ПРИЛОЖЕНИЕ 1

МИНИСТЕРСТВО НА ОБРАЗОВАНИЕТО И НАУКАТА

ПРОФЕСИОНАЛНА ГИМНАЗИЯ ПО БИТОВА ТЕХНИКА

гр. Пловдив, ул.„Иван Перпелиев“ №2; тел.:032/628 524

www.pgbt-plovdiv-bg.com; e-mail: pgbt2005@abv.bg

ПРОФЕСИЯ: „ТЕХНИК НА КП СИСТЕМИ“

СПЕЦИАЛНОСТ: „Компютърна Техника и Технологий“

ДИПЛОМЕН ПРОЕКТ

за придобиване на трета степен професионална квалификация

ТЕМА:

Разработване и оптимизиране на WEB сайт – фотоалбум на завършващите ученици от 12 В клас, специалност КТТ в ПГБТ гр. Пловдив 2022 / 2023 г.

Дипломант: Павлин Попов

Ръководител-консултант: *инж. Димитрина Цветкова*

Клас $12^{В}$

E-mail: pacopichmena123@gmail.com

1. Въведения

Като ученик прекарах последните пет години в създаване на спомени и развитие на образованието ми. Въпреки че имаше и добри, и лоши моменти, реших да ги уловя и съхраня на уникално място - уебсайт. Избрах да създам уебсайт поради многото му предимства пред други алтернативи. Като универсално достъпна платформа чрез интернет, уебсайтът може да бъде достъпен от всяко място и по всяко време от компютър или мобилно устройство.

Уебсайтът предоставя много начини, с които може да се покажат най-ценните ни спомени от училище. Включвайки мултимедийни елементи като изображения, видеоклипове и аудио, уебсайтът не само съживява спомените, но и пренася към дадения момент. Функционалността му позволява да бъде модифициран и персонализиран до последния детайл, за да съживи картината независимо дали е чрез използване на графики или анимации.

Уебсайтът се състои от 5 основни страници, всяка със свой собствен стил. Всяка от страниците има различна функционалност. Началната страница се фокусира върху класните ръководители. Те ще имат последната дума в края на завършването ни. Учениците имаха за задача всеки от тях да обобщи в едно изречение отнасящо се за годините ни в ПГБТ. Албумът съдържа снимки от 2018 до 2023. Делнична програма, която пресмята колко време остава до края на текущия час или до края на междучасието. Показва кой учебен час е сега и следващия час или междучасие. Последната страница се отнася за мен. В нея има кратко описание на проекта и мотивацията ми.

Уебсайтът ще продължи да се поддържа до следващата учебна година. След това ще се трансформира в pdf файл и на всеки ще му бъде предоставено копие.

С Ъ Д Ъ Р Ж А Н И Е

8. 12vkttpgbt.ml
9. Пиратство
10. Защита от атаки

2. Интернет

Интернет е голяма мрежа, която свързва компютри и устройства по целия Свят, позволявайки комуникация помежду си. Това става възможно чрез използването на протоколи, които са набор от правила, управляващи как устройствата комуникират едно с друго. Интернет ни дава достъп до огромно количество информация, което прави намирането на това, от което се нуждаем, по-бързо и по-лесно. Интернет улесни комуникацията ни с хора от всички краища на земното кълбо и ще продължава да разширява възможностите си.

3. Какво е Уебсайт?

Уебсайтът е съвкупност от уеб страници, достъпни чрез интернет. За да се създаде уебсайт, уеб разработчикът пише код, обикновено в HTML, CSS и JavaScript, който след това се поддържа в уеб сървър.

Уебсайтовете са вече необходим ресурс на обществото и предлагат множество приложения, които са от полза на всеки. Една важна употреба на уебсайтовете, която ще разгледаме в тази дипломна работа, е способността да съхраняват информация и да е показват по интерактивен и визуално красив начин.

3.1. Начин на достъп

Когато потребител иска да получи достъп до уебсайт, той въвежда URL адреса на уебсайта (Uniform Resource Locator) в своя уеб браузър. Браузърът изпраща заявка до сървъра на уебсайта за уеб страницата, която включва HTML код, изображения и друго съдържание, което съставлява страницата.

URL адресът на уебсайта всъщност е четим от човека начин за представяне на IP адрес, който е уникален идентификатор за устройство в интернет. Всяко устройство в интернет има IP адрес, който се използва за маршрутизиране на пакети с данни между устройствата. Въпреки това, IP адресите са трудни за запомняне от хората, така че DNS (система за имена на домейни) е създадена, за да съпостави четливи имена на домейни към IP адреси.

DNS е разпределена система, която се състои от хиляди сървъри по целия свят. Когато потребител въведе URL адреса на уебсайт в браузъра си, браузърът изпраща заявка до DNS резолвер, който обикновено се предоставя от ISP (доставчик на интернет услуги) на потребителя. След това резолверът изпраща заявка до основен DNS сървър, който връща IP адреса на домейн сървъра от първо ниво за името на домейна на уебсайта (като .com или .org).

Резолверът изпраща заявка до домейн сървъра от най-високо ниво, който връща името на домейна на уебсайта. Сървъра за имена е отговорен за запис на IP адресите на устройствата, свързани с името на домейна на уебсайта.

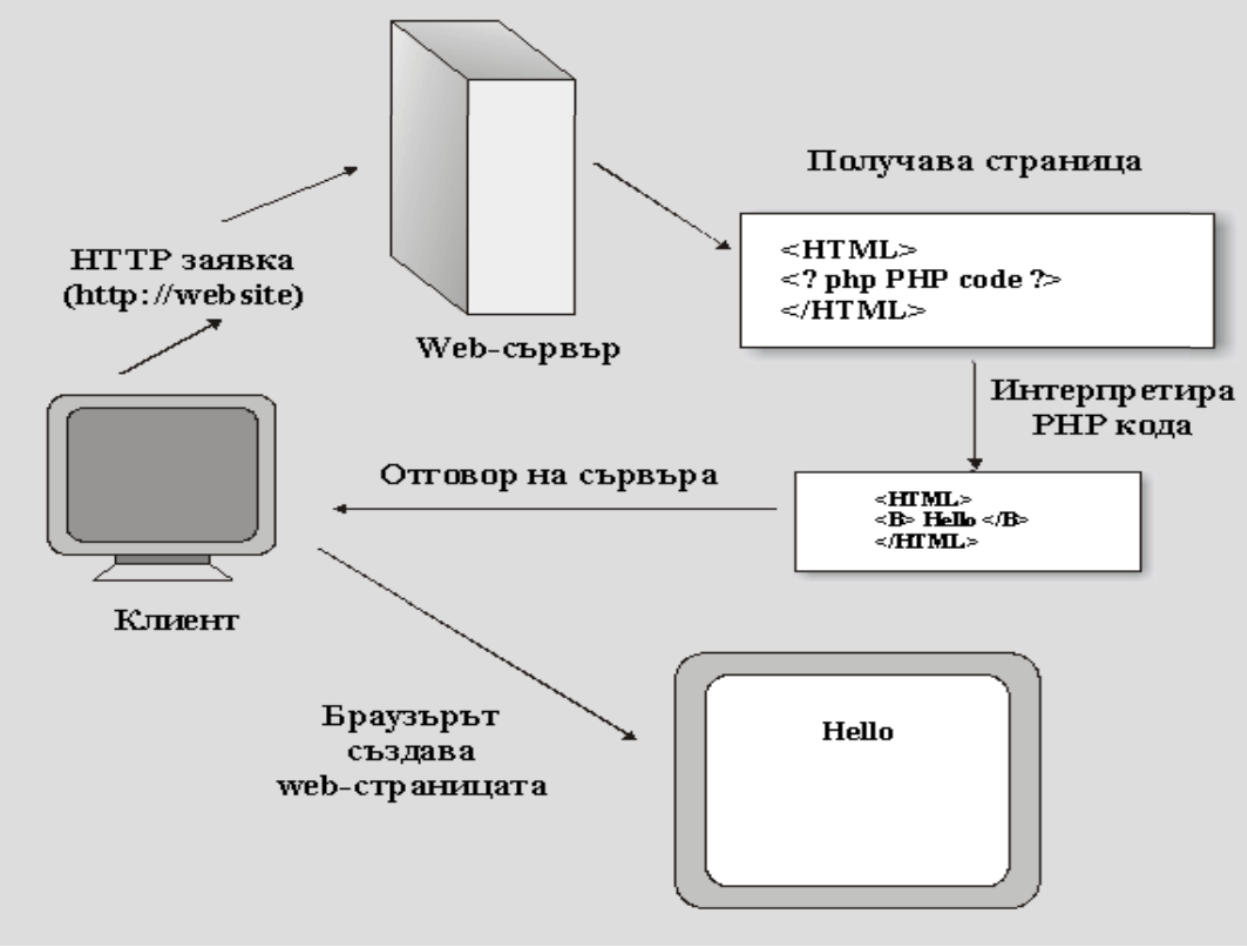
След като браузърът получи IP адреса на уебсайта, той изпраща заявка до сървъра за уеб страницата. Тази заявка включва информация за вида на използвания браузър, исканата страница и всички допълнителни параметри.

Сървърът получава заявката и започва да я обработва. Първо, сървърът може да провери дали исканата страница е в неговия кеш. Кеширането е процес на съхраняване на често достъпни данни в локално пространство за съхранение за по-бърз достъп. Ако исканата страница е в кеша, сървърът може да я върне веднага, без да се налага достъп до базата данни или други ресурси.

Ако страницата не е в кеша, сървърът извлича необходимите данни и генерира HTTP (Hypertext Transfer Protocol) отговор. Отговорът включва HTML код, изображения и друго съдържание, което съставлява уеб страницата, както и всякаква допълнителна информация, като например бисквитки.

Сървърът изпраща HTTP отговор обратно към браузъра на потребителя, който го получава и започва изобразяването на страницата. Браузърът използва HTML кода, за да покаже текста и изображенията на екрана.

След като страницата приключи изобразяването, потребителят може да взаимодейства със страницата, като кликва върху връзки, попълва формуляри или да извършва други действия. Когато потребителят щракне върху връзка, неговият браузър изпраща нова заявка до сървъра, който отговаря с подходящото съдържание за новата страница. Този процес продължава, докато потребителят напусне уебсайта или затвори своя браузър. По време на целия този процес сървърът на уебсайта непрекъснато комуникира с браузъра на потребителя, като обменя данни и изобразява съдържанието на страницата в реално време.



Едно допълнение към процеса са бисквитките. Те са малки файлове, които се съхраняват в браузъра и се използват за съхранение на информация за предпочитанията на потребителя, информация за влизане и други данни, които се

използват за подобряване на работата на уебсайта. Бисквитките също могат да се използват за проследяване на активността на потребителя на уебсайта, въпреки че тази практика е предмет на разпоредби за поверителност.

3.2. Видове

Има голямо разнообразие от функции, които уебсайтовете могат да обслужват, като различни видове отговарят на различни нужди. Уебсайтовете могат да варират от лични блогове до онлайн магазини и от корпоративни сайтове до образователни ресурси. Съответно те могат да бъдат класифицирани в различни типове въз основа на тяхната предвидена функция и цел.

- Информационни: Много уебсайтове служат като източник на информация за определена тема, като новинарски уебсайтове или образователни уебсайтове. Тези уебсайтове обикновено включват статии, видеоклипове и други ресурси, които имат за цел да информират или образоват потребителя.
- Електронна търговия: Онлайн магазините са често срещан тип уебсайт, който позволява на потребителите да купуват продукти или услуги по интернет. Уебсайтовете за електронна търговия обикновено включват функция за пазарска количка, продуктови списъци и функционалност за плащане – eMag.bg.
- Социални: Социалните мрежи като Facebook, Twitter и Instagram позволяват на потребителите да се свърже с приятели и семейство чрез тях. Могат да споделят снимки и видеоклипове и да създават нови запознанства онлайн.
- Развлечения: Уебсайтове като YouTube, Netflix и Spotify предоставят развлекателно съдържание като видеоклипове, филми, телевизионни предавания и музика.

- Блогове: Блогове като WordPress и Medium позволяват на потребителя да създават и публикуват собствено съдържание в интернет, обикновено под формата на писмени статии.
- Портфолио: Много хора използват уебсайтовете като начин да покажат своята работа и умения пред потенциални работодатели или клиенти. Тези уебсайтове обикновено включват портфолио от работата на лицето, заедно с автобиография или друга информация за неговия опит.
- Правителствени: Държавните уебсайтове предоставят информация за държавни услуги, закони и разпоредби. Тези уебсайтове обикновено се поддържат от държавни агенции на местно, щатско или национално ниво.
- Дарения: Организациите с нестопанска цел често използват уебсайтове като начин за популяризиране на тяхната кауза и събиране на дарения от поддръжници. Тези уебсайтове обикновено включват информация за фонда, програмите и събитията на организацията.
- Автобиографични: Хората изразят себе си онлайн, обикновено чрез блог. Тези уебсайтове да включват информация за хобитата, интересите или личния живот на автора.

Важна подробност, която не бива да се изпуска, е вида на уебсайта според интерактивността му. Статичните показват едно и също съдържание на всеки потребител и не позволяват потребителско да въвеждат информация в базата данни. Динамичните, от друга страна, могат да генерират различно съдържание въз основа на въведени от потребителя данни

3.3. Широка съвместимост

Широката съвместимост на уебсайтове в интернет е критичен аспект, който позволява достъп до уеб съдържание на различни устройства и платформи. Уебсайтовете трябва да бъдат проектирани и разработени така, че да са съвместими с различни уеб браузъри, операционни системи, размери на екрана и интернет скорости.

За да постигнат съвместимост, уеб разработчиците използват уеб стандарти и най-добри практики, които гарантират, че уеб страниците се изобразяват правилно на всички устройства. Уеб стандартите са насоки за създаване на уеб страници, които гарантират съвместимост, достъпност и използваемост. Тези стандарти се поддържат от организации като World Wide Web Consortium (W3C) и Internet Engineering Task Force (IETF).

Удобният уеб дизайн е друг подход, използван от разработчиците за осигуряване на съвместимост. Тази техника включва проектиране на уеб страници, които могат автоматично да се адаптират към различни размери на екрана и различни устройства. Адаптивният уеб дизайн преоразмерява елементите на различните устройства, за да оптимизира страницата на различните устройства.

Обема на информация, който се пренася през интернет трябва да в границата на нормалното. В повечето случай се налага да се оптимизира информацията, за да забърза зареждането на страницата. Чрез компресиране на изображения, минимизиране на HTTP заявки и използване на мрежи за доставка на съдържание (CDN). CDN са мрежи от сървъри, които кешират съдържанието на уебсайтове на различни места по света, което улеснява достъпа до уебсайта от различни части на света.

3.4. Стратегии за съдържание

Стратегията за съдържание има голямо значение за създаването на уебсайта. Включил съм няколко дизайнерски идеи и методи в процеса на проектиране му.

В началото трябваше да направя дълбоко и обширно проучване за това какво ще бъде най-подходящо да се показва на самия уебсайт, така че да бъде в контекст с предстоящите събутия. Това ми помага да определя кои теми ще бъдат най-подходящи. Търсех начини да създам комбинация от най-доброто съдържание и най-добрия дизайн съчетани в едно. Това в крайна сметка се оказа от ключово значение за темата на проекта.

3.4.1. Ключови думи

Ключовите думи също са важен елемент от стратегията. Ключовите думи са думи или фрази, които посетителя обикновено използва, за да опишат информацията, която търсят. Те играят критична роля при насочването на потребителите към подходящо съдържание. Думите се ориентират около темата, така че посетителя да се навигира по страницата и като цяло в целия уебсайт. Тези думи са кратки и се намират от началото до края на всяка страница.

След като идентифицирахме нашите целеви ключови думи, важно е да ги използвате стратегически във нашето съдържание. Това не означава да напълним съдържанието си с възможно най-много ключови думи - това е тактика, известна като препълване с ключови думи, която всъщност ще да навреди на съдържанието и оформлението. Стремим се да използваме естествено количество ключови думи в съдържанието. Това означава да ги включим в заглавията, подзаглавията и в навигационното меню по начин, който има смисъл и се чете добре. Ако пък не можем да намерим нашите целеви ключови думи,

можем да използваме свързани ключови думи и синоними. Ключовите думи са само едно парче от пъзела за стратегия за съдържание.

3.4.2. Концепции

След като вече имаме установена цел с дадени критерии, започнаме да разработваме план за съдържание. Той включва комбинация от различни стратегически практики за съдържание, като се стремим да покрием широк набор от теми.

Една важна концепция на дизайна е визуалната йерархия, която приоритизира най-важната информация на страницата и ориентира потребителите. Постигаме визуална йерархия чрез използване на типография, разнообразни цветове и достатъчно разстояние между елементите.

Друг важен метод за проектиране, който използвам, е адаптивният дизайн. Той се адаптира към размера на екрана и ориентацията на устройството, осигурявайки оптимално преоразмеряване на елементите при навигиране. Тъй като все повече и повече хора използват мобилни устройства за сърфиране в интернет, адаптивният дизайн е основен метод за проектиране, за да се гарантира, че уебсайтът изглежда добре и функционира добре на всички устройства.

Минимализмът е една от популярните тенденции в планирането на дизайна, която е включена в повечето от нашите страници. Минималистичният дизайн използва прости форми, цветове и типография, за да създаде чист и непретрупан дизайн, който е лесен за четене.

Цветовото разнообразие е друга дизайнерска концепция, която е вложена. Цветовете могат да предизвикат определени емоции и настроения и разработчиците на уебсайтове могат да използват това в своя полза, като изберат

правилните цветове за своя уебсайт. Например, синьото често се свързва с доверие и надеждност, докато червеното се свързва със страст и вълнение.

Типографията е шрифтът, размерът и разстоянието на текста. Те са важни фактори, които трябва да имате предвид. Правилната типография прави уебсайта да изглежда професионален.

Добре планирана стратегия за съдържание, която взема предвид желаните ни резултати, значително подобрява ефективността на уебсайт като предоставя ясна информация, лесна навигация и стратегически етикети.

3.5. Дизайн

След като имаме изградена стратегия за съдържание, трябва да изготвим подходящ дизайн. Дизайна е много важна част от стратегията, защото визуализира съдържанието, така че да е лесен за възприема от читателя.

Дизайнът на уебсайт е критичен компонент на всеки уебсайт. Това включва планиране, създаване и актуализиране на оформлението, съдържанието и визуалните елементи на уебсайта. Добре проектираният уебсайт е от съществено значение за създаването на завладяващо и интуитивно потребителско изживяване, което насърчава посетителите да останат на сайта, да изследват съдържанието му и да предприемат действия.

3.5.1. Ключови елементи

Ключовите елементи

3.5.1.1. Оформление

Оформлението е първото нещо, което забелязваме, когато попаднем на уебсайта. Добре структурираното оформление помага се насочим към съдържанието, което

търсим по-лесно. За да създадат успешно оформление, дизайнерите трябва да обърнат внимание на няколко аспекта, включително разположението на съдържанието, използването на бяло пространство и цялостния поток на страницата. Може да се използва решетка или Wrapper, за да сме сигурни, че дизайнът е правилно подравнен и визуално балансиран. Важно е също уебсайтът да изглежда добре на всички устройства.

Едно от ключовите съображения при проектирането на оформлението на уебсайт е разположението на съдържанието. Ние се стремим да поставим най-важното съдържание на видно място в горната част на страницата или в центъра на страницата. Това може да включва основното послание на уебсайта, призив за действие или представен продукт. Поставянето на важно съдържание на видно място помага да се привлече вниманието на потребителя към него и ги насърчава да се ангажират с него.

Друг важен аспект на оформлението на уебсайта е бялото пространство, известно още като празни полета. Оставяме достатъчно бяло пространство между елементите, за да създадем визуален баланс и контраст между различните елементи на страницата. Освен това прави съдържанието по-лесно за четене и разбиране.

Потокът и йерархията също са важни съображения при проектирането на оформление на уебсайт. Стремим се да създадем ясна йерархия и поток от съдържание, който насочва погледа на потребителя от една секция към друга. Това помага да се създаде логична прогресия на информацията и позволява на потребителите лесно да разберат структурата на сайта. Йерархията може да бъде постигната чрез използване на различни размери и стилове на шрифта, цветови контраст и визуални знаци като стрелки или икони.

Често се използва Wrapper система, за да се създаде балансирано и структурирано оформление. Wrapper системата е рамка, която разделя страницата на колони и редове, което ни позволява да поставяме съдържание по последователен и организиран начин. Това също помага да се гарантира, че дизайнът и се адаптира към различни размери на екрана.

Визуалният баланс е важен аспект от оформлението на уебсайта. Стремим се да постигнем балансирано разпределение на визуални елементи, като изображения, текст и празно пространство.

3.5.1.2. Цветова схема

Изборът на правилната цветова схема е също важно за създаването на визуално привлекателен уебсайт. Цветовата схема трябва да се основава на идентичността на марката и целевата аудитория. Цветовете могат да предадат различни емоции и настроения, така че е важно да изберем цветове, които отговарят на желания тон на уебсайта или страницата. Една последователна цветова схема също може да помогне да създадем добро усещане.

Различните цветове могат да предизвикат различни емоции и да имат различно значение. За да изберем правилната цветова схема, първо вземаме предвид стила на уебсайта. Съобразяваме цветовата схема с емоцията или нещото, което искаме да покажем. Например в нашият случай началната ни страница има сив и златист оттенък, който цели да покаже колко скъпи са за нас класните ни ръководители. Като разбираме емоционалните и психологическите ефекти на различните цветове, ние избираме цветова схема, която е в съответствие с целите и посланията на уебсайта.

Контрастът на цветовете трябва да бъде балансиран. Високият контраст може да изпъкне важни елементи и да подобри четливостта, докато ниският контраст може да създаде по-фин и хармоничен ефект.

Цветовата хармония също е от съществено значение за един сплотен дизайн. Ние създаваме хармонична цветова схема, като използваме цветове, които се допълват взаимно. Можем да използваме цветове, които са един до друг в цветното колело, като аналогични цветове, или цветове, които са един срещу друг в цветното колело, като допълващи се цветове. Както в страницата ЗаМен.html цветовете се допълват един в друг и сменят позицията си.

И накрая, разглеждаме цялостната естетика на цветовата схема. Добре проектираната цветова схема трябва да бъде визуално привлекателна и подходяща за предназначението и аудиторията на уебсайта. Можем да експериментираме с различни цветови комбинации и да използваме инструменти като цветови палитри или инструменти за избор на цветове, за да си помогнем да изберем най-добрите цветове за нашия дизайн.

3.5.1.3. Типография

Когато проектираме уебсайта, избираме шрифт, който е подходящ за нашата целева група – училището, учителите, класа. Вземаме предвид фактори като четливост, мащабност и стил. Шрифт, който е твърде труден за четене, може да не изпъкне елементите достатъчно добре и да доведе до неудобство на читателите. Те ще предпочетат да не прекарват повече време и да ги накара да напуснат уебсайта. От друга страна, шрифт, който е твърде прост или скучен, може да не успее да привлече достатъчно внимание и да не успее да придаде нашето послание.

Също така вземаме предвид йерархията на съдържанието, когато избираме шрифт. Избираме шрифт, който прави разлика между различните видове съдържание на уебсайта, като заглавия, подзаглавия и основен текст. Използвайки различни размери и стилове на шрифта, ние създаваме ясна йерархия от информация, която помага на потребителите лесно да навигират и разбират съдържанието на уебсайта.

Обръщаме внимание и на разстоянието и височината на реда при избора на типография. Доброто разстояние между редовете и знаците подобрява четливостта прави съдържанието на уебсайта по-приветливо. Важно е да вземем предвид оформлението на уебсайта, когато избираме типография.

3.5.1.4. Навигация

Навигацията се отнася до откриването на съдържанието на уебсайта, за да помогне на потребителите да се навигират и пренасят до различни страници. Добре проектираната система подобрява ефективността.

Трябва да организираме съдържанието на уебсайта в логични категории и подкатегории, за да улесним потребителите да намерят информацията, от която се нуждаят. Трябва да се уверим, че система е лесна за използване и разбиране, дори за потребители, които не са запознати с уебсайта.

Използваме горния конти тур за меню с навигация в горната част на страницата. Там са представени страниците като при натиск те ще сменят цвета си в жълто и по този начин ще се активират като ви пренесат в страницата. Те са за цел да помогнем на потребителя да разбере какво ще намери, когато кликнат върху тях. При мобилната версия се използват икони, за да съберем връзките в оптимално пространство.

3.5.1.5. Практичност

Практичността се отнася до това колко лесно е на посетителя да взаимодейства с уебсайт. За да постигнем добра практичност, вземаме предвид различни фактори при проектирането на уебсайт.

Оптимизираме скоростта за зареждане, за да избегнем дългото чакане при зареждане на уебсайта. Компресираме изображенията и оптимизираме кода, за да осигурим кратко време за зареждане.

Фокусът ни върху доброто съдържание гарантира, че то е добре написано, лесно за четене и подходящо за конкретната страница. Използваме ясен език и не изключваме думи като жаргони или фрази, които биха загрозили съдържанието.

Тестваме функционалността на уебсайта, за да гарантираме, че всички функции работят правилно и идентифицираме и коригираме всички проблеми, преди уебсайтът да бъде готов за показване. Използваме анализи, за да проследяваме поведението на уебсайта и да го коригираме при нужда. Също така събираме обратна връзка от посетителите, за да идентифицираме евентуални проблеми и да подобрим приложимостта на уебсайта.

3.5.1.6. Снимки и видеосъдържание

Снимките и видеосъдържанието са голяма част от албума. Те имат за цел да покажат всичките спомени от нашето пребиваване в

гимназията. Снимките и видеоклиповете са сортирани според датата на създаване. Всички снимки и видеоклипове са компресиране и конвертирани в различни формати, за да се избегне бавното зареждане на страницата.

Снимките са компресиране чрез помощта на езика Python и различни библиотеки. Тъй като имаме голямо количество снимки и видеоклипове, необходимостта от конвертиране и компресиране е належаща. Създадохме програмен код, който да свърши всичката тази работа. Програмите работят в настоящата директория където се намират снимките и самата програма, която използваме.

4. Създаване на уебсайта

Уебсайтът е създаден посредством комбинация от софтуерни инструменти и програми за създаване. Те включват предимно текстови редактори, системи за актуализации, системи за управление на съдържанието, скриптови езици от страна на сървъра и от страна на клиента, уеб сървъри, инструменти за тестване и отстраняване на грешки и инструменти за внедряване.

4.1. Описание

Сайтът съдържа 5 главни страници и 5 подстраници, всяка която изпълнява различна роля. На всяка една от страниците има Header, който служи за навигация към различните страници. При натискане на текста, който всъщност е бутон, с името на страницата посетителят на уебсайта ще се пренесе в съответната страница, върху която е натиснал. Header е форматиран с шрифт "Open Sans" и

"Lobster", за да може да излгежда добре и лесен за навигация. В центъра на Header е изписано името на нашият клас, от ляво е текста(бутон) с навигация към началната страница, от дясно се намират и останалите страници "Ученици", "Албум", "Програма", "За мен". При посещение на всяка от страниците ще се активира текста и ще промени цвета си в златен цвят. Ще остане активен(текста(бутона)), докато не посети друга страница.

4.2. Страници

4.2.1. Заглавна

4.2.2. Ученици

Целта на тази страница е да представи всеки ученик от класа с негова снимка и изречение или цитат, с което да опише престоя си в ПГБТ. Те имаха право да изберат своя снимка или да изкачат до бала, от който ще се сдобием с хубави снимки. Цитатът или изречението, което трябва да напишат трябва да е нещо, което те са усетили през престоя си в гимназията. Имат право да напишат всичко, което сметнат за правилно.

Снимката и цитатът всъщност представляват карта, която ученикът може да персонализира. Лицето на картата е самата снимка на ученика с името му в долния ляв ъгъл. При натискане на картата или снимката ще се създаде анимация на завъртане и гърба на картата ще се появи. Те също така имаха право персонализират картата си.

Вариантите бяха:

- Да се сложи друга снимка по избор на ученика.
- Да се сложи background снимка на гърба на картата.
- Да се сложи бутон, който да включва звук, избран от ученика.
- Текста или цитата са напълно избираеми от всеки ученик.

Каквито идеи бяха предложени, такива бяха имплементирани. Никой не беше разочарован!

4.2.2.1. Оформление

4.2.2.2. Карти

Картите както споменахме по-горе са основен елемент в тази страница. Те представляват лице с снимка на ученика и текст с името на ученика, който се намира долу в ляво. И гръб с текст или цитат и свободно избираеми елементи по избор. Допълнителните елементи са:

- Снимка
- Видеоклип
- Текст
- Звук
- Анимация
- Допълнително форматиране на картата (по избор)
- Линк към социална медия (Facebook, Instagram, Github...)

Функцията на картата се крие в преобръщането ѝ. За тази цел трябва да напишем код на JavaScript, който да преобръща картата при натискане върху нея. Програмата не е нужно да е на отделен файл, а може да е в HTML страницата. Този код използва jQuery, което е JavaScript библиотека, която опростява манипулирането на HTML документи, обработката на съобщения, анимации и взаимодействия с Ajax. jQuery използва синтаксис, който е подобен на CSS селекторите, за да намира и манипулира елементи в HTML документа. Кода на програмата

4.2.3. Албум

Както вече споменахме по-рано в дипломния проект снимките и видеосъдържанието се намират точно в нашия албум. Той се състои от 1 главна страница и 5 подстраници, които всяка от тях изпълнява различна роля, но с еднаква цел. Тук ще разгледаме как сме оформили съдържанието, конвертирали, сортирали и накрая показали снимките и видеосъдържанието.

Главната страница на албума служи като навигация за отделните години, в които сме разположили нашите снимки. На нея намираме навигационни бутони с текст годината, която искаме да посетим.

Нашите снимки и видеоклипове се нуждаят от специална обработка, за която ще използваме набор от трикове и инструменти. След обработката на снимките, те ще се покажат на страницата с помощта на библиотека наречена Nanogallery2. Тя е най-добрият вариант за нашия дипломен проект, тъй като предлага много настройки и позволява да се персонализира.

4.2.3.1. Снимки

Снимките са важна част от нашия уебсайт, тъй като представляват уловени спомени от конкретен момент. Те са общо около 500 и са събрани от всякакви източници като: Facebook, Instagram, галерии и други. Снимките са разположени в библиотека за прожектиране на снимки. Събрани са от началото на престоя ни заедно с класа до абитуриентския бал. Те са групирани по класове, пример: 8 клас – 2018 до 2019 година. Те са официални и не толкова официални.

Тъй като снимките са много и се налага да се направят няколко необходими оптимизации, които са важни за финалния продукт. Разглеждаме ги в няколко стъпки, които следвахме докато създавахме албума.

Първата стъпка е да разделим снимките по дати и събития. Много е важно тази стъпка да се случва с необходимото време и

внимание, тъй като е много лесно да объркаме времето и мястото на конкретната снимка, ако тя няма правилно зададена дата и час на създаване.

Втора стъпка е да изберем снимките, които искаме да конвертираме с цел оптимизация. От собствения си файлов формат ще се трансформират в WebP файлов формат. В сравнение с други формати на изображения, като JPEG и PNG, WebP предлага по-добро компресиране и по-малки размери на файловете, което помага за времето за зареждане на уеб страницата и намалява разхода на интернет трафик.

Трета стъпка е да компресираме WebP файловете. WebP изображенията се компресират, за да се намали размерът на файла и да се улесни зареждането им. Компресирането на този вид формат не е задължително. Но в нашия случай, тъй като имаме голямо количество снимки, които трябва също да могат да се заредят и от мобилни устройства с мобилни данни, ние трябва да намалим размера им възможно най-много. Размерът на файла ще се намали с около 20%, за да не доведе до известна загуба на качество на изображението.

Четвърта стъпка е сортирането на снимките. То се осъществява чрез програма, която ние създадохме и има за цел да преименува всички снимки от 001 до NNN според датата на създаването им.

Пета стъпка е да качим нашите снимки в HTML страницата ни. Това се осъществява с нашата среда за програмиране Visual Studio Code, като прибавяме всяка снимка на отделен ред и според номерацията ѝ в тага, който е всъщност нашата галерия.

4.2.3.1.1. Конвертор

Втората стъпка е да преместим всички снимки, които искаме да конвертираме. За да конвертираме снимките по бърз и ефективен

```python
import os
import sys
import logging
from PIL import Image

logging.basicConfig(level=logging.ERROR)

if len(sys.argv) > 1:
    dir_path = sys.argv[1]
else:
    dir_path = os.path.dirname(os.path.abspath(__file__))

try:
    os.chdir(dir_path)
except FileNotFoundError:
    logging.error('Директорията не е намерена: {}'.format(dir_path))
    sys.exit(1)

img_folder = os.getcwd()

output_folder = os.path.join(img_folder, 'webp_images')
if not os.path.exists(output_folder):
    os.makedirs(output_folder)

for filename in os.listdir(img_folder):
    if filename.endswith('.jpg') or filename.endswith('.png'):
        try:
            with Image.open(filename) as im:
                output_filename = os.path.splitext(filename)[0] + '.webp'
                output_path = os.path.join(output_folder, output_filename)
                im.save(output_path, 'webp')
        except OSError:
            logging.error('Грешка при конвертиране на файл: {}'.format(filename))

print('Всички JPEG и PNG файлове бяха конвертирани в WEBP формат и запазени в новосъзданата папка ' + output_folder)
```

начин ще се наложи да създадем програма, която да спести много от нашето време загубено в конвертиране с използването на онлайн конвертори. Програмата е на езика Python и използва няколко библиотеки за тази цел. Кода на програмата:

Обяснение:

Първо, кодът импортира необходимите библиотеки и модули - os, sys, logging и модула Python Imaging Library (PIL), по-специално класа Image (4.2.3.1. фиг. 1).

```
import os
import sys
import logging
from PIL import Image
```

4.2.3.1. фиг. 1

След това регистрационният модул е конфигуриран да извежда само съобщения за грешка (4.2.3.1. фиг. 2).

```
logging.basicConfig(level=logging.ERROR)
```

4.2.3.1. фиг. 2

След това кодът проверява дали е предоставен аргумент от командния ред при изпълнение на скрипта. Ако е предоставен аргумент, той се присвоява на променливата dir_path. В противен случай абсолютният път на директорията, съдържаща скриптовия файл, се присвоява на dir_path (4.2.3.1. фиг. 3).

Методът os.chdir() променя текущата работна директория към директорията, указана от dir_path. Ако посочената директория не

```
if len(sys.argv) > 1:
    dir_path = sys.argv[1]
else:
```

4.2.3.1. фиг.3

```
try:
    os.chdir(dir_path)
except FileNotFoundError:
    logging.error('Directory not found: {}'.format(dir_path))
    sys.exit(1)
```

4.2.3.1. фиг. 4

съществува, се записва съобщение за грешка и скриптът излиза с код на състояние 1 (4.2.3.1. фиг. 4).

След смяна на правилната директория, скриптът задава променливата img_folder на текущата работна директория (4.2.3.1, фиг. 5).

4.2.3.1. фиг. 5

След това скриптът създава нова директория с име „webp_images“ в текущата работна директория, ако тя все още не съществува (4.2.3.1. фиг. 6).

```
output_folder = os.path.join(img_folder, 'webp_images')
if not os.path.exists(output_folder):
    os.makedirs(output_folder)
```

4.2.3.1. фиг. 6

След това програмата преминава през всички файлове в директорията img_folder и проверява дали името на файла завършва с „.jpg“ или „.png“. Ако го направи, скриптът се опитва да отвори файла с изображение с помощта на метода Image.open() на PIL и го преобразува във формат WebP с помощта на метода im.save(). След

```
for filename in os.listdir(img_folder):
    if filename.endswith('.jpg') or filename.endswith('.png'):
        try:
            with Image.open(filename) as im:
                output_filename = os.path.splitext(filename)[0] + '.webp'
                output_path = os.path.join(output_folder, output_filename)
                im.save(output_path, 'webp')
        except OSError:
            logging.error('Failed converting files: {}'.format(filename))
```

4.2.3.1 фиг. 7

това полученият WebP файл се записва в предварително създадената директория „webp_images“ със същото основно файлово име като оригиналното изображение, но вместо това с разширението „.webp“ (4.2.3.1. фиг. 7).

Накрая се отпечатва съобщение, което показва, че всички JPG и PNG файлове са успешно конвертирани и поставени в директорията „webp_images“ (4.2.3.1. фиг. 8).

Програмата след като бъде задействана снимките биват конвертирани.

4.2.3.1.2. Компресия

Следваща стъпка от обработката на снимките е тяхната компресия. За да може размерът на снимките да не бъде толкова голям, те трябва да се компресират и в същото време да не губят качеството на изображението. Отново ще използваме езика за програмиране

4.2.3.1. фиг. 8

Python, за да създадем програма, която да улесни нашата работа по снимките. Процесът с тази програма е автоматичен и е необходимо програма да се намира в директорията на снимките. Важно нещо обаче, което трябва да кажем е, че програмата ще конвертира .webp файлов формат. Той е много по-лек и е по-добрата опция, вместо да конвертираме традиционните файлови снимкови файлови формати като .png или .jpg (4.2.3.1. фиг. 9).

```
print('All JPG and PNG files have been converted and placed in: ' + output_folder)
```

Кода на програмата за компресиране:

```python
import os
from PIL import Image

script_dir = os.path.dirname(os.path.abspath(__file__))

compressed_dir = os.path.join(script_dir, 'compressed')
if not os.path.exists(compressed_dir):
    os.mkdir(compressed_dir)

for filename in os.listdir(script_dir):
    if filename.endswith('.webp'):
        with Image.open(filename) as img:
            quality = 70
            output_path = os.path.join(compressed_dir, filename)
            img.save(output_path, 'webp', quality=quality)
```

4.2.3.1. фиг. 9

Обяснение:

Първо, кодът импортира необходимите модули - os и модула Python Imaging Library (PIL), по-специално класа Image (4.2.3.1. фиг. 10).

```python
import os
from PIL import Image
```

4.2.3.1. фиг. 10

Променливата script_dir е зададена на абсолютния път на директорията, съдържаща скриптовия файл (4.2.3.1. фиг. 11).

```python
script_dir = os.path.dirname(os.path.abspath(__file__))
```

4.2.3.1. фиг. 11

Променливата compressed_dir е зададена на пътя на нова директория, наречена „compressed“ вътре в директорията, съдържаща скриптовия

```python
compressed_dir = os.path.join(script_dir, 'compressed')
if not os.path.exists(compressed_dir):
```

4.2.3.1. фиг. 12

файл, и ако тази директория не съществува, тя се създава с помощта на метода os.mkdir() (4.2.3.1. фиг. 12).

След това кодът преминава през всички файлове в директорията script_dir и проверява дали името на файла завършва с „.webp“. Ако го направи, скриптът се опитва да отвори файла с изображение с помощта на метода Image.open() на PIL и го компресира с ниво на качество 70 с помощта на метода img.save(). След това полученото компресирано изображение се записва в директорията „компресирано“ със същото файлово име като оригиналното изображение (4.2.3.1. фиг. 13).

```
for filename in os.listdir(script_dir):
    if filename.endswith('.webp'):
        with Image.open(filename) as img:
            quality = 70
            output_path = os.path.join(compressed_dir, filename)
            img.save(output_path, 'webp', quality=quality)
```

4.2.3.1. фиг. 13

Накрая се отпечатва съобщение, което показва, че компресията на изображението е завършена (4.2.3.1 фиг. 14).

```
print("Image compression is complete!")
```

4.2.3.1. фиг. 14

4.2.3.1.3. Сортиране

Последната програма, която ще създадем е програма, която да намира всички .webp файлове, които да сканира и подреди по дата на създаване. Този процес е нужен да се осъществи, за да се открият и подредят заснетите моменти в хронологичен ред.

Кода на програмата за сортиране (4.2.3.1 фиг. 15):

```python
import os
import datetime

dir_path = os.path.dirname(os.path.realpath(__file__))
os.chdir(dir_path)
webp_files = [f for f in os.listdir() if f.endswith('.webp')]
webp_files.sort(key=lambda x: os.path.getctime(x))

for i, filename in enumerate(webp_files):
    new_name = f"{i+1:03d}.webp"
    os.rename(filename, new_name)
```

4.2.3.1. фиг. 15

Обяснение:

Първо, кодът импортира необходимите модули - os и datetime (4.2.3.1 фиг. 17).

```python
import os
import datetime
```

4.2.3.1. фиг. 16

Променливата dir_path е зададена на абсолютния път на директорията, съдържаща скриптовия файл, като се използва модулът

```
dir_path = os.path.dirname(os.path.realpath(__file__))
os.chdir(dir_path)
```

4.2.3.1. фиг. 17

os.path. Методът os.chdir() променя текущата работна директория в директорията, в която се намира скриптът (4.2.3.1 фиг. 17).

След това кодът създава списък с всички файлове в текущата работна директория, които завършват с „.webp“, като използва разбиране на списъка и метода os.listdir() (4.2.3.1 фиг. 18).

Списъкът webp_files се сортира въз основа на времето за създаване на всеки файл с помощта на метода os.path.getctime() като ключ за

```
webp_files = [f for f in os.listdir() if f.endswith('.webp')]
```

4.2.3.1. фиг. 18

```
webp_files.sort(key=lambda x: os.path.getctime(x))
```

4.2.3.1. фиг. 19

сортиране (4.2.3.1 фиг. 19).

След това кодът преминава през сортирания списък от .webp файлове и преименува всеки файл с ново име във формат „001.webp“, „002.webp“ и т.н., като използва форматиран низ и метода os.rename() (4.2.3.1 фиг. 20).

Функцията enumerate() се използва за преминаване през всеки файл

```
for i, filename in enumerate(webp_files):
    new_name = f"{i+1:03d}.webp"
    os.rename(filename, new_name)
```

4.2.3.1. фиг. 20

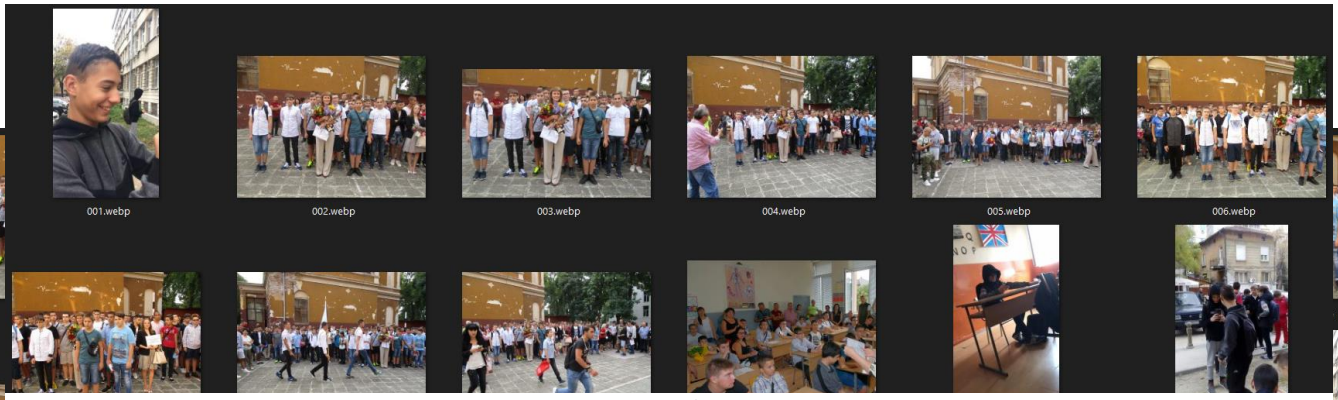
в списъка, а стойността i+1 се използва за създаване на цифровата част от новото име на файл. Спецификаторът на формат :03d се използва за допълване на цифровата част с водещи нули, така че имената на файловете да бъдат сортирани в числов ред.

4.2.3.1.4. Ръчно преглеждане за грешки и преименуване

Оказа се, че снимките, макар и от надеждни източници, са разбъркани и в голямата си част нямаше как да ги подредим по дата на създаване. Затова се наложи да прегледаме всяка снимка, след като вече сме ги компресирали, конвертирали и сортирали. Всъщност това не се оказа загуба на време, защото така или иначе не извършихме по-горните процеси ръчно. Снимките, които сметнахме, че са по-напред от останалите, ги преименувахме с номерация от 1 до N. Това в крайна

сметка ни позволи да обърнем по-голямо внимание на добрите кадри и ги изместихме по-напред.

Преди (4.2.3.1.4. фиг. 1):



4.2.3.1.4. фиг. 1

4.2.3.1.4. фиг. 2

След (4.2.3.1.4. фиг. 2):

4.2.3.1.5. Импортиране

Импортираме или качваме снимките в HTML страниците ни с помощта на още една програма на Python, която създадохме за да автоматизираме процеса на въвеждане. Целта на програмата е да въведе всички необходими думички и ключови думи и елементи, с които да импортираме снимките в страницата по бърз и лесен начин.

Кода на програмата: (4.2.3.1.5. фиг. 1):

```
1   with open('numbers.txt', 'w') as f:
2       for i in range(1, 126):
3           f.write(f'<a href="album/12/sporten/{i}.webp"></a>\n')
```

4.2.3.1.5. фиг. 1

```
with open('numbers.txt', 'w') as f:
```

4.2.3.1.5. фиг. 2

Кодът отваря файл (ако файлът не съществува, ще създаде нов) с име 'numbers.txt' за запис с помощта на функцията open() с режим 'w'. Режимът 'w' (Write) означава, че файлът ще бъде отворен за запис и всяко съществуващо съдържание ще бъде презаписано (4.2.3.1.5. фиг. 2).

След това кодът преминава през диапазон от цели числа от 1 до 125, използвайки for цикъл. Не е нужно да задаваме точно число до колко да спре да пише. Зависещо от колко снимки имаме в наличност до толкова може да зададем крайното число. Добър вариант, който използвахме е, че копирахме само редовете, които ни трябват (4.2.3.1.5. фиг. 3).

```
for i in range(1, 126):
```

4.2.3.1.5. фиг. 3

В рамките на цикъла методът write() на файловия обект се използва за запис на низ във файла. Низът е HTML anchor таг с празна връзка () и символ за нов ред (\n). Синтаксисът на формат на низ f се използва за вмъкване на текущата стойност на i във връзката (4.2.3.1.5. фиг. 4).

Резултата от програмата намираме в numbers.txt (4.2.3.1.5 фиг. 5).

```
f.write(
```

```
</a>\n')
```

```
<a href="album/12/sporten/1.webp"></a>
<a href="album/12/sporten/2.webp"></a>
<a href="album/12/sporten/3.webp"></a>
<a href="album/12/sporten/4.webp"></a>
<a href="album/12/sporten/5.webp"></a>
<a href="album/12/sporten/6.webp"></a>
<a href="album/12/sporten/7.webp"></a>
<a href="album/12/sporten/8.webp"></a>
<a href="album/12/sporten/9.webp"></a>
<a href="album/12/sporten/10.webp"></a>
<a href="album/12/sporten/11.webp"></a>
<a href="album/12/sporten/12.webp"></a>
<a href="album/12/sporten/13.webp"></a>
<a href="album/12/sporten/14.webp"></a>
<a href="album/12/sporten/15.webp"></a>
<a href="album/12/sporten/16.webp"></a>
<a href="album/12/sporten/17.webp"></a>
<a href="album/12/sporten/18.webp"></a>
<a href="album/12/sporten/19.webp"></a>
<a href="album/12/sporten/20.webp"></a>
<a href="album/12/sporten/21.webp"></a>
<a href="album/12/sporten/22.webp"></a>
<a href="album/12/sporten/23.webp"></a>
<a href="album/12/sporten/24.webp"></a>
<a href="album/12/sporten/25.webp"></a>
<a href="album/12/sporten/26.webp"></a>
<a href="album/12/sporten/27.webp"></a>
<a href="album/12/sporten/28.webp"></a>
<a href="album/12/sporten/29.webp"></a>
<a href="album/12/sporten/30.webp"></a>
<a href="album/12/sporten/31.webp"></a>
<a href="album/12/sporten/32.webp"></a>
<a href="album/12/sporten/33.webp"></a>
<a href="album/12/sporten/34.webp"></a>
<a href="album/12/sporten/35.webp"></a>
```

4.2.3.1.5 фиг. 5

По този начин елиминираме постоянното копиране и поставяне, което улеснява процеса ни, и спестяваме време.

4.3.2