# Резюме на дипломния проект

1. Представяне

Уважаема г-жо Директор, Уважаема комисия, уважаеми гости (ако има такива/ или поименно – ако е един конкретен гост)

Аз съм Петър Павлевичин.

Темата на моят дипломен проект е „Разработване на сайт за електронна търговия с компютри и компютърни компоненти, фокусиран най-вече върху вграден конфигуратор - приложение за съставяне на компютърна конфигурация от наличните компоненти“

1. Увод

Идеята за създаване на уеб сайт за електронна търговия, специализиран в компютърни системи и компютърни компоненти, е логично следствие от бързото развивитие на пазар на технологии. Такъв сайт може да предлжи богато разнообразие от продукти, включително процесори, видеокарти, дънни платки, рам памет, твърди дискове и много други компоненти, които потребителите избират според техните нужди, бюджет и предпочитания.

Важна за такъв сайт е и възможността да се предлагат допълнителни услуги. Това допълнително засилва доверието на потребителите към марката и увеличава лоялността им. Такъв уеб сайт представлява не само място за пазаруване, но и платформа за създаване на персонализирани решения и удовлетворяване на специфичните нужди на потребителите в сферата на компютърните технологии.

1. Обща теория: MVC

От гледна точка на начина на създаване на такъв сайт първо трябва да се каже, че има множество платформи, фреймуъркове и др. инструменти, създадени специално за тази цел. И за всеки от тях може да са каже и напише много. Но почти всички следват един и същ архитектурен шаблон за проектиране на софтуерни приложения – MVC.

Същността на MVC e в разделяне на приложението на три основни компонента: модел (Model), изглед (View) и контролер (Controller). Вотът:

Модел (Model) - съдържа логиката за обработка на данни.

Изглед (View) - представлява потребителския интерфейс на приложението.

Контролер (Controller) - представлява посредник между модела и изгледа и отговаря за обработката на потребителските заявки и управлението на потокът на данни в приложението.

MVC позволява ясно разделяне на отговорностите между модела, изгледа и контролера, което прави кода по-организиран и лесен за разбиране и поддръжка.

1. Обща теория: MySQL

Заговори ли се за данни, на преден план излиза терминът База Данни и първата асоциация тогава е MySQL.

MySQL е една от най-популярните релационни бази от данни (RDBMS), която се използва широко за съхранение, управление и извличане на данни в уеб приложения и други видове софтуерни системи. Тя се предлага като предварително инсталирано приложение от болшенството хостинг компании. Това допълнително засилва популярността и.

MySQL поддържа стандартни SQL език за манипулиране на данни, което го прави съвместим с множество различни софтуерни приложения и инструменти. Освен това, MySQL поддържа и стандартни мрежови протоколи като TCP/IP, HTTP и други, което позволява на разработчиците да се свързват и комуникират с базата данни от различни платформи и програмни езици.

1. Обща теория: REST

А стане ли дума за технологии не можем да не отбележим REST.

REST (Representational State Transfer) е архитектурен стил за проектиране на разпределени системи, които се използват в уеб разработката.

REST архитектурата се фокусира върху ресурсите, които могат да бъдат идентифицирани чрез уникални URI адреси (Uniform Resource Identifiers).

REST използва стандартните HTTP методи, като GET, POST, PUT, DELETE и други, за да извършва операции върху ресурсите.

REST няма междинни състояния, което означава, че всеки заявка към сървъра съдържа всичката необходима информация за изпълнението на операцията.

1. Описаниe: Изборът на технология

REST е в основата на повечето сайтове за електронна търговия. Това е и изборът ми за реализиране на моя проект. Конкретно използвам библиотеката Django-REST-framework известна като DRF. Този избор е предопределен от избора на платформа – Python и Django защото са най-познати, а според статистиката и едни от най добрите за целта. От страна на фронтенда за основа избрах подходящ html шаблон, базиран на Bootstrap. Към него са добавени необходимите функциии посредством VUE. A достъпът до услугите на REST API на сървъра са реализирани посредством библиотеката Axios.

1. Описаниe:Структурата

До толкова, доколкото конкретния проект има за цел да реализира само определена функционалност а не цялостен и завършен он-лайн магазин, той се сътои от три основни части: фронтенд на магазин, конфигуратор и административен панел.

Магазинът и конфигураторът ползват „обща“ количка за пазаруване. Двата скрипта, които реализират функциите на VUE за магазина и за конфигуратора са отделени в отделна папка за по добра пригледност.

Администраторският панел е мястото, от което се управлява сайта – следят се поръчките, добавят се стоки и т.н

1. Описание: Backend поглед

От страна на backend-а искам да обърна внимание на ‘urls.py’ Той играе ролята на маршрутизатор, който съпоставя URL адресите с изгледите (views) или функциите в приложението, които обработват заявките.

В контекста на уеб приложението, което предлага API, имаме дефиниран маршрут за всяка функция от интерфейса. Когато се направи заявка към сървъра, Django проследява дефинираните маршрути и ако намери съвпадение извикава съответстващата функция, която от своя страна връща JSON отговор с информация за категориите.

Това е основната идея зад изграждането на API в Django.

1. Заключение

Считам работата по проекта за успешна. Поставените в заданието задачи са решени.

В бъдеще, онлайн магазинът може да се развие като пълноценнаа платформа за персонализирано изграждане на компютърни системи, което да включва не само хардуерни компоненти, но и софтуерни и сервизни услуги. Той би могъл, например, да предложи интелигентни алгоритми за препоръчване на оптимални конфигурации в зависимост от нуждите и предпочитанията на потребителите.

1. Финал

Благодаря за вниманието!

Ако имате въпроси съм готов да отговоря.