do nastavení sa dostaneme tak, že klikneme v pravo hore na obrázok profilu následne na nastavenia. Po tom, ako sa dostaneme do nastavení vyhľadáme na ľavej strane nastavenia developera. A znova na ľavej strane nájdeme personálny vstupný token ten rozklikneme a klikneme na tokeny (klasické). Na tejto podstránke klikneme na tlačidlo generovať nový token a klikneme na klasický. Token môžeme pomenovať a taktiež mu nastaviť dobu trvania. Následne musíme nastaviť čo môžeme robiť v prípade, že tento token používame. Keďže tento token budeme vlastniť len mi tak mu dáme všetky práva. Klikneme na generovať token čo nám už vygeneruje token, ktorý použijeme. Je dôležite tento token niekde uložiť pretože ho budeme na tejto podstránke vidieť len raz. Teraz sa pomocou Git bashu a tokenu, ktorý sme si vytvorili môžeme dostať cez Git bash do repozitára, ktorý sme vytvorili. Na toto použijeme príkaz.

**git remote add origin main https://TOKEN@github.com/GitHub meno/meno repozitáru)**

Po tomto všetkom máme Git a Github vytvorený a nastavení. Teraz pri každej zmene budeme nahrávať zmeny na GitHub. Pri nahrávaní môžu byť v troch stavoch a to: modified, staged, committed. Na nahrávanie budeme potrebovať ďalšie príkazy.

* git status – Zobrazí všetky zmenené súbory, zelenou farbou sa ukážu tie, ktoré sú už v stagingu a červenou budú tie, ktoré nie s v stagingu.
* git add (názov súboru) – Keď chceme odovzdať súbory, ktoré sme upravili, pridáme ich do stagingu pomocou tohto príkazu
* git commit – Keď máme súbory v stagingu tak použijeme tento príkaz aby sme súbory uložili a napíšeme do popisu committu krátku správu o tom čo sme robili a kde sme skončili.
* git push -u origin master – Tento príkaz sa použije na odoslanie súborov na vzdialený repozitár. Použijeme ho pri prvom odoslaní súborov.
* git push – Tento príkaz ma takú istú funkciu ako predošlý príkaz. Ale používame ho až keď odošleme prvý krát súbory pomocou predošlého príkazu.
* git pull – Vďaka tomuto príkazu môžeme stiahnuť súbory zo vzdialeného repozitáru do lokálneho pracovného adresára.

## Dizajnovanie vo Figme

Pred tým ako sme začali programovať samotnú stránku sme si dizajn webovej stránky a jej podstránok urobili vo Figme. Vďaka tomu, že základne funkcie Figmy sú nenáročne na realizáciu to nám umožnilo naše nápady v rýchlosti pretvoriť na prototyp webovej stránky, ktorý sme si upravili do finálneho dizajnu.

### Dizajn domovskej stránky (Index)

Pri dizajnovani domovskej alebo teda hlavnej stránky sme si museli určiť tematiku dizajnu, ktorú sme potom museli dodržať pri každej podstránke. To znamená, aké farby použijeme, písmo, obrázky a vcelku či to bude futuristický, retro, alebo moderný dizajn. Pretože robíme stránku o kyberbezpečnosti a samotne slovo má v sebe v anglickom výraze “cyber“ tak sme si zvolili futuristickú cyberpunk tematiku. Použili sme 3 farby a odtiene týchto farieb a to sú modrá, rúžová a fialová. Na pozadie sme použili hlavne tmavé odtiene kvôli tomu sme ako hlavnú farbu písma dali bielu aby vynikol text. Keď sme mali určené tieto hlavné faktory od, ktorých sa bude odrážať náš celkový dizajn pustili sme sa na dizajnovanie rôznych sekcií domovskej stránky.

Prvá sekcia bude obsahovať navigáciu a krátky text na upútanie. Navigácia bude fungovať ako skratka k témam, ktoré budú na hlavnej stránke. Dizajn navigačného panelu bude priesvitný vytvoríme efekt skla. Panel naprogramujeme tak aby bol aj po posunutí na obrazovke to znamená, že aj keď budeme na konci domovskej stránky tak budeme mať panel stále k dispozícii.

Druhá sekcia domovskej stránky bude krátka a výstižná. Táto sekcia sa volá (“why CyberSec“ alebo “prečo CyberSec“). V tejto sekcií sme dali 3 dôvody prečo si študenti majú vybrať práve nás.

Tretia sekcia obsahuje slovné hodnotenia našich študentov. Nazvali sme ju (“What our students say“ alebo “Čo hovoria naši študenti“ ). Obsahuje avatara alebo personálnu fotku študenta s jeho recenziou. Pomocou šipiek sa dá prehadzovať medzi študentami. Pre túto sekciu sme si niektoré obrázky vytvorili práve pomocou Figmi.

Ďalšia sekcia a teda štvrtá hovorí o tom akým spôsobom funguje naša learning platforma. Názov tejto sekcie je (“How we teach you” alebo “Ako vás učíme“). Obsahuje krátky text, ktorý vyzdvihuje naše curriculum a motivuje. Následne sme napísali o 3 hlavných okruhov o tom ako fungujeme a o čo nám ide.

Predposledná sekcia piata je na to aby ešte viac upútala užívateľa a prinútila ho sa registrovať na našu webovú stránku. Obsahuje krátky text s obrázkom a tlačidlom, ktoré presunie užívateľa na registráciu.

Posledná šiesta sekcia alebo aj takzvaná päta stránky slúži na dôležité linky a na propagáciu. V päte sa nachádzajú linky na sociálne média ako je Instagram alebo discord. Taktiež obsahuje linky na dôležité podstránky podmienky a súkromie. Súčasťou je aj copyright nášho projektu.

### Dizajn podstránok prihlasovania a registrácie

Dizajnovanie podstránok prihlasovania a registrácie bolo o niečo ľahšie. Keďže sme už mali hlavné faktory ako sú farby či tematika. Tieto dve podstránky sa od seba moc neodlišujú. Dizajn sme spravilo čisto len na jednu sekciu to znamená, že sa nedá ísť nižšie. Pri stlačení tlačidla na prihlásenie na domovskej stránke nás to presunie na podstránku.

Formulár na prihlasovanie sa nachádza v strede stránky. Do pozadia sme dali gradient s farbami, ktoré sme si určili. Za formulárom je urobené plátno, ktoré spĺňa funkciu skleneného efektu. Sklenený efekt sme použili aj na pozadie riadkov kde sa zapisujú prihlasovacie údaje. Na vrchu sa nachádza logo CyberSec projektu. Na pravej hornej strane plátna je krížik, ktorý nás má presunúť na domovskú stránku. Podstránka na prihlasovanie je skoro totožná ako podstránka na registráciu rozdiel je len v počte riadkov na zapisovanie údajov.

### Posledné úpravy a dokončenie dizajnu

Po tom ako sme mali na všetky podstránky urobené prototypy tak ostávalo už len doladiť maličkosti a trochu zmeniť kompozíciu elementov v jednotlivých sekciách. Napríklad v prototypovej verzií bola na domovskej stránke navigácia na ľavom boku. No po pár problémoch, ktoré sme si uvedomili, že môžu nastať v prípade že by sme nechali navigáciu na ľavej strane. Sme sa rozhodli urobiť navigáciu v strede stránky a to tak, že sme ju urobili ako ostrovček, ktorý sa snami neustále posúva. Potom sme museli trochu prerobiť sekciu “čo hovoria naši študenti“. Prerábali sme ju kvôli logike ako malo fungovať premieňanie komentárov a avatarov určitých študentov. Dizajn bol zbytočne komplikovaný tak sme ho trochu upravili aby bola sekcia viac atraktívnejšia a ľahšie čitateľná. Maličkosti ktoré sme dolaďovali boli napríklad rozmedzia medzi elementami alebo veľkosti elementov či písmen. Keď sme mali všetko nadizajnované a mali sme čistý prototyp hotový tak sme sa pustili do programovania webovej stránky. Je dôležité spomenúť, že aj keď sme mali všetko nadizajnované tak sme robili zmeny aj počas toho ako sme programovali .

## Programovanie a štýl ovanie webovej stránky

Prototyp dizajnu máme hotový teraz musíme vytvoriť súbory webovej stránky. HTML súbor sa bude volať index.html sem budeme písať len HTML. Vždy sa to musí nazývať index. Následne si vytvoríme druhý súbor ktorý nazveme index.css sem budeme písať len CSS. Od tohto momentu začneme aktívne používať aj Git a GitHub. Prvé čo spravíme keď otvoríme HTML súbor tak napíšeme jeden výkričník to nám da základnú štruktúru HTML súboru. Do hlavičky pridáme príkaz na prepojenie súboru CSS s našim html súborom a to príkaz nižšie

**<link rel="stylesheet" href="./index.css">**

Tento príkaz použijeme vždy keď budeme chcieť prepojiť HTML s nejakým externym súborom. V prípade, že budeme chcieť prepojiť JavaScriptovy súbor tak musíme tento príkaz napísať úplne dole do tela HTML. Je to preto lebo JavaScript načítava všetko čo sa nachádza na príkazom pre prepojenie. Je možne napísať ho aj do hlavičky a použiť špeciálny príkaz. HTML štruktúra je dôležitá preto nebudeme zbrklo písať ale budeme programovať pekne po sekciách.

Po tom ako budeme mať nastavené HTML otvoríme CSS súbor. Napíšeme príkaz, ktorý bude vypadať takto:

**\*{**

**margin: 0;**

**font-family: 'Krona One', sans-serif;**

**}**

Hviezdičku sme použili aby sme tieto zmeny použili na každý tag, ktorý použijeme. Najdôležitejšie v tomto kóde je margin, ktorý sme nastavili na nulu. Margin používame na to aby sme tagom pridali okraje. Dali sme ho na nulu aby sme zrušili už dané margin pri niektorých tagoch a mohli si ich sami meniť bez toho aby sme sa divili, že to má okraja aj keď sme margin nepoužili. Druhý príkaz, ktorý sme použili je **font-family** tento príkaz nie je moc dôležitý ale v prípade že chceme aby každý tag do, ktorého budeme písať mal určite písmo tak použijeme tento príkaz. Použitie tohto príkazu nám trochu skráti celkovú veľkosť kódu kvôli tomu že nemusíme definovať pri každom tagu font písma. Vždy keď budeme chcieť použiť niečo na celú stránku tak to dáme práve do tohto elementu s hviezdičkou. Je dôležité dodať ak v prípade, že používame font ktorý CSS neobsahuje musíme ho pridať pomocou

### Vzorové programovanie v HTML

Vieme že máme domovskú stránku rozdelenú na 6 sekcií tak vieme že budeme potrebovať 5 divov, ktoré si očíslujeme podľa sekcie. Div používame na zoskupovanie logických súvisiacich blokov alebo elementov. Kód bude vyzerať nejak takto

**<div class="section-1">**

***ďalší kód tu***

**</div>**

Na pozadie prvej sekcie sme použili obrázok. Tento obrázok môžeme dať na pozadie dvoma spôsobmi a to že to dáme ako značku to znamená, že použijeme <img> tag alebo druhý spôsob kedy to dáme cez CSS. Ľahšia varianta je druhá a aj ta, ktorú sme použili. Na to aby sme tomuto tagu dali pozadie sme použili v CSS tento kód ktorý obsahuje aj ďalšie potrebne charakteristiky:

**.section-1{**

**background-image: url(odkaz\_na\_obrazok);**

**background-size: cover;**

**background-position: center center;**

**display: flex;**

**justify-content: center;**

**}**

V prvom riadku môžeme vidieť práve príkaz ktorý použijeme na pridanie obrázku do pozadia. Môžeme vidieť, že máme tam aj iné príkazy ktoré majú v sebe slovo background. Tieto príkazy sú taktiež dôležité. Príkaz **background-size: cover;** slúži na to aby sme obraz, ktorý sme použili na pozadie zaberal celé miesto elementu v našom prípade element **section-1.** Druhý príkaz, ktorý je súčasťou pozadia je **background-position: center center;** tento príkaz slúži na to aby sme obrázok v elemente dali na nami určenú pozíciu. Tento príkaz obsahuje dvakrát hodnotu center. Je to z toho dôvodu, že tento príkaz bere dve hodnoty, jedná hodnota sa aplikuje horizontálne a druhá vertikálne. Znamená to, že ak chcem obrázok presne v strede musíme práve hodnotu center použiť dvakrát, pretože v prípade, že by sme to použili jeden krát tak by sa hodnota aplikovala len na prvú hodnotu a to horizontálnu.

Pod príkazmi na pozadie môžeme vidieť ďalšie dva príkazy. Príkaz **display: flex;** používame na to aby sme kontajner premenili na flex kontajner. Hodnota **flex** umožňuje sa položkám daného elementu zarovnávať sa flexibilne a usporiadať sa flexibilne. Je to jeden zo spôsobov ako urobiť webovú stránku viacej responzívnu. Tento príkaz v práci používame často. **Justify-content: center;** príkaz môžeme používať len keď používame príkaz **display: flex;** v elemente.Slúži na to aby sme vnútorný obsah elementu to znamená text, obrázky alebo iné položky čo sa nachádzajú v kontajneri, vycentrovali horizontálne.

Dole môžeme vidieť celý HTML kód ktorý sme použili na prvú sekciu a zároveň sme použili rovnaké tagy, ktoré boli inak pomenované a mali inú kompozíciu. Popíšeme si na čo slúžili elementy, ktoré sme nespomenuli a rozoberieme aj CSS vlastnosti, ktoré sme použili.

**<div class="section-1">**

**<div class="side-menu" data-visible="true">**

**<img src="./Images/Logo SVG.svg" alt="Logo" class="logo" id="nav-logo" >**

**<div class="main-nav">**

**<nav>**

**<a href=""></a>**

**<a href=""></a>**

**<a href=""></a>**

**<a href=""></a>**

**</nav>**

**</div>**

**<a href="" class="singin-button">Sign in</a>**

**</div>**

**<div class="mobile-menu" data-visible="false">**

**<img src="./Images/Logo SVG.svg" alt="Logo" class="logo" id="nav-logo">**

**<a href=""></a>**

**<a href=""></a>**

**<a href=""></a>**

**<a href=""></a>**

**</div>**

**<div class="main-content1">**

**<span class="motto">Cybersecurity: Your Gateway to Digital Knowledge.</span>**

**<img src="./Images/Earth.svg" alt="ilustration" class="illus-1">**

**</div>**

**</div>**

Môžeme vidieť viacero elementov ktoré sme ešte nevideli a to je napríklad **img, a, nav** a **span** ale aj rôzne hodnoty v týchto elementoch ako napríklad **class, alt, id** alebo **href**.

V rýchlosti si povieme o hodnotách elementov a na čo slúžia. Prvý element a to je jeden z najpoužívanejších a to je hodnota **class**. Táto hodnota slúži na pomenovanie elementov používame ju hlavne v vtedy keď niektoré elementy používame viac krát ale potrebujeme aby každý element mál iné vlastnosti. Nemusíme používať túto hodnotu ak používame element vždy len na jeden účel. Ďalšia hodnota, ktorá je podobná hodnote **class** je hodnota **id.** Hodnotu id používame len na jeden element ktorý chceme špeciálne upraviť. Uľahčuje to aj prácu s javascriptom aby sme sa vyhli nechceným zmenám na webovej stránke. Hodnota **alt** sa používa na presne pomenovanie elementov má to slúžiť ako navigácia pre slepých. Do hodnotí **href** zapisujeme linky na ďalšie stránky alebo podstránky.

Element **img** používame na importovanie obrázka na webovú stránku z adresára. Cestu do adresára zapisujeme do hodnoty **src**. Tag ktorý používame s **href** hodnotu je **a**tento element definuje hyperlink. Môžeme linkovať na stránky, emailové adresy alebo iné. Tag **nav** používame na definovanie navigačných linkov. Je dôležité povedať, že nie všetky linky by mali byť v **nav** tagu. Je zaužívane používať tento tag len v navigačnom panely. Potom tu máme element **span**. Používame ho na text tak, ako **div** nemá žiadne vlastnosti. Svojim spôsobom nám vie nahradiť element na paragraph. Ďalší dôležitý element, ktorý nie je vo vyššie uvedenom HTML kóde je tag **h1** až **h6.** Je to nápisový element, **h1** definuje najdôležitejší nadpis a **h6** je najmenej dôležitý.

### Vzorové programovanie v CSS

V tejto podkapitole si podrobnejšie rozoberieme dôležité CSS atribúty, ktoré sme najčastejšie používali.

O niektorých atribútoch sme hovorili už v predošlej kapitole tie už rozoberať nebudeme. Ale rozoberieme ešte ďalšie dôležité atribúty, ktoré používame spoločne s flex. Jedným z dôležitejších atribútov pri používaní flexu je **align-items: center;** tento atribut robí to iste čo justify ale tento sa zameriava na vertikálne umiestnenie. Je dôležite povedať že sú rôzne hodnoty nie len **center** sú napríklad aj **flex-start**, ktorý slúži aby sme obsah dali na začiatok kontajneru alebo **flex-end**, ktorý nám to dá na koniec kontajneru. Ďalším dôležitým atribútom je **gap.** Tak ako už aj meno tohto atribútu napovedá tento atribút využívame na vytvorenie medzery medzi riadkami a stĺpcami vo flexi. G**ap** je skratka pre atribúty **row-gap** a **column-gap.** Tento atribút používame pre design aby bol viac responzívnješí a lepšie čitateľný.

Povieme si aj o pár atribútoch, ktoré nie sú nijako spojené alebo sa dajú využívať aj bez flexboxu. Sú to atribútu, ktoré robia jednoduché veci ako napríklad **color**. Využívame ho na zmenu farby písma. Základne atribúty pre písmu sú aj **font-size** alebo **font-family.** Prvý atribút ktorý sme spomenuli používame na zmenu veľkosti písma, druhý atribút využívame na zmenu štýlu písma.

Ďalej používame aj atribúty ako sú **width** alebo **height** tieto atribúty môžeme napísať aj ako **max-height** či **min-width.** Už z mena atribútov vieme povedať, že nám budú meniť výšku alebo šírku kontajneru. V prípade že používame s atribútmi **max** tak nastavujeme maximálnu výšku a to iste pratí pre **min** kedy nastavujeme minimálnu výšku alebo šírku.

Je pár atribútov, ktoré sme využili len v špeciálnych prípadoch. Atribút **text-align** sme používali v prípade, že sme chceli text umiestniť na nami určené miesto. Môže nadobudnúť niekoľko hodnôt. Najpoužívanejšia hodnota je **text-align: center;** ale aj napríklad **left** či **right** tieto hodnoty nám presúvajú text na ľavú alebo pravú stranu. **Object-fit** je ďalší dôležitý atribút ktorý sme použili v prípade keď sa nám obrázok nevmestí do kontajneru alebo je nejaké iné chyby. Má rôzne hodnoty napríklad **cover,** ktorý vyplní celý kontajner a v prípade, že sa nevmestí do kontajneru tak ho odreže. Potom tu máme **contain** ten má za úlohu vyplniť obrázok celý s tým, že zachová pomer strán.

Posledná dôležitá vec, ktorá je vhodná spomenúť je akým spôsobom sa robí responzivita. Responzivitu robíme pomocou takzvaných media queries tie nám dovoľujú meniť kod závislosti od zariadenia kód vyzerá takto:

**@media only screen and (max-width: 1020px){**

**nav{**

**display: none;**

**}**

Zadáme nami požadovanú šírku a už len meníme kód css.

# Závery práce a zhrnutie

Neustále informovať a vzdelávať užívateľov bol náš cieľ. Na stránke, ktorú sme vytvorili budeme neustále pridávať nové informácie, ktoré budú aktuálne v oblasti kyberbezpečnosti. Curriculum a informácie, ktoré sme vytvorili a zozbierali pridávajú hodnotu nie len ľuďom, ktorý sa zaujímajú o kyberbezpečnosť ale aj ľuďom, ktorý využívajú internet a informačné technológie len pre zábavu. Cieľom bolo vytvoriť webovú stránku, ktorá bude priateľská a náučná pre každého používateľa. Vytvorili sme webovú stránku, ktorú by sme chceli do budúcnosti určite vylepšiť a spustiť ako jednu veľkú platformu pre ľudí, ktorý sa radi vzdelávajú.

# Zoznam použitej literatúry

[1] WIKIPEDIA 2024: *Figma.* [cit. 2024-01-05] Dostupné na internete: https://en.wikipedia.org/wiki/Figma

[2] WEBGLOBE 2022: *Čo je to HTML a jeho základy. Všetko, čo na začiatok potrebujete vedieť.* (Blogový článok). [cit. 2024-01-05], Dostupné na internete: https://www.webglobe.sk/blog/html-zaklady

[3] AYRIS 2014: *Ako vytvoriť animáciu v CSS3: Prvé kroky.* (Blogový článok). [cit. 2024-01-07], Dostupné na internete: https://www.ayris.sk/ako-vytvorit-animaciu-v-css3-prve-kroky/

[4] FINE: *JavaScript* (Blogový článok). [cit. 2024-01-07], Dostupné na internete: https://www.fine.sk/pojmy-a-skratky/javascript/

[5] MICROSOFT 2023: *Použitie rozšírenia Visual Studio Code.* [cit. 2024-01-07], Dostupné na internete: https://learn.microsoft.com/sk-sk/power-apps/maker/portals/vs-code-extension

[6] UŽIVATEL: *Rozšírení VS Code.* [cit. 2024-01-10], Dostupné na internete: https://uzivatel.cz/vs-code-rozsireni/

[7] CODERAMA: *Základne pojmy spojené s GIT.* [cit. 2024-01-11], Dostupné na internete: https://coderama.com/slovnik/git

[8] KRISTEĽ Marián 2022: *14 Git príkazov, ktoré musí poznať každý vývojár.* [cit. 2024-01-11], Dostupné na internete: https://skillmea.sk/blog/14-git-prikazov-ktore-musi-poznat-kazdy-vyvojar

# Prílohy

Projekt.zip