# 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 Мария Дашева.

Темата на моят дипломен проект е „Разработване на web-базирано приложение „Борса за second hand дрехи“.“

1. Увод

Съществуват множество причини, поради които хората търсят или са склонни да купят дрехи втора употреба. Има и много хора, които могат да предложат такива дрехи.. И тука идва идеята да се направи софтуерна он-лайн платформа, която да срещне тези две групи хора директно, без посредници. Това е идеята, която се опитва да реализира моят проект.

1. Обща теория: SDI vs MDI

Интерфейс с единичен документ (SDI) е парадигма за потребителски интерфейс, при която приложението показва само един документ или изглед едновременно. За разлика от Множествените документни интерфейси (MDI), при които могат да бъдат отворени едновременно множество документи или изгледи в едно и също прозорец, SDI се фокусира върху представянето на единичен документ или изглед в отделен прозорец.

Още в самото начало беше решено за моето приложение да използвам SDI.

1. Обща теория: VUE - въведение

В контекста на уеб-базираните разработки терминът Single Document Interface придобива по-специфично значение, като ни препраща към шаблон за проектиране който разделя данните и представянето. Това води до изнасяне на функционалност във фронтенда. Задачата намира своето решениие чрез използване на подходящи софтуерни рамки. В конкретния случай се спрях на VUE.

Vue.js е изключително популярна библиотека за създаване на уеб приложения, която е базирана на JavaScript. Vue.js предоставя едно от най-добрите решения за създаване на модерни, динамични и многофункционални уеб приложения. Това е поради факта, че Vue.js е много лека библиотека, която може да бъде използвана за разработване на малки до средно големи приложения.

Vue.js е изключително гъвкава и лесна за употреба библиотека, която се интегрира лесно с други технологии като, например, Bootstrap.

1. Обща теория: DRF и Axios

Фактът, че VUE лесно се интегрира в HTML, а слeдователно и в Django не е достатъчен. Необходимо е също така да се построи съответния бекенд и да се организира комуникацията между тях. Това по същество е REST. А стане ли дума за REST и Django изборът е DRF – Django rest framework.

И, естествено, комуникацията. Тук едно добро решение е Axios – лека и удобна библиотека за програмно генериране на заявки към сървъра.

1. Описаниe: Визия фронтенд

Практическата разработка обикновенно започва от разработване на визуален макет на приложението. WEB-дизайна и WEB-разработката в момента са се отдалечили до толкова едно от друго, че са две различни професии. Разработчиците обикновенно ползват услугите на дизайнер, но се предлагат и т.н. „готови“ HTML шаблони. Аз също ползвам такъв, но с известни корекции. Променена е леко визията, но най-важното, променена е структурата. За да се подчертае идеята за еднодокументен интерфейс, цялото приложение има един единствен html файл. Страниците, които потребителят вижда са всъщност секции в този базов шаблон.

1. Описаниe:Функции фронтенд

Може би най-голямото предимство на избрания подход е ясната структура – един html файл събира в себе си цялото визуално представяне и един JavaScript файл внася цялата необходима функционалност. Така работи VUE – съпоставя един JavaScript обект на един html таг. Промените в обекта водят до промени в в това, което потребителят вижда в браузъра, а неговите реакции се предават същия този обект за обработване. Ако е необходимо да се получат или изпратят данни от или до сървъра този обект използва axios. Това малка, но функционална библитека за асинхронна комуникация клиент-сървър.

1. Описание: Backend поглед

Обменят се само данни. От страна на сървъра използваме DRF. С нейна помощ изграждаме т.н. API – списък от възможни заявки, на които сървърът трябва да отговори като изпрати определн набор от данни или извърши някакво действие – например, добавяне на нов запис в БД. Звучи сложно, но с помощта на т.н сериализатори на DRF заадачата е почти тривиална. Само указваме кои полета от коя таблица. Останалото е работа на DRF – извличане на данните, сериализира, формиране на JSON и връщане на отговор на клиента.

1. Заключение

Ако обобщим - успешното съчетаване на подходящи технологии позволява даже и на програмисти с по-малък опит да създадат еднодокументно приложение, което изглежда добре, функционира правилно и реализира всички поставени в заданието изисквания.

Считам, че поставената задача е решена успешни и изцяло.

За такъв тип приложения обикноовенно се използват по-тежки, но и по-мощни, библиотеки като Angular и React. Също така най-често се използва node.js или друга подобна близка до JavaScript среда за разработване на съответния backend. Но, както показва и моето приложение, Python и Django също са добър избор.

Приложението може да се доразвие в бъдеще в редица посоки. Например, може да се добави автоматично уведомяване по е-mail или със SMS на подалият обява, може също да се добави допънителна защита от създаването на „фалшиви“ акаунти. Би могло да се добавят и други функции, но това ще надхвърли значително изискванията на заданието.

Във вида в който е представено приложението е напълно функционално и би могло да се използва и в реална среда.

1. Финал

Благодаря за вниманието!

Ако имате въпроси съм готов да отговоря.