**П****РОФЕСИОНАЛНА ГИМНАЗИЯ “ГЕН. ВЛАДИМИР ЗАИМОВ” гр. СОПОТ**

**Дипломна работа**

на тема

**„Разработка на Уеб сайт организиращ дейността на летище за чартърни полети“**

Дипломант : **Атанас Георгиев Колев**

Специалност : **Системно програмиране**

Ръководител консултант :

**Стоян Пършев**

Карлово , 2022 г.

**СЪДЪРЖАНИЕ**

**УВОД**

**ПЪРВА ГЛАВА**

ПРОУЧВАНЕ НА ……… ПОТРЕБНОСТИТЕ, НУЖДИТЕ, ЦЕЛЕВА ГРУПА…

1.1. Предпоставка за създаване на продукта

1.2. Съществуващи решения и реализации

**ВТОРА ГЛАВА**

ПРОЕКТИРАНЕ НА СТРУКТУРАТА НА WEB БАЗИРАН ……..

2.1. Функционални изисквания към WEB сайт организиращ дейността на летище за чартърни полети /условието на задачата/

2.2. Изготвяне на Use Cases Diagrams

2.3. Потребителски интерфейс /менюта, екрани, Views/

2.4. Описание на избраната технология и софтуерните средства за разработка на приложението /технологии: ASP.Net, Web Forms, Windows Forms App; средства: C#, HTML, CSS, JavaScript/

**ТРЕТА ГЛАВА**

ПРОГРАМНА РЕАЛИЗАЦИЯ НА WEB БАЗИРАН ……………….

3.1. Структурата на базата от данни

3.2. Структура на MVC приложението – Controllers Class Diagram, Routs Table

3.3. Използване на технологии и алгоритми

**ЗАКЛЮЧЕНИЕ**

**Увод**

**Kратко описание на тема и областта на дипломната работа :**

Програмирането е на път да измени своя облик. От чисто

императивно в преобладаващата си част то преминава към

декларативно. Появяват се редица инструменти и технологии за

разработка, при които части от програмата се програмират

императивно, а други се описват декларативно и така се комбинират

преимуществата на двата модела.

Сайта , който ще създам ще е в областта на въздушният транспорт или по точно – летище за чартърни полети , като ще предлага възможност за резервация на самолетни билети до различни дестинации в целият свят , които авиокомпаниите предлагат и ще се поддържа информация за разписанията на полетите.

**Цели и задачи :**

Дипломният ми проект има за цел да се реализира трислойно MVC WEB приложение , организиращо дейността на летище за чартърни полети, което трябва да поддържа CRUD операции за дестинациите, полетите с авиокомпании и клиентски заявки за билети. Сайтът ще разполага със следните публично достъпни секции – „За нас“, „Разписание“, „Резервация“, „Контакти“.

Регистриран в системата клиент ще може да резервира билет/и за полет от разписанието на летището.

Сайтът ще може да се администрира с помощта на администраторско меню, което да позволи добавяне на полет и авиокомпания, изготвяне на разписание на полети и проследяване на резервираните билети от клиентите.

Една от целите и задачите е да се сподпомогне и улесни крайния потребител в запазването на билети до дадена дестинация , като ще има лесен достъп до всичко нужно за едно хубаво и приятно изживяване , от полето до регистрация до момента на запазване на билет.

**ГЛАВА 1.**

**ПРОУЧВАНЕ**

**1.1. Предпоставка за създаване на продукта**

Компаниите, които не успяват да разработят нови продукти, се подлагат на голям риск. С времето съществуващите продукти стават уязвими при променящите се потребности и вкусове на потребителя, новите технологии, скъсения жизнен цикъл на продукта и увеличаващата се конкуренция. Но и разработката на нови продукти също носи след себе си значителен риск. Една компания може да добави нови продукти по два начина: чрез придобиване или като го разработи.

Разработването на нов продукт в организацията трябва да бъде постоянна дейност. Това се налага от факта, че чрез новия продукт организацията може да решава проблеми от следния характер:

- поддържане престижа на организацията, като развиваща се и просперираща в крак с изискванията на времето;

- финансови затруднения - подобряване на паричния поток, печалбата и др.;

- по-пълно използване на технологиите, странични продукти и др.;

- спечелване на конкурентни предимства;

- запазване и разширяване на пазарни дялове и поддържане на постоянен брой клиенти;

- намаляване нивото на риска, ако се премахва зависимостта на организацията само от един продукт;

- бързо реагиране на промените във външната среда;

- преодоляване на цикличността и сезонността в търсенето и др.

За жалост понякога колкото и да е съвършено направен един продукт , то може да не потръгне така както ни се иска. Към момента пазара е изключително голям и гъмжи от голяма конкуренция , това може да е една от причините , поради която нашият продукт не пожънва желания успех, който сме очаквали. Също така това при направата на нов продукт – то няма как да е перфектен , напълно е възможно да има някои технически несъвършенства по направата му , което е напълно нормално , когато правим нещо за първи път , но дори и вече да имаме опит с това – то пак няма как да е напълно съвършено , понеже вече има наложили се сайтове , продукти и др. , които имат дългогодишен опит с това и са се наложили на пазара от дълги години насам. Потребителите вече са си наложили някакви очаквания от даден сайт или продукт и знаят на кого да се доверяват , когато искат да използват дадена услуга.

**1.2. Съществуващи решения и реализации**

За да може даденият ни продукт да има успех , то трябва да се предложи продукт, който макар подобен, носи някаква допълнителна изгода на потребителите или е позициониран по различен начин. С оглед на подобни продукти като моят , забелязвам някои неща които не ми харесват , било то дизайн или функционалност , които неща няма да допусна да са така в моят готов продукт , ще се опитам да направя всичко по най-добрият начин. За да няма дефекти е необходимо да не се пропускат ключовите технически тествания и тестванията на продукта от потребителите. Високата цена на продукта също било попречило за успеха на сайта ни. Затова е важно да е добро съотношението цена-качество и е хубаво да огледаме конкуренцията какво предлага и на каква цена , но ако се чувстваме доста уверени в нашият готов продукт и смятаме , че това , което предлагаме е най-доброто , то можем леко да вдигнем „летвата“ спрямо всичко останало , което се предлага.

**ГЛАВА 2.**

**ПРОЕКТИРАНЕ НА СТРУКТУРАТА НА WEB БАЗИРАН….**

**2.1. Функционални изисквания към WEB базиран......... /условието на задачата/**

Функционалните изисквания на сайта , който ще направя са :

* Организиране на дейността и предлагане на добра инфраструктура на авиокомпании;
* Подобряване на дейността и улесняване на достъпа на потребители до услугите , които предлага;
* Сайтът трябва да предлага възможност за резервация на билети за чартърни полите, които авиокомпаниите предлагат и да поддържа информация за разписанията на полетите.
* Сайтът трябва да разполага със следните публично достъпни секции „За нас“, „Разписание“, „Резервация“, „Контакти“.
* Регистриран в системата клиент трябва да може да резервира билет/и за полет от разписанието на летището.
* Сайтът трябва да може да се администрира с помощта на администраторско меню, което да позволи добавяне на полет и авиокомпания, изготвяне на разписание на полети и проследяване на резервираните билети от клиентите.
* Сайтът трябва да може да се администрира с помощта на администраторско меню, което да позволи добавяне на полет и авиокомпания, изготвяне на разписание на полети и проследяване на резервираните билети от клиентите.

**2.2. Изготвяне на Use Cases Diagrams – описание на отделните на случаи на употреба**

Случаят на употреба играе съществена роля в отделните фази на жизнения цикъл на разработката на софтуер. Случаят на употреба зависи от „Действия на потребителя“ и „Отговор на системата“ на действията на потребителя.

Това е документацията на „Действия“, извършени от Актьора / Потребителя и съответното „Поведение“ на Системата към Потребителя „Действия“. Случаи на употреба може или не може да доведе до постигане на цел от „Актьор / Потребител“ относно взаимодействията със системата.

В процеса на проектиране на софтуерния продукт, диаграмата на потребителските случаи е първата диаграма, която се създава от проектантите, когато се започне проект. Тази диаграма позволява да се опишат на най-високо ниво целите на потребителя, които системата трябва да изпълнява. Тези цели не е необходимо да са задачи или действия, а може да са по общи изисквания към функционалността на системата. С други думи това е техника за определяне на функционалните изисквания на една система. Те описват типичните взаимодействия между потребителите и системата, предоставят описание на начина, по който тя се използва.



**Фиг. 1**

****

Както е показано на **Фиг. 1** администратора има пълен достъп и контрол. Той ще има достъп до всеки потребител. Регистрираните потребители имат достъп до техните часове и основно данни и статистики. Нерегистрираните потребители ще имат достъп само имат до интерфейса на уебсайта.

При отварянето на уебсайта потребителят влиза в началната страница къде то може да влезне в потребителският си профил или да създаде нов. При създаването на нов профил потребителят автоматично става клиент. Администраторски акаунт може да се създаде с вградени данни в приложението. Администратора може да създава нови полети , нови разписания , да проследява резервираните билети от клиентите и да редактира тяхната информация. Той има контрол върху цялата система на сайта. Може да вижда цялата информация и също така различни статистики.

**Use Case - Примери за използване:**

Примерите за използване (Use-cases) са основани на сценарий техника в UML, която идентифицира актьорите в едно взаимодействие и която описва самото взаимодействие.

Множество от примери на използване би трябвало да опише всички взаимодействия със системата. Диаграми на последователностите могат да се използват, за да добавят подробности на примерите за използване като показват последователността от обработки на събития в системата.

**Use case е създаден от множество сценарии**:

**Сценарии**

Сценариите са примери от реалния живот, как системата може да се използва.

Те трябва да включват:

• Описание на началната ситуация

• Описание на нормалния поток от събития

• Описание на това, което не е както трябва

• Информация за други дейности, извършвани в същия момент

• Описание на състоянието, в което сценарият завършва

**Основен сценарий на успех:**

1. Клиент преглежда каталога и избира артикули за покупка.

2. Клиента преминава към потвърждаване на покупката.

3. Клиента попълва информацията за доставка

4. Системата представя пълна информация за цената, включително разходите за

доставка.

5. Клиента попълва данните от своята лична карта.

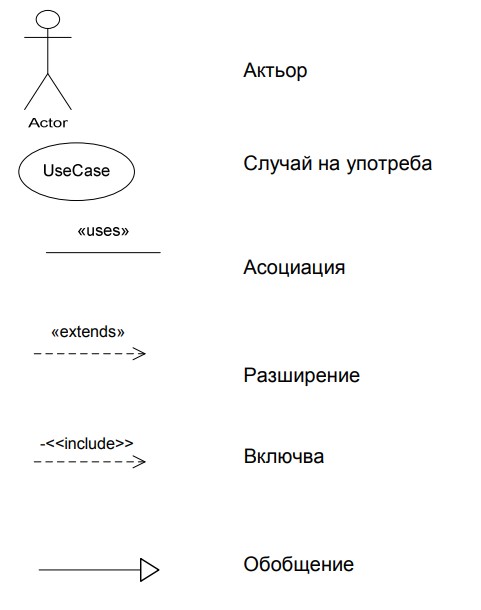
6. Системата оторизира покупката.

7. Системата потвърждава незабавно продажбата.

8. Системата изпраща на клиента потвърждаващо съобщение по e-mail.

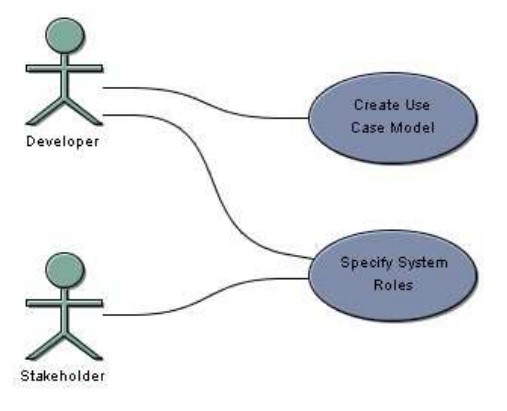
**Компоненти при Use case диаграми:**

**Actor (Актьор)** – роля, която един потребител играе по отношение на системата. Може да бъде клиент, представител по поддръжка на клиенти, мениджър продажби, продуктов аналитик. Актьорите изпълняват случаите на употреба. Един актьор може да изпълнява много случаи на употреба и един случай на употреба може да се изпълнява от много актьори. Актьора клиент може да представлява множество хора. Един човек може да играе ролята на няколко актьора. Актьора може да не е човек, а друга компютърна система, ако системата извършва услуга за нея.

****

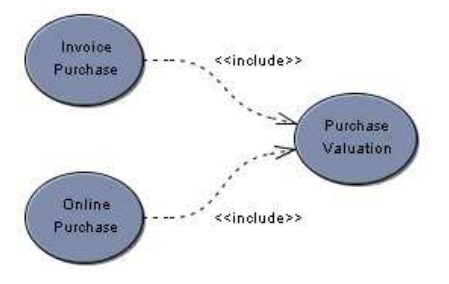
**Фиг.2**

**Associations** – плътна връзка без стрелка. Аssociations са връзка между actors и use cases и означава, че актьора осъществява потребителския случай:

**Фиг. 3** 

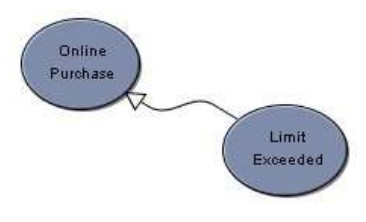
В горната диаграма човечетата са актьорите, а елипсите – потребителски случаи. И двамата потребители разработчик и спонсорът са отговорни за определяне на ролите на системата, но само разработчика създава модела.

**Includes** – Връзка показваща, че един Use sace включва друг Use case. Примера илюстрира как и покупката с фактура и покупката онлайн се включват в сценария описан от покупателната оценка. В обобщение, includes връзката е да отстрани повторенията на сценария в съставните случаи. Може да се каже, че includes е като извикване на процедура.



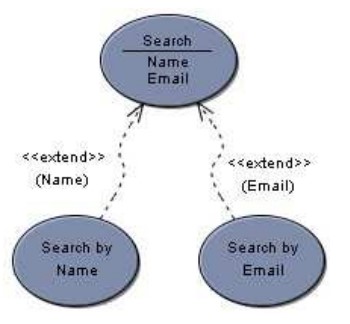
**Фиг. 4**

**Generalization -** Когато един потребителски случай описва разновидност на друг потребителски случай се използва връзката generalization. Тя е специализация на някои случаи. Целта трябва да бъде една и съща за потребителските случаи.



**Фиг. 5**

**Extends -** В някой случаи е възможно да се опишат различията в поведението. Тогава трябва да се дефинира extension points в разширения потребителския случай. Може да се каже, че extend е като извикване на процедура, която се извиква понякога, в зависимост от някакво условие.



**Фиг.6**

**2.3. Потребителски интерфейс /менюта, екрани, Views/**

**2.4. Описание на избраната технология и софтуерните средства за разработка на приложението /технологии: ASP.Net, Web Forms, Windows Forms App; средства: C#, HTML, CSS, JavaScrip, Bootstrap/**

1. **ASP.NET Core :**

ASP.NET Core е безплатен и с отворен код уеб рамка и наследник на ASP.NET, разработен от Microsoft. Това е модулна рамка, която работи и на двете .NET Framework, На Windows, и междуплатформена .NET Core.

Рамката е цялостно пренаписване, което обединява по-рано отделните ASP.NET MVC и ASP.NET уеб API в един модел на програмиране.

Въпреки че е нова рамка, изградена върху нов уеб стек, тя има висока степен на съвместимост на концепцията с ASP.NET. Приложенията на ASP.NET Core поддържат успоредно управление на версиите, при което различни приложения, работещи на една и съща машина, могат да насочват различни версии на ASP.NET Core. Това не е възможно с предишни версии на ASP.NET.

Всяка рамка е насочена към различен стил на разработка. Този, който изберете, зависи от комбинацията от вашите програмни активи (знания, умения и опит в разработката), типа приложение, което създавате, и подхода за разработка, който ви харесва.

ASP.NET MVC ви дава мощен, базиран на модели начин за изграждане на динамични уебсайтове, който позволява ясно разделяне на проблемите и който ви дава пълен контрол върху маркирането за приятно и пъргаво развитие. ASP.NET MVC включва много функции, които позволяват бърза, удобна за TDD разработка за създаване на сложни приложения, които използват най-новите уеб стандарти.

ASP.NET уеб страниците и синтаксисът на Razor осигуряват бърз, достъпен и лек начин за комбиниране на сървърния код с HTML за създаване на динамично уеб съдържание. Свържете се с бази данни, добавете видео, връзки към сайтове за социални мрежи и включете много повече функции, които ви помагат да създавате красиви сайтове, които отговарят на най-новите уеб стандарти.

И трите ASP.NET рамки са базирани на .NET Framework и споделят основна функционалност на .NET и на ASP.NET. Например и трите рамки предлагат модел за сигурност при влизане, базиран на членство, и трите споделят едни и същи средства за управление на заявки, обработка на сесии и т.н., които са част от основната функционалност на ASP.NET.

Освен това трите рамки не са напълно независими и изборът на една не изключва използването на друга. Тъй като рамките могат да съществуват съвместно в едно и също уеб приложение, не е необичайно да видите отделни компоненти на приложения, написани с помощта на различни рамки.

**В заключение** – ASP.Net Core е по-гъвкава, по-пъргава, по-разработена и като цяло по-добра от предишно-запознатата с нас версия ASP .NET Framework.

1. **C# :**

C# е обектно-ориентиран език т.е. той изгражда системи посредством обекти, които взаимодействат помежду си. Обектно-ориентираното програмиране се отличава от традиционното програмиране, при което програмата е просто списък от инструкции, които компютърът изпълнява.

C# е програмен език от високо ниво. При работа с такъв език разработчикът няма нужда от настройване на връзките между детайлите на компютъра и нуждите на програмата т.е. той се концентрира върху самата програма, която създава.

**Къде и за какво се използва?**

- Десктоп приложения;

От Microsoft използват C# във всички свои продукти в това число всичките си десктоп приложения за Windows 8 и 10 т.е. езикът става все по-актуален и широкоизползван, което дава повече шансове за реализация. Това е така, тъй като езикът все пак е разработен като част от инициативата .Net (чете се „Дот нет“) – софтуерната рамка (framework) на Microsoft.

- Уеб приложения;

ASP.NET е софтуерна рамка с отворен код, която също е част от Miscrosoft семейството .NET Framework. Бидейки част от едно и също семейство с програмния език C#, това улеснява тяхното взаимодействие и дори го предполага. Това означава, че от Microsoft се стремят да предразположат разработчиците да използват колкото се може повече MS продукти в разработването на своите приложения и платформи. Доброто взаимодействие между C# и уеб фреймуърка ASP.NET правят програмния език подходящ и за уеб сайтове, платформи и уеб приложения.

- Мобилни и Cross platform мобилни разработки

C# е подходящ също и за създаване на мобилни и междуплатформени приложения. Visual Studio – IDE-то на Microsoft e съвместимо с Xamarin – платформа за разработка на native Android, iOS и Windows приложения.

- Игри

****За много от вас може би това звучи несериозно, но все пак това също е една от възможностите за реализация със С#. Един от най-популярните гейм енджини, подходящ както за PC, така и за конзолни и мобилни приложения е Unity, който е микс от няколко езика от семейството на „С“ – С, С++ и разбира се С#. С това далеч не се изчерпват игровите енджини, писани частично или изцяло на C# и това също е една добра насока за развитие.

1. **HTML:**

**Как работи HTML?**

HTML ни помага да зададем основната структура на дадена уеб страница. Също така е полезен за да редактираме параграфи, заглави и други основни елементи на една уеб страница. Когато някой кликне на линк в страница или въведе нов URL в адресното поле, браузърът изпраща заявка за документ от Web server-a, където е качена страницата, посредством Hypertext Transport Protocol (HTTP). Сървърът след това изпраща документа обратно до потребителя, като се визуализира на дисплея от браузъра. Всички неща, които са на този документ – текст, снимки, аудио или видео файлове – всички те са подредени посредством HTML структура.

**Какво е HTML 5 и какви са неговите предимства?**

HTML5 е осъвременена версия на едноименния език. Подобренията на HTML5 спрямо предходните версии са много, но ето най-ключовите от тях:

- Подобрен код – HTML5 предлага на уеб дизайнерите да използват по-чист, по-хубав код. Вече може да се премахнат <div> таговете и да се заменят със семантични HTML5 елементи.

- По-елегантни формуляри – Опциите за полета за въвеждане на потребителски данни са много по-гъвкави в сравнение с преди.

****- Поддръжка на мултимедия – Кажете „сбогом“ на „плъг-ин“-ите в кода за аудио и видео. HTML5 има вградени опции за работа с аудио и видео файлове.

1. **CSS:**

CSS езикът е език, който определя стила на HTML документите. Той включва опции, свързани с шрифтове, цветове, полета, линии, височина, ширина, фонови изображения, наред с други.

В днешно време е възможно да се използва HTML език за разработване на формата на уеб страници. Езикът CSS обаче предлага повече опции и е по-точен, освен че е съвместим с всички текущи браузъри.

**Значение на използването на езика CSS:**

CSS езикът се превърна в революция в областта на уеб дизайна, особено. Сред предимствата, открити при използването му, се открояват следните:

* Повече точност: Когато използвате CSS, размерът и позиционирането на елементите, които изграждат мрежата, ще бъдат точни. Можете да кажете на браузъра в кой пиксел да поставите определено изображение, както и техните измервания.
* CSS дава по-добра достъпност и структура. Чрез комбиниране на CSS език и описателни маркери е възможно уебсайтът да се вижда правилно, тъй като информацията ще остане структурирана и подредена.
* Подобрете времето за зареждане: С въвеждането на CSS, съдържанието и външният вид са разделени, което води до по-леки файлове. Това е изгодно за намаляване на времето за зареждане на сайта в браузъра и за намаляване на обема на трафика на сървъра, който е избран.

1. **Java Script :**

**За какво се използва JavaScript?**

Функциите на JavaScript са се разширили значително през годините. Първоначално езикът просто е регулирал начинa, по който потребителят си взаимодейства с уебсайта: онлайн въпросници и форми за контакт, подаване на обратна връзка под формата на известия и важни съобщения и др. Днес обаче езикът е еволюирал до такава степен, че с него може дори да управлявате дрон. За какво още се използва JavaScript:

* Създаване на мобилни приложения
* Разработване на десктоп и уеб апликации
* Зареждане на информация чрез AJAX
* Изработване на операционни системи за смарт часовници
* Разпознаване на използвания браузър и променяне на уеб страниците спрямо това
* Анимиране на елементи от уеб страниците
* Създаване на поп-ъп прозорци
* Изготвяне на презентации под формата на уебсайт
* Периодично обновяване на променливи данни без намесата на потребителя
* Функционира дори и офлайн
* и още, и още

**В заключение :**

Предимствата на JavaScript са много. Именно те го превръщат в един от любимите езици за програмиране и на нашите девелъпъри.. Освен че разширява функционалността на уеб страниците, той е изключително лесен език за научаване и практикуване. Насочен е изцяло към осигуряване на добро потребителско преживяване, при това бързо и качествено.

1. **Bootstrap :**

Bootstrap е платформа , която е комбинация от HTML, CSS и JavaScript код.

Bootstrap напълно поддържа CSS3 и HTML5 и вече се превръща в стандарт за уеб разработки.

**Ползите от него са:** Bootstrap е съвместим с всички основни браузъри. Bootstrap сайтовете се адаптират много лесно за всякакви устройства. Използвайки комбинация от JavaScript и CSS, Bootstrap позволява да се изпълни почти всеки елемент от дизайна, който е необходим за съответния проект. Bootstrap е съвместим с всички основни браузъри. Bootstrap сайтовете се адаптират много лесно за всякакви устройства. Използвайки комбинация от JavaScript и CSS, Bootstrap позволява да се изпълни почти всеки елемент от дизайна, който може да се изиска.

**ГЛАВА 3.**

**ПРОГРАМНА РЕАЛИЗАЦИЯ НА WEB БАЗИРАН ……………….**

**3.1. Структурата на базата от данни**

**3.2. Структура на MVC приложението – Controllers Class Diagram**

- Описание и конфигуриране на моделите за работа и обмен на данни между контролери и изгледи

- Описание на CRUD операциите- особености на отделните ACTION

- Работа с View моделите (където е необходимо). Разлика с използването на ViewBag и ViewData ( описание на двата обекта)

- Описание на Razor Pages и Razor Views (TagHelpers и HtmlHelpers)

- Описание и конфигурация на Services (ако е приложимо)

**3.3. Използване на технологии и алгоритми**

- Как работи схемата Routs Engine

- Как работи шаблина на Contoler с GET и POST заявки към Actions. Извикване на конкретни View, изпозвайки подходящ Model.

- Алгоритми за филтриране и сортиране

**ЗАКЛЮЧЕНИЕ**

Общение на реализирания проект, предимства и недостатъци, теденции за бъдещи разработка и развитие

**ИЗПОЛЗВАНА ЛИТЕРАТУРА**

**Примери:** /трябва да се следва точно формата/

1. <https://www.bg-ikonomika.com/2010/11/20_14.html>
2. <https://bg.myservername.com/mysql-data-types-what-are-different-data-types-mysql>
3. <https://ff.tu-sofia.bg/~bogi/StopFak/SoftEngr/uml_lab1.pdf>
4. <https://docs.microsoft.com/en-us/aspnet/overview>
5. <https://softuni.bg/blog/what-is-c-sharp-and-where-to-use-it>
6. <https://softuni.bg/blog/what-is-html5-and-how-it-works>
7. <https://bg.economy-pedia.com/11036001-css-language>
8. <https://speedflow.bg/blog/what-is-javascript-what-it-is-used-for/>