**Съдържание**

[1. Увод 3](#_Toc133546248)

[2. Website 3](#_Toc133546249)

[2.1. Bидове уебсайтове 5](#_Toc133546250)

[2.1.1. Статичен уебсайт 5](#_Toc133546251)

[2.1.2. Динамичен уебсайт 6](#_Toc133546252)

[2.1.3. Предимства на динамичния уеб сайт 7](#_Toc133546253)

[2.1.4. Недостатъци на динамичните уебсайтове 8](#_Toc133546254)

[2.1.5. Статичен или динамичен сайт ? 9](#_Toc133546255)

[2.2. Мултимедия и интерактивно съдържание 11](#_Toc133546256)

[3. HTML 12](#_Toc133546257)

[3.1. Какво е HTML 12](#_Toc133546258)

[3.2. CSS и скриптове 13](#_Toc133546259)

[3.3. Редактори 13](#_Toc133546260)

[3.3.1. Текстови редактори 13](#_Toc133546261)

[3.3.2. Обектни редактори 14](#_Toc133546262)

[3.3.3. WYSIWYG HTML редактори 14](#_Toc133546263)

[**4.** Python 14](#_Toc133546264)

[4.1. Python frameworks и библиотеки 17](#_Toc133546265)

[**4.2.** Защо използваме Frameworks 18](#_Toc133546266)

[4.3. Видове frameworks 19](#_Toc133546267)

[4.4. Framework за уеб приложения 19](#_Toc133546268)

[4.5. Използване на Python 20](#_Toc133546269)

[5. Django framework (Python) 21](#_Toc133546270)

[5.1. Използване на Django 22](#_Toc133546271)

[5.2. История 24](#_Toc133546272)

[**5.3.** Колко популярен е Django 24](#_Toc133546273)

[**5.4.** Как изглежда кодът на Django 25](#_Toc133546274)

[6. PyCharm 26](#_Toc133546275)

[**6.1.** За какво се използва PyCharm 27](#_Toc133546276)

[6.2. Характеристики на PyCharm 28](#_Toc133546277)

[6.3. Какво е IDE 30](#_Toc133546278)

[7. Реализация 30](#_Toc133546279)

[7.1. Анализ на заданието 30](#_Toc133546280)

[7.2. Структура 31](#_Toc133546281)

[7.2.1. Модели 31](#_Toc133546282)

[7.2.2. Изгледи 32](#_Toc133546283)

[7.2.3. Заявки 34](#_Toc133546284)

[8. Заключение 34](#_Toc133546285)

[9. Списък на използваната литература 35](#_Toc133546286)

# Увод

Изработения от мен сайт е сайт на художествено ателие, подпомагащо начинаещи артисти. Сайта е за всеки начинаещ ходожник който винаги е искал да се потопи в изкуството, но не е знаел от каде да започне или просто за всеки, който цени доброто изкуство.

Тук идва сайта ни с лесните за използване итерактивни уроци за начинаещи и пълната ни динамична галерия каято използва база дани с красиви рисунки лично нарисувани от мен самата. Самия уебсайт е красиво изработван на Python кой е модерен, лек за научаване и с голям брой frameworks език за програмиране.

А за framework избрах Django - най добрата framework for web applications. Django е най-добрия frameworks за уеб приложения, тъй като позволява на разработчиците да използват модули за по-бързо създаване на динамични сайтове като този. Самия код е написан на PyCharm.

# Website

Уеб сайтът е колекция от публично достъпни, свързани помежду си уеб страници, които споделят едно име на домейн. Уебсайтовете могат да бъдат създадени и поддържани от индивид, група, бизнес или организация, за да служат за различни цели.

Заедно всички публично достъпни уебсайтове съставляват световната мрежа.

Въпреки че понякога се нарича „уеб страница“, това определение е грешно, тъй като уебсайтът се състои от няколко уеб страници. Уеб сайтът е известен още като „уеб присъствие“ или просто „сайт“.

Уебсайтовете се предлагат в почти безкрайно разнообразие, включително образователни сайтове, новинарски сайтове, порно сайтове, форуми, сайтове за социални медии, сайтове за електронна търговия и т.н. Страниците в рамките на един уебсайт обикновено са комбинация от текст и други формати. Това доказва, че няма правила, диктуващи формата на уебсайт.

Човек може да създаде уебсайт само с черно-бели снимки на рози или думата "котка", свързана с друга уеб страница с думата "мишка". Много сайтове обаче следват стандартен модел на начална страница, която препраща към други категории и съдържание в уебсайта.

Началната страница представлява главната страница на самия сайт. Често началната страница е нещо като „хъб“, от който можете да получите достъп до всички други страници. Вместо това вътрешна уеб страница, към която няколко други страници са свързани в съгласувана структура (като конкретна категория теми), се нарича „родителска страница“.

Всяка страница е единичен HTML документ и всички те са свързани чрез хипервръзки (или просто „връзка“), които могат да се комбинират в навигационна лента за по-лесно използване.

Навигационната лента се показва на всяка страница, а не само на началната страница, и позволява на потребителя бързо да се движи през основната структура на уебсайта.

Друг важен раздел на повечето уебсайтове е долният колонтитул, който е друг повтарящ се раздел, който се намира в долната част на всяка страница. Обикновено долният колонтитул съдържа външни връзки, сочещи към подобни уебсайтове и други външни ресурси, заедно с друга жизненоважна информация като отказ от отговорност, връзки към условията на услугата, политиката за поверителност и страниците за контакт, както и физическия адрес на компанията, която притежава сайта .

Уебсайтовете се хостват на сървъри и изискват уеб браузър като Chrome, Firefox, Internet Explorer, Edge или друг, за да бъдат посетени (на компютър или мобилно устройство).

Уеб сайт може да бъде достъпен директно чрез въвеждане на неговия URL адрес или чрез търсене в търсачка като Google или Bing.

Първоначално уебсайтовете бяха категоризирани според техните домейни от първо ниво.

Някои примери включват:

* Уебсайтове на държавни агенции = .gov
* Уебсайтове на образователни институции = .edu
* Уебсайтове на организации с нестопанска цел = .org
* Търговски уебсайтове = .com
* Информационни сайтове = .info

Въпреки че тези разширения на домейни от първо ниво все още съществуват, те казват малко за действителното съдържание на уебсайта.

В съвременния интернет разширението „.com“ е най-популярният домейн, заедно с много други разширения, специфични за всяка държава (.it, .de, .co.uk, .fr и др.).

Първият уебсайт е създаден през 1990 г. от Тим ​​Бърнърс-Лий, британски физик в CERN. 3 години по-късно, през 1993 г., CERN обяви, че всеки може да има достъп и да използва безплатно световната мрежа.

## Bидове уебсайтове

### Статичен уебсайт

Статичен уебсайт е този, който има уеб страници, съхранени на сървъра във формат, който се изпраща на клиентски уеб браузър. Основно е кодиран в Hypertext Markup Language (HTML); Каскадни стилови таблици(CSS) се използват за контролиране на външния вид извън основния HTML. Изображенията обикновено се използват за създаване на желания външен вид и като част от основното съдържание. Аудио или видео може също да се счита за „статично“ съдържание, ако се възпроизвежда автоматично или като цяло е неинтерактивно.

Този тип уебсайт обикновено показва една и съща информация на всички посетители. Подобно на раздаването на печатна брошура на клиенти или клиенти, статичният уебсайт обикновено предоставя последователна, стандартна информация за продължителен период от време. Въпреки че собственикът на уебсайта може да прави периодични актуализации, това е ръчен процес за редактиране на текст, снимки и друго съдържание и може да изисква основни умения за проектиране на уебсайтове и софтуер.

 Прости формуляри или маркетингови примери за уебсайтове, като класически уебсайт , уебсайт с пет странициили уебсайт с брошура често са статични уебсайтове, тъй като те представят предварително дефинирана, статична информация на потребителя.

Това може да включва информация за компания и нейните продукти и услуги чрез текст, снимки, анимации, аудио/видео и навигационни менюта.

Статичните уебсайтове все още могат да използват включвания от страна на сървъра (SSI) като удобство за редактиране, като например споделяне на обща лента с менюта на много страници. Тъй като поведението на сайта към читателя все още е статично, това не се счита за динамичен сайт.

### Динамичен уебсайт

[](https://en.wikipedia.org/wiki/File:Server-side_websites_programming_languages.PNG) Динамичният уебсайт е този, който се променя или персонализира често и автоматично. Динамичните страници от страна на сървъра се генерират „в движение“от компютърен код, който произвежда HTML (CSS са отговорни за външния вид и следователно са статични файлове). Има широк набор от софтуерни системи, като CGI, Java Servlets и Java Server Pages (JSP), Active Server Pages и ColdFusion (CFML), които са налични за генериране на динамични уеб системи и динамични сайтове. Различни рамки за уеб приложения и системи за уеб шаблони са налични за езици за програмиране с обща употреба като Perl, PHP, Python и Ruby за по-бързо и по-лесно създаване на сложни динамични уебсайтове.

Един сайт може да показва текущото състояние на диалог между потребителите, да наблюдава променяща се ситуация или да предоставя информация по някакъв начин, персонализирана според изискванията на отделния потребител. Например, когато се поиска началната страница на новинарски сайт, кодът, работещ на уеб сървъра, може да комбинира съхранени HTML фрагменти с новинарски истории, извлечени от база данни или друг уебсайт чрез RSS, за да създаде страница, която включва най-новата информация.

Динамичните сайтове могат да бъдат интерактивни, като използват HTML формуляри , съхраняват и четат бисквитки на браузъраили чрез създаване на поредица от страници, които отразяват предишната история на кликванията. Друг пример за динамично съдържание е, когато уебсайт за продажба на дребно с база данни от медийни продукти позволява на потребителя да въведе заявка за търсене, например за ключовата дума Beatles .

 В отговор съдържанието на уеб страницата спонтанно ще промени начина, по който е изглеждало преди, и след това ще покаже списък с продукти на Beatles като компактдискове, DVD дискове и книги. Динамичният HTML използва JavaScript код, за да инструктира уеб браузъра как да променя интерактивно съдържанието на страницата. Един от начините за симулиране на определен тип динамичен уебсайт, като същевременно се избягва загубата на производителност от инициирането на динамичния двигател на базата на потребител или на връзка, е периодичното автоматично регенериране на голяма поредица от статични страници

### Предимства на динамичния уеб сайт

Сега, след като разбрахте какво е динамичен уебсайт, нека да разгледаме някои от предимствата на изграждането на такъв. Това ще ви помогне да прецените по-добре дали те са подходящи за вашия бизнес. Някои от тези предимства са.

* По-добро потребителско изживяване

Динамичният уебсайт е проектиран да бъде много удобен за потребителя, позволявайки на потребителите лесно да правят промени в уебсайта в зависимост от техните изисквания. Тези сайтове са интерактивни по природа, което позволява на посетителите да оставят коментар или отговор въз основа на опита си, докато са на уебсайта. Това от своя страна помага на създателя да знае за рецензиите в реално време, които след това може да подобри. Изграждането на интуитивна навигация, добавянето и надграждането на функционалности, приспособяването на изживяването при сърфиране на всеки посетител и т.н., всичко това може да се направи лесно, подобрявайки цялостното изживяване.

* По-лесна поддръжка

За динамични уебсайтове се използва система за управление на съдържанието, която има база данни за всички елементи (включително съдържанието на страницата). Те също така използват скриптови езици, които основно взаимодействат с цялата информация на базата данни, което прави възможно извършването на промени в целия сайт, без да е необходимо да правите промени една по една. Качването на съдържание, редактирането на дизайнерски елементи или фонове могат да се извършват относително лесно.

* Лесно актуализиране

Едно от най-търсените предимства на изграждането на динамичен уебсайт е, че ви позволява лесно да актуализирате вашата информация/съдържание в зависимост от вашите изисквания. Не се нуждаете от много сложни технически познания, за да направите промени в сайта. Във вашия шаблонен файл, след като направите една промяна, тя автоматично засяга всички страници с този конкретен файл. В сравнение със статичен сайт, динамичните страници могат бързо да се актуализират, за да станат отзивчиви на различни екрани/платформи.

* Професионално изглеждащ

Динамичните уебсайтове ви дават издигнат вид и изглеждат по-професионални от статичните уебсайтове. Това обикновено се дължи на факта, че на статичните сайтове липсват доста функции и елементи, които потребителите очакват от съвременните сайтове. Функционалността, интерактивността и лекотата на редактиране правят възможно динамичните уеб сайтове да изглеждат много по-професионално.

* Футуристично

Динамичните уебсайтове дават възможност на потребителя да бъде по-ангажиран и ангажиран със съдържанието на сайта. Като им предлагат интерактивно и отзивчиво изживяване, те ги издигат нагоре чрез персонализирано сърфиране. Този вид въздействащо преживяване е по-вероятно да остане за дълго време, за разлика от статичните стени от текстове и изображения.

### Недостатъци на динамичните уебсайтове

Като всеки технологичен напредък, има някои трудности, с които човек може да се сблъска с динамичните уебсайтове, като например:

* Скорост

Тъй като статичната HTML страница не изисква външен ресурс, тя се зарежда много по-бързо всеки път, отколкото в сравнение с динамична страница, която се генерира от страна на сървъра.

* Трябва да се промени

Един основен недостатък на тези сайтове е, че те трябва да бъдат променяни през определен период.

* Може да бъде по-скъпо

Технологията за автоматизация, която разработчиците изискват, не винаги е лесна за получаване. Добавете към него бюджетите на разработчиците, маркетинговите разходи, маркетинга на съдържанието и т.н., които са трудни за намиране, и имате рецептата за висок бюджет. Въпреки това, ако имате добра компания за разработка на уебсайтове, която ви помага по време на вашето пътуване, този недостатък може да бъде смекчен.

* Мислите ли да създадете динамичен уебсайт за вашия бизнес

Информацията по-горе адресира въпроси като „Какво е динамичен уебсайт? “ и „какво е статичен уебсайт“ и, надяваме се, ви е дал ясна представа дали изборът на услуги за разработка на уебсайтове за изграждане на динамичен уебсайт би бил от полза за вашия бизнес. За да повторя, с прости думи, ако планирате да стартирате уебсайт или магазин за електронна търговия, които смятате, че ще изискват интерактивност и ангажираност, изборът на динамичен уебсайт е правилният начин. Ако искате нещо, което е чисто информационно, например фирмен сайт с брошура, тогава статичен уебсайт ще работи добре.

Независимо дали сте сигурни в решението си или имате съмнения относно това, което е най-подходящо за вашия бизнес, получаването на консултация от правилната компания за разработка на уебсайтове никога не може да навреди.

### Статичен или динамичен сайт ?

Както беше отбелязано по-горе, уеб сайтовете се делят на два вида според начина им на изграждане и функционалностите, с които разполагат – статични или динамични сайтове.

Отличителна характеристика при статичните уеб сайтове е липсата на система за управление на съдържанието (CMS), което означава, че процесът по изграждане и развитие на сайта се случва само посредством промяна в неговия код.

Най-често статични са личните уеб сайтове, както и малките презентационни и информационни страници, които не се нуждаят от ежедневно обновяване на съдържанието. Такива сайтове са изградени основно чрез езика HTML, чрез който се указва структурата на сайта и страниците в него – меню, текстови полета, изображения и др. За по-добър изглед на съдържанието на помощ идва и CSS езикът, който се използва за визуалното представяне и стилизиране на информацията. На следващ етап уеб сайтът може да се сдобие и с интерактивност посредством JavaScript технологията.

Освен, че са сравнително лесни за създаване и използване, статичните сайтове имат и някои ключови предимства, от които е много по-лесно да се възползвате в сравнение с динамичните.

* **Бързина на уеб сайта** – опростената функционалност и структура на статичните уеб сайтове води и до много по-добра скорост на зареждане спрямо динамичните.
* **Добра SEO оптимизация** – при правилно изграждане и структуриране на статичен сайт всяка от страниците му би била лесна за индексиране от търсещите машини.
* **Добро ниво на сигурност и защита от злонамерени атаки -**При този вид сайтове, липсва PHP или база данни, което означав, че страницата става много по-малко уязвима към хакерски атаки.
* **По-малък файлов архив и липса на база данни** – това позволява лесна и бърза миграция на съдържанието, улеснено възстановяване (backup) на данни, както и възможност за съхранение на локално архивно копие на сайта.
* **Развихрено въображение** – на практика, няма ограничения пред това да създадете сайт с дизайн по ваша представа и вкус.

Но с развитието на уеб технологиите става ясно, че статичните уеб сайтове далеч не могат да отговорят на потребностите на бизнесите да ангажират своите клиенти и, най-вече, на потребителите - за да имат едно приятно преживяване на сайтовете.

* **Ограничена функционалност** – статичният ви сайт просто няма да бъде „способен“ да изпълнява определени функции и няма да разполага с всички необходими инструменти, които потребителите са свикнали да използват.
* **Усложнена актуализация на съдържанието** – такава може да бъде извършена само ръчно посредством намесата на уеб разработчик, който добре познава структурата на сайта.

Динамичните сайтове са доста по-популярният вид уеб сайтове за бизнеси и потребители просто защото те предлагат много по-модерен изглед, функционалности и възможности за взаимодействие. При динамичните сайтове *съдържанието се намира в бази данни*, които динамично „сервират“ информацията при зареждане на всяка уеб страница от браузъра на потребителя.

Вече се досещате, че основната разлика за собственика на сайта е наличието на система за управление на съдържанието (cms), която дава много повече възможности за лесно управление на сайта и създаване на съдържанието. Естествено трябва да споменем, че не винаги липсата на CMS означава, че сайта е статичен (пример за това може да бъде Laravel)

Динамичните сайтове разполагат и с някои незаменими ползи спрямо статичните:

* Лесно управление и гъвкавост – вече споменахме, че създаването, редактирането, оптимизацията и управлението на цялото съдържание може да се случи през системата за управление на съдържанието дори от човек без опит в уеб разработването.
* Разнообразие от допълнителни функционалности – уеб сайтовете вече разполагат с редица сложни функционалности, които правят потребителското изживяване все по-стойностно, приятно и полезно.

## Мултимедия и интерактивно съдържание

Ранните уебсайтове имаха само текст, а скоро след това и изображения. След това бяха използвани добавки за уеб браузър за добавяне на аудио, видео и интерактивност (като например за богато уеб приложение , което отразява сложността на настолно приложение като текстообработваща програма ). Примери за такива добавки са Microsoft Silverlight, Adobe Flash Player, Adobe Shockwave Player и Java SE. HTML 5 включва разпоредби за аудио и видео без добавки. JavaScriptсъщо е вграден в повечето съвременни уеб браузъри и позволява на създателите на уебсайтове да изпращат код до уеб браузъра, който го инструктира как интерактивно да променя съдържанието на страницата и да комуникира с уеб сървъра, ако е необходимо. Вътрешното представяне на съдържанието на браузъра е известно като Document Object Model (DOM).

WebGL (Web Graphics Library) е модерен JavaScript API за изобразяване на интерактивни 3D графики без използването на добавки.

Той позволява интерактивно съдържание като 3D анимации, визуализации и видео обяснения на представените потребители по най-интуитивния начин.

Тенденцията от епохата на 2010 г. в уебсайтовете, наречена „отзивчив дизайн“, даде най-доброто изживяване при гледане, тъй като предоставя оформление, базирано на устройството, за потребителите.

 Тези уебсайтове променят оформлението си според устройството или мобилната платформа, като по този начин предоставят богато потребителско изживяване.

# HTML

## Какво е HTML

HTML (съкращение от термина на английски: HyperText Markup Language, произнасяно най-често като „ейч-ти-ем-ел“, в превод „език за маркиране на хипертекст“) е основният маркиращ език за описание и дизайн на уеб страници. HTML е стандарт в интернет, а неговите стандарти се определят от международния консорциум W3C. Текущата версия на стандарта е HTML 5.0 (от 28 октомври 2014 г.), а предходната стабилна версия е HTML.

Описанието на документа става чрез специални елементи, наречени HTML елементи или техните маркери, които се състоят от тагове и съответстващите етикети (HTML tags) и ъглови скоби (като например елемента <html>). HTML елементите са основната градивна единица на кода, който изграждат уеб страниците. Чрез тях се форматира, графично оформя текста и неговите отделните части в рамките на една уеб страница, като например заглавия, цитати, текстови раздели, хипертекстови препратки и т.н. Най-често HTML елементите са групирани по двойки <h1> и </h1>.

В повечето случаи HTML кодът е написан в текстови редактори, с файлов формат .html, .htm, dhtml и се качва и хоства на сървъри, които са онлайн в интернет или са част от www мрежата. Тези .html файлове съдържат програмно на таговете на HTML и текстово съдържание със маркери и коментари – също инструкции за браузъра, за това какъв точно тип е .html страницата, а също за това как да се показва текстът, особено що се отнася до езиковите характеристики. За да се илюстрира как се включва текст в HTML код: <маркер> Някакъв текст. </край на маркера>. уеб браузърите са програмирани от своя страна така, в повечето случаи, макар че някои браузъри могат да имат съответно проблеми на версията, за да могат да прочетат HTML документите и да ги покажат на екрана като уеб страници. Браузърите не показват самите HTML тагове, освен ако не се отиде в менюто за да се направи това, така че те „интерпретират“ (тоест парсват) съдържанието на страницата като код и текст за да могат след работа на процесора да покажат желаното уеб-съдържание.

## CSS и скриптове

Основното предимство на HTML е, че уеб-страниците, които са го включват в кода си, могат да се разглеждат чрез показването им от браузъра на екрана на повечето устройства. Уебстраницата може да има дизайн, който дори изглежда с добър дизайн с помощта на CSS или „правилно оформен“ (например с помощта на C#), както върху монитора на персоналния компютър, но също и върху миниатюрния дисплей на пейджър или дисплея на мобилен телефон.

HTML може да прикрепя скриптове писани на езици като JavaScript, който е помощен за HTML, и това променя поведението на дадена уеб страница. Cascading Style Sheets (CSS) се използват, като това се прави за да се определя изгледа и оформлението на текста и други включени в страницата изображения и илюстриращи материали. World Wide Web Consortium (W3C) поддържа както HTML, така и CSS, и насърчава използването на CSS в HTML страниците още от 1997. Това допринася за разделяне съдържанието и структурата на уеб страниците от тяхното визуално представяне.

## Редактори

### Текстови редактори

Текстовите (сорсови) редактори, предназначени за употреба с HTML обикновено предоставят възможност за селектиране на синтаксиса. Шаблони, работни ленти и клавиатурни преки пътища често пъти могат да включват стандартни HTML елементи и структури.

Текстовите редактори обикновено включват или вградени функции, или интегрирани външни инструменти за цели като сорс контрол и контрол на версията, проверка на връзки, проверка и валидация на кодове, почистване и форматиране на кодове, ъплоуд чрез FTP или WebDav и проектово структуриране.

Текстовите редактори изискват потребителски познания в областта на HTML и други уеб технологии, които дизайнерът би искал да използва, като CSS, Java и сървърно-ориентирани езици. Тези редактори се наричат също така Прости HTML редактори( A Simple HTML Editor (ASHE)).

Някои обикновени HTML редактори като Windows Notepad могат да съхраняват HTML файлове като използват разширения като .html .htm .css и т.н.

### Обектни редактори

Някои редактори позволяват алтернативно редактиране на сорсовия текст по по-визуално ориентирани способи отколкото обикновеното цветно селектиране с мишката, но не в WYSIWYG режим. Някои WYSIWYG редактори включват опцията да използват палитрени прозорци които позовляват да се редактират текстовите параметри на селектираните обекти. Тези палитри позволяват или редактирането на параметри в полета за всеки параметър, или текстови прозорци, чрез които се редактират пълни групи текст за селектирания обект. Те могат да включват механизми, които предоставят и селектират различни опции при редактирането на параметри. ADOBE Go Live разполага с контурен редактор, чрез който се създават разширяеми и падащи HTML обекти и свойства.

### WYSIWYG HTML редактори

WYSIWYG HTML редакторите предоставят интерфейс за редакция който показва страницата такава, каквато ще бъде изобразена директно в браузъра. Тези едитори могат да бъдат както самостоятелни програми, като Adobe Dreamweaver или Microsoft Frontpage(вече неподдържана), така и под формата на добавки за браузъра, позволяващи директно редактиране. Стиловият енджин трябва да бъде достатъчно развит, за да позволява на разработчиците да пишат, пействат, трият и манипулират съдържанието. Целта е такава, че по всяко време на редактирането видимият резултат трябва да представлява това, което по-късно ще се вижда в стандартния уеб браузър.

# Python

Python е популярен език за програмиране. Създаден е от Guido van Rossum и е издаден през 1991 г.

Използва се за:

* уеб разработка (от страната на сървъра),
* разработване на софтуер,
* математика,
* системен скрипт.
* Приложения на Python

Python е известен със своя характер с общо предназначение, което го прави приложим в почти всяка област на разработка на софтуер. Python прави своето присъствие във всяка нововъзникваща област. Това е най-бързо развиващият се език за програмиране и може да разработи всяко приложение.

Тук посочваме области на приложение, където може да се прилага Python.

* Уеб приложения

Можем да използваме Python за разработване на уеб приложения. Той предоставя библиотеки за обработка на интернет протоколи като HTML и XML, JSON, обработка на имейли, заявка, beautifulSoup, Feedparser и т.н. Една от уеб-рамките на Python, наречена Django, се използва в Instagram . Python предоставя много полезни рамки и те са дадени по-долу:

* Django и Pyramid framework (използвайте за тежки приложения)
* Колба и бутилка (микрорамка)
* Plone и Django CMS (Advance Content management)
* Настолни GUI приложения

GUI означава графичен потребителски интерфейс, който осигурява гладко взаимодействие с всяко приложение. Python предоставя Tk GUI библиотека за разработване на потребителски интерфейс. По-долу са дадени някои популярни GUI библиотеки.

* Tkinter или Tk
* wxWidgetM
* Kivy (използва се за писане на мултитъч приложения)
* PyQt или Pyside
* Конзолно базирано приложение

Базираните на конзолата приложения се изпълняват от командния ред или обвивката. Тези приложения са компютърни програми, които използват команди за изпълнение. Този вид приложение беше по-популярно в старото поколение компютри. Python може да разработи този вид приложение много ефективно. Той е известен с REPL, което означава цикъл за четене-оценка-печат , което го прави най-подходящия език за приложенията на командния ред.

Python предоставя много безплатни библиотеки или модули, които помагат за изграждането на приложения от командния ред. Необходимите IO библиотеки се използват за четене и запис. Помага за анализиране на аргументи и създаване на помощен текст за конзолата извън кутията. Има и усъвършенствани библиотеки, които могат да разработват независими конзолни приложения.

* Разработка на софтуер

Python е полезен за процеса на разработка на софтуер. Той работи като поддържащ език и може да се използва за изграждане на контрол и управление, тестване и т.н.

* SCons се използва за изграждане на контрол.
* Buildbot и Apache Gumps се използват за автоматизирано непрекъснато компилиране и тестване.
* Round или Trac за проследяване на грешки и управление на проекти.
* Научни и числени

Това е ерата на изкуствения интелект, в която машината може да изпълнява задачата като човека. Езикът Python е най-подходящият език за изкуствен интелект или машинно обучение. Състои се от много научни и математически библиотеки, което улеснява решаването на сложни изчисления.Внедряването на алгоритми за машинно обучение изисква сложно математическо изчисление. Python има много библиотеки за научни и числени, като Numpy, Pandas, Scipy, Scikit-learn и др. Ако имате основни познания за Python, трябва да импортирате библиотеки в горната част на кода. По-долу са дадени няколко популярни рамки на машинни библиотеки.

* SciPy
* Scikit-учат
* NumPy
* Panda
* Matplotlib
* Бизнес приложения

Бизнес приложенията се различават от стандартните приложения. Електронната търговия и ERP са пример за бизнес приложение. Този вид приложение изисква широка мащабируемост и четливост, а Python предоставя всички тези функции.

* Аудио или видео базирани приложения

Python е гъвкав за изпълнение на множество задачи и може да се използва за създаване на мултимедийни приложения. Някои мултимедийни приложения, създадени с помощта на Python, са TimPlayer, cplay и т.н. По-долу са дадени няколкото мултимедийни библиотеки.

* Gstreamer
* Пиглет
* QT фонон
* Корпоративни приложения

Python може да се използва за създаване на приложения, които могат да се използват в предприятие или организация. Някои приложения в реално време са OpenERP, Tryton, Picalo и др.

* Приложение за обработка на изображения

Python съдържа много библиотеки, които се използват за работа с изображението. Изображението може да се манипулира според нашите изисквания. По-долу са дадени някои библиотеки за обработка на изображения.

* OpenCV
* Pillow
* SimpleITK

В тази тема сме описали всички типове приложения, при които Python играе съществена роля в разработването на тези приложения. В следващия урок ще научим повече концепции за Python.

## Python frameworks и библиотеки

Технологичните рамки – т.нар. frameworks, са комплексни инструменти, играещи ролята на основа, върху която програмистите могат да надграждат софтуерните си разработки. Те осигуряват множество функционалности наготово и ускоряват процеса на работа.

Две са водещите имена, които ще срещнеш, докато изграждаш уменията си, когато говорим за рамки. На първо място, имаш Django – може би най-популярният Python framework. За по ефективна разработка на по-малки проекти, можеш да разчиташ на по-удачната Flask микро рамка.

От друга страна, библиотеките осигуряват отделни функционалности, с които можеш да разшириш разработката си. Популярен пример за впечатляваща Python библиотека е TensorFlow, безценна в сферите на машинното обучение и deep learning.

Това са все инструменти и технологии, с които ще работиш, ако тръгнеш по пътя на специализация с езика. Но за да стигнеш до този момент, трябва да направиш първите си крачки днес.

## Защо използваме Frameworks

Разработването на софтуер е сложен процес. Това изисква множество задачи, включително кодиране, проектиране и тестване. Само за кодиращата част програмистите трябваше да се погрижат за синтаксиса, декларациите, събирането на боклука, изразите, изключенията и др.

Frameworks улесняват живота на разработчиците, като им позволяват да поемат контрол върху целия процес на разработка на софтуер или по-голямата част от него от една платформа.

Предимства от използването на frameworks:

* Подпомага установяването на по-добри практики за програмиране и подходящо използване на шаблони за проектиране
* Кодът е по-сигурен
* Дублиран и излишен код може да бъде избегнат
* Помага за последователно разработване на код с по-малко грешки
* Улеснява работата по сложни технологии
* Човек може да създаде своя софтуерна рамка или да допринесе за рамки с отворен код. Следователно има непрекъснато подобряване на функционалността
* Няколко кодови сегмента и функционалности са предварително изградени и тествани. Това прави приложенията по-надеждни
* Тестването и отстраняването на грешки в кода е много по-лесно и може да се направи дори от разработчици, които не притежават кода
* Времето, необходимо за разработване на приложение, се намалява значително
* Какво влиза в един framework

Когато инсталирате framework, първото нещо, за което трябва да се погрижите, са системните изисквания. След като framework-a е инсталиран и конфигуриран, той създава структура на директория.

* Разлика между библиотека и рамка

Някои може да приемат, че framework-а е колекция от библиотеки, точно както библиотеките са колекция от предварително компилирани рутинни процедури. Това обаче не е вярно, тъй като не всички frameworks използват или зависят от библиотеки.

Разликата между библиотека и framework е, че последният извиква кода. Обратно на това, кодът извиква софтуерната библиотека.

* Език за програмиране срещу Frameworks

Езикът за програмиране казва на компютъра какво трябва да прави. Всеки език за програмиране разполага със синтаксис и определен набор от правила, които трябва да се спазват всеки път, когато се пише кодът.

Framework-а е изградена върху език за програмиране. Например,

Rails, известен също като Ruby on Rails, е framework, изграден върху езика за програмиране Ruby.

Django и Flask са два различни framework, изградени върху езика за програмиране Python . Следователно те са известни също като  Python frameworkс. React и Angular са front-end frameworks, изградени върху езика за програмиране JavaScript.

## Видове frameworks

Като разработчик трябва да сте нащрек за рамки, които най-добре отговарят на вашите нужди. Независимо дали става въпрос за работа върху уебсайт, наука за данни, управление на бази данни или мобилни приложения, съществуват софтуерни рамки за всички жанрове софтуерно програмиране.

Има много видове софтуерни рамки, които улесняват разработването на приложения за широк спектър от области за разработка на приложения. Нека се потопим в някои от софтуерните рамки, които са на мода днес:

## Framework за уеб приложения

* + Angular

Angular е базирана на машинописи JS рамка с отворен код, която улеснява изграждането на приложения в мрежата. Angular решава предизвикателствата при разработката на приложения чрез комбиниране на декларативни шаблони, инжектиране на зависимости, инструменти от край до край и много други.

Angular дава възможност на разработчиците да създават приложения, които живеят в мрежата, мобилни устройства и настолни компютри.

Популярната JavaScript рамка се използва в публични приложения и сайтове като Google Cloud Platform и AdWords, както и много вътрешни инструменти на Google.

Някои популярни уебсайтове, разработени с помощта на AngularJS, са Нетфликс, Paypal, Upwork, Youtube, Джанго

Django е безплатна рамка за уеб приложения с отворен код, написана на Python. Създаден от екип от опитни разработчици, Django се грижи за уеб разработката, така че разработчиците да могат да се съсредоточат върху писането на приложения, без да преоткриват колелото.

Големите организации активно използват Django в своето развитие. Някои популярни уебсайтове, разработени с помощта на Django, са Disqus, Instagram, Mozilla, Pinterest

## Използване на Python

Python е език за програмиране на високо ниво с общо предназначение, с отворен код и също така предоставя редица библиотеки и рамки. Python придоби популярност поради своята простота, лесен синтаксис и удобна за потребителя среда. Използването на Python, както следва.

* Настолни приложения
* Уеб приложения
* Наука за данни
* Изкуствен интелект
* Машинно обучение
* Научни изчисления
* роботика
* Интернет на нещата (IoT)
* Игри
* Мобилни приложения
* Анализ и предварителна обработка на данни

В следващата тема ще обсъдим приложението Python , където сме дефинирали подробно използването на Python.

# Django framework (Python)

Django е изключително популярна и пълнофункционална уеб рамка от страна на сървъра, написана на Python. Този модул ви показва защо Django е една от най-популярните рамки за уеб сървъри, как да настроите среда за разработка и как да започнете да я използвате, за да създавате свои собствени уеб приложения.

* + Предпоставки

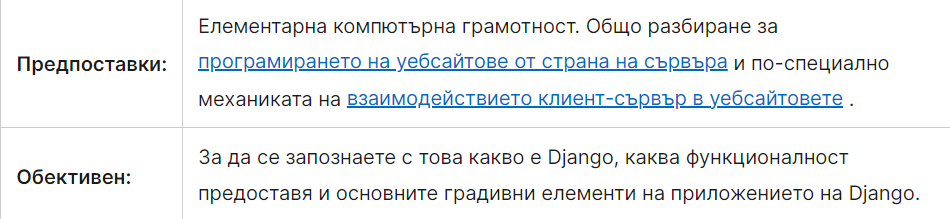
Преди да започнете този модул, не е необходимо да имате никакви познания по Django. В идеалния случай ще трябва да разберете какво представлява уеб програмирането и уеб рамките от страна на сървъра, като прочетете темите в нашия модул за първи стъпки в програмирането на уеб сайт от страна на сървъра .

Препоръчват се общи познания за концепциите за програмиране и Python , но не са от съществено значение за разбирането на основните концепции.

* + Какво е Django?

В тази първа статия за Django отговаряме на въпроса "Какво е Django?" и ще ви даде общ преглед на това, което прави тази уеб рамка специална.

Ще очертаем основните функции, включително някои от разширените функции, които няма да имаме време да разгледаме подробно в този модул. Ще ви покажем и някои от основните градивни елементи на Django приложение (въпреки че на този етап все още нямате среда за разработка, в която да го тествате).



Django е Python уеб рамка от високо ниво, която позволява бързо разработване на сигурни и поддържаеми уебсайтове. Създаден от опитни разработчици, Django се грижи за голяма част от неприятностите на уеб разработката, така че можете да се съсредоточите върху писането на вашето приложение, без да е необходимо да преоткривате колелото. Той е безплатен и с отворен код, има процъфтяваща и активна общност, страхотна документация и много опции за безплатна и платена поддръжка.

## Използване на Django

Django ви помага да пишете софтуер, който е:

* + Завършен

Django следва философията „Включени батерии“ и предоставя почти всичко, което разработчиците биха искали да направят „извън кутията“. Тъй като всичко, от което се нуждаете, е част от един „продукт“, всичко работи безпроблемно заедно, следва последователни принципи на проектиране и има обширна и актуална документация .

* + Разнообразен

Django може да се използва (и е бил) използван за изграждане на почти всякакъв вид уебсайтове - от системи за управление на съдържанието и wiki, до социални мрежи и новинарски сайтове. Може да работи с всяка клиентска рамка и може да доставя съдържание в почти всеки формат (включително HTML, RSS емисии, JSON и XML).

Вътрешно, въпреки че предоставя избор за почти всяка функционалност, която бихте искали (напр. няколко популярни бази данни, модули за шаблони и т.н.), той може също да бъде разширен, за да използва други компоненти, ако е необходимо.

* + Сигурен

Django помага на разработчиците да избегнат много често срещани грешки в сигурността, като предоставя рамка, която е проектирана да "прави правилните неща", за да защити уебсайта автоматично. Например Django предоставя сигурен начин за управление на потребителски акаунти и пароли, като избягва често срещани грешки като поставяне на информация за сесията в бисквитки, където е уязвима (вместо това бисквитките просто съдържат ключ и действителните данни се съхраняват в базата данни) или директно съхраняване на пароли а не хеш парола.

Хешът на паролата е стойност с фиксирана дължина, създадена чрез изпращане на паролата чрез криптографска хеш функция . Django може да провери дали въведената парола е правилна, като я пусне през хеш функцията и сравни изхода със съхранената хеш стойност. Въпреки това, поради „еднопосочния“ характер на функцията, дори ако съхранена хеш стойност е компрометирана, за атакуващия е трудно да разработи оригиналната парола.

Django позволява защита срещу много уязвимости по подразбиране, включително SQL инжектиране, междусайтови скриптове, фалшифициране на междусайтови заявки и щракане (вижте Защита на уебсайтове за повече подробности за подобни атаки).

* + Мащабируеми

Django използва базирана на компоненти „ споделено нищо “ архитектура (всяка част от архитектурата е независима от другите и следователно може да бъде заменена или променена, ако е необходимо). Наличието на ясно разделение между различните части означава, че може да се мащабира за увеличен трафик чрез добавяне на хардуер на всяко ниво: кеширащи сървъри, сървъри на бази данни или сървъри на приложения. Някои от най-натоварените сайтове успешно мащабираха Django, за да отговорят на техните изисквания (напр. Instagram и Disqus, за да назовем само два).

* + Поддържа се

Кодът на Django е написан с помощта на принципи и модели на дизайн, които насърчават създаването на поддържаем и многократно използваем код. По-специално, той използва принципа Не се повтаряй (DRY), така че няма ненужно дублиране, намалявайки количеството код. Django също насърчава групирането на свързана функционалност в „приложения“ за многократна употреба и на по-ниско ниво групира свързания код в модули (по линията на модела на контролера за изглед на модел (MVC)  .

* + Преносим

Django е написан на Python, който работи на много платформи. Това означава, че не сте обвързани с никоя конкретна сървърна платформа и можете да стартирате приложенията си на много версии на Linux, Windows и macOS.

 Освен това Django се поддържа добре от много доставчици на уеб хостинг, които често предоставят специфична инфраструктура и документация за хостване на сайтове на Django.

## История

Първоначално Django е разработен между 2003 и 2005 г. от уеб екип, който отговаря за създаването и поддържането на уебсайтове на вестници. След като създаде редица сайтове, екипът започна да отделя и използва повторно много общи кодове и модели на дизайн. Този общ код еволюира в обща рамка за уеб разработка, която беше отворена като проект "Django" през юли 2005 г.

Django продължи да расте и да се подобрява от първото си важно издание (1.0) през септември 2008 г. до наскоро издадената версия 4.0 (2022 г.).

Всяко издание добавя нова функционалност и корекции на грешки, вариращи от поддръжка за нови типове бази данни, механизми за шаблони и кеширане, до добавянето на „общи“ функции и класове за изглед (които намаляват количеството код, за който разработчиците трябва да пишат редица програмни задачи).

Django сега е процъфтяващ, съвместен проект с отворен код, с много хиляди потребители и сътрудници. Въпреки че все още има някои функции, които отразяват неговия произход, Django се превърна в гъвкава рамка, която е в състояние да разработи всякакъв тип уебсайт.

## Колко популярен е Django

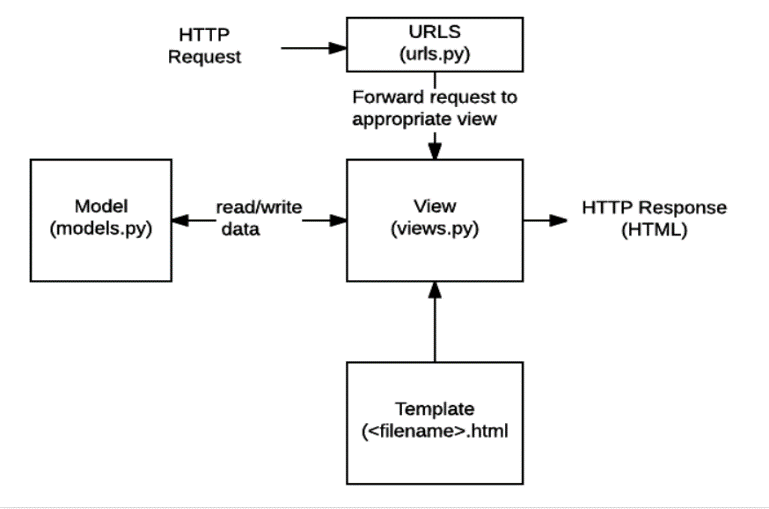
Няма никакво леснодостъпно и окончателно измерване на популярността на рамки от страна на сървъра (въпреки че можете да оцените популярността, като използвате механизми като преброяване на броя проекти на GitHub и въпроси на StackOverflow за всяка платформа). По-добрият въпрос е дали Django е „достатъчно популярен“, за да избегне проблемите на непопулярните платформи. Продължава ли да се развива? Можете ли да получите помощ, ако имате нужда от нея? Има ли възможност да получите платена работа, ако научите Django?

Във основа на броя сайтове, които използват Django, броя на хората, които допринасят за кодовата база и броя на хората, предоставящи както безплатна, така и платена поддръжка, тогава да, Django е популярен framework!

Сайтове, които използват Django, включват: Disqus, Instagram, Knight Foundation, MacArthur Foundation, Mozilla, National Geographic, Open Knowledge Foundation, Pinterest и Open Stack (източник: страница с преглед на Django).

## Как изглежда кодът на Django

В традиционен уебсайт, управляван от данни, уеб приложение чака HTTP заявки от уеб браузъра (или друг клиент). Когато се получи заявка, приложението определя какво е необходимо въз основа на URL адреса и евентуално информация в POSTданни или GETданни. В зависимост от това какво се изисква, след това може да чете или записва информация от база данни или да изпълнява други задачи, необходими за удовлетворяване на заявката. След това приложението ще върне отговор на уеб браузъра, като често динамично създава HTML страница, която браузърът да показва чрез вмъкване на извлечените данни в контейнери в HTML шаблон. Уеб приложенията на Django обикновено групират кода, който обработва всяка от тези стъпки, в отделни файлове:

* URL адреси: Въпреки че е възможно да се обработват заявки от всеки отделен URL чрез една функция, много по-лесно е да се напише отделна функция за преглед, която да обработва всеки ресурс. URL картографът се използва за пренасочване на HTTP заявки към подходящия изглед въз основа на URL адреса на заявката. URL картографът може също така да съпостави определени модели от низове или цифри, които се появяват в URL, и да ги предаде на функция за преглед като данни.
* Изглед: Изгледът е функция за обработка на заявки, която получава HTTP заявки и връща HTTP отговори. Изгледите имат достъп до данните, необходими за задоволяване на заявки чрез модели , и делегират форматирането на отговора на шаблони .
* Модели: Моделите са обекти на Python, които дефинират структурата на данните на приложението и предоставят механизми за управление (добавяне, модифициране, изтриване) и заявки за записи в базата данни.
* Шаблони: Шаблонът е текстов файл, определящ структурата или оформлението на файл (като HTML страница), с контейнери, използвани за представяне на действително съдържание. Един изглед може динамично да създаде HTML страница с помощта на HTML шаблон, като я попълни с данни от модел . Шаблонът може да се използва за дефиниране на структурата на всеки тип файл; не е задължително да е HTML!
* Секциите по-долу ще ви дадат представа как изглеждат тези основни части на приложението на Django (ще разгледаме повече подробности по-късно в курса, след като настроим среда за разработка).
* Django е рамка на Python, която улеснява създаването на уеб сайтове с помощта на Python.

Django се грижи за трудните неща, за да можете да се концентрирате върху изграждането на вашите уеб приложения.

Django набляга на многократната употреба на компоненти, наричана още DRY (не повтаряйте себе си) и се предлага с готови за използване функции като система за влизане, връзка с база данни и CRUD операции (Създаване, четене, актуализация, изтриване).

* Django е особено полезно за уебсайтове, управлявани от бази данни

Препоръчват се общи познания за концепциите за програмиране и Python , но не са от съществено значение за разбирането на основните концепции.

Забележка: Python е един от най-лесните за начинаещи езици за програмиране и разбиране. Въпреки това, ако искате да разберете по-добре този модул, в интернет има множество безплатни книги и уроци, които да ви помогнат (новите програмисти може да искат да разгледат страницата Python за непрограмисти в wiki на python.org).

# PyCharm

PyCharm е интегрирана среда за разработка (IDE), използвана за програмиране в Python . Той осигурява анализ на кода, графичен дебъгер, интегриран модулен тестер, интеграция със системи за контрол на версиите и поддържа уеб разработка с Django . PyCharm е разработен от чешката компания JetBrains .

Той е междуплатформен , работи на Microsoft Windows , macOS и Linux . PyCharm има професионално издание, издадено под патентован лиценз и издание на общността, пуснато под лиценза на Apache .  PyCharm Community Edition е по-малко обширно от Professional Edition.

Предлага се като приложение за различни платформи. Разположен грациозно сред най-добрите IDE на Python, PyCharm осигурява поддръжка за двете версии на Python 2 (2.7) и Python 3 (3.5 и по-нови).

PyCharm идва с множество модули, пакети и инструменти за ускоряване на разработката на Python, като същевременно намалява усилията, необходими за извършване на същото до голяма степен, едновременно. Освен това PyCharm може да бъде персонализиран според изискванията за разработка и личните предпочитания. Той беше пуснат за обществеността за първи път през февруари 2010 г. В допълнение към предлагането на анализ на кода, PyCharm включва:

* Графичен дебъгер
* Интегриран тестер на устройства
* Поддръжка на интеграция за системи за контрол на версиите (VCS)
* Поддръжка за наука за данни с Anaconda

## За какво се използва PyCharm

Основната причина Pycharm за създаването на тази IDE беше програмирането на Python и работата на множество платформи като Windows, Linux и macOS. IDE включва инструменти за анализ на код, програма за отстраняване на грешки, инструменти за тестване, както и опции за контрол на версиите. Той също така подпомага разработчиците при изграждането на плъгини на Python с помощта на различни налични API. IDE ни позволява да работим директно с няколко бази данни, без да я интегрираме с други инструменти. Въпреки че е специално проектиран за Python, HTML, CSS и Javascript файлове също могат да бъдат създадени с тази IDE. Той също така идва с красив потребителски интерфейс, който може да бъде персонализиран според нуждите с помощта на добавки.

PyCharm се предлага в три варианта:

* Безплатна версия, наречена The Community Edition, достъпна под лиценза на Apache.
* Търговска версия, обозначена като Professional Edition, достъпна под лиценз за собственост.
* Безплатна за използване образователна версия, наречена Edu Edition, насочена към студенти и професионалисти, които се интересуват от изучаването на Python , достъпна под лиценза на Apache.

## Характеристики на PyCharm

* Интелигентен редактор на код

PyCharm идва с интелигентен редактор на код, който улеснява писането на висококачествен Python код. Той предлага подобрено ниво на разбиране и четливост на кода посредством различни цветови схеми за ключови думи, класове и функции, т.е. синтаксис и подчертаване на грешки.

В допълнение към предлагането на функцията за интелигентно завършване на код, редакторът на код генерира инструкции за попълване на текущия код.

Идентифицирането на грешки и проблеми е много по-удобно, заедно с интегрирането на Linter и бързите корекции.

* + Наличие на инструменти за интегриране

PyCharm осигурява поддръжка за интегриране на набор от инструменти. Тези инструменти варират от подпомагане за подобряване на производителността на кода до улесняване на работата с проекти за наука за данни. Някои от най-важните инструменти за интеграция, налични за PyCharm, включват:

* Anaconda – Безплатна дистрибуция на Python с отворен код, насочена към научни изчисления с опростено управление и внедряване на пакети.
* IPython - Стабилна командна обвивка за интерактивни изчисления.
* Kite - плъгин за автоматично довършване, задвижван от AI.
* Pylint - Изходен код, програма за проверка на грешки и качество.
* pytest - рамка за писане на малки тестове за Python код.
* WakaTime - табло за разработчици с показатели за производителност и автоматично проследяване на времето
  + Наука за данни и машинно обучение

PyCharm идва с поддръжка за научни библиотеки, като Matplotlib и SciPy, за да помогне на разработчиците на Python да реализират проекти за наука за данни и машинно обучение.

* + Интегрирано отстраняване на грешки и тестване

IDE идва с поддръжка за отстраняване на грешки и тестване на програми. За да постигне същото, PyCharm разполага с интегриран дебъгер на Python и интегрирано тестване на единици с кодово покритие ред по ред.

* + Мултитехнологична разработка

Разработчиците на Python могат също да използват PyCharm за създаване на уеб приложения. Като такава, Python IDE осигурява поддръжка за популярни уеб технологии, включително CoffeeScript, CSS, HTML, JavaScript, TypeScript. Освен това включва поддръжка за Cython, шаблонни езици и SQL.

Редактирането на живо също е налично в PyCharm, т.е. разработчиците могат да създават/променят уеб страница, докато я пускат на живо едновременно. Следователно промените могат да бъдат проследени директно в уеб браузър. Изграждането на уеб приложения с помощта на AngularJS или NodeJS също е налично.

* + Навигация по проекти и кодове

Функцията за навигация в кода улеснява много разработчиците да навигират до клас, функция или файл. Той също така помага за значително намаляване на усилията и времето, необходими за редактиране и подобряване на кода на Python. Изгледи на файлова структура и специализирани изгледи на проекти са лесно достъпни.

Режимът на обектива позволява на разработчика да инспектира и отстранява напълно целия изходен код на Python.

 С кодова навигация, намирането на елемент, променлива и т.н. се извършва почти за нула време. Разработчиците могат бързо да прескачат между класове, файлове и методи.

* + Рефакторинг

Функцията за рефакторинг в PyCharm помага за подобряване на вътрешната структура на програма на Python, без да засяга външната производителност на същата. Правенето на промени в локални и глобални променливи е ефективно и бързо.

Методът за извличане е там и за разделяне на разширени класове и функции.

PyCharm позволява стартиране, отстраняване на грешки, тестване и внедряване на приложения на отдалечени хостове или виртуални машини. За целта Python IDE предлагат.

## Какво е IDE

PyCharm е изключително популярна Python IDE. Интегрирана среда за разработка или IDE включва редактор на код и компилатор за писане и компилиране на програми на един или много езици за програмиране.

Освен това IDE идва с изобилие от функции, които улесняват цялостното разработване на софтуер. Тъй като IDE разпределя различни цветове на различни програмни обекти, обикновено известни като подчертаване на синтаксис, тя става по-достъпна за:

* Правете разлика между различните програмни единици, като клас и функция, и ги забелязвайте.
* Потърсете грешните ключови думи.
* Прочетете и разберете кода.

Повечето IDE включват функция за автоматично попълване, която създава предложения при писане на код. Това прави писането на код по-ефективно, бързо и по-малко склонно към грешки и правописни грешки. Други стандартни функции, предлагани от модерна IDE са:

* Прозорец за редактор на проекти за ефективно управление и организиране на файлове, необходими за програма/проект.
* Проверка на изхода на кода, написан чрез изходния прозорец
* Предложения за разрешаване на грешки и предупреждения
* Гама от модули и пакети, лесно достъпни на едно място

# Реализация

## Анализ на заданието

Заданието за дипломния проект предполага, че група художници от едно художествено ателие имат идея да подпомогнат първите стъпки в рисуването на начинаещи художници и, за целта, искат да създадат сайт, който предлага серия от нагледни уроци по рисуване за начинаещи.

Сайтът трябва да се състои от 4 различн страници - „Начало“, „За нас“, „Уроци“ и „Контакти“. Секцията „Уроци“ трябва да включва подменю с отделните уроци. Отделните уроци трябва да следват общ шаблон – серия от последователни стъпки, описващи даден процес, техника или тема. Всяка стъпка в даден урок трябва да е представена по формата на изображение и кратко описание към него.

Сайтът трябва да може да се администрира с помощта на администраторско меню, което да позволи добавяне на нови уроци или промяна на някоя от другите страници.

Дизайнът на сайта трябва да е адаптивен и да изглежда еднакво добре на различни устройства.

Заданието, според мен, предполага да се изработи един красиво оформен динамиен сайт. Като основа за разработката избрах езика Python, който има много предимства - от лекотата на работа до голям брой frameworks и библиотеки, опростяващи процеса на разработка. Python често се използва за разработка на бекенд уеб приложения. И за самия frameworks избрах Django защото Python и Django заедно са една от най-популярните технологии за бързо разработване на сигурни и поддържаеми уебсайтове. Създаден от опитни разработчици, Django се грижи за голяма част от неприятностите на уеб разработката, така че можете да се съсредоточите върху писането на вашето приложение, без да е необходимо да преоткривате колелото. А самия код на сайта е изработен на PyCharm

## Структура

Както е известно, всеки сайт на Django представялва трислойно приложение, което се състои от слой за данни, слой за бизнес-логиката и интефейсен слой.

### Модели

Слоят за данни в Django е представен от файла models.py на приложението, който представя моделите които го свързват с базата данни. В конкретният случай съм избрала БД MySQL. Всеки модел в този файл е клас, който описва съовтетна таблица в БД. Например, моделът Picture описва структурата на таблица, която ще се съхранват отделните рисунки за галерията и уроците.

# списък рисунки  
class Picture(models.Model):  
 title = models.CharField('Заглавие', max\_length=200)  
 order = models.SmallIntegerField('Номер', default=1, blank=True, null=False)  
 picture = models.ImageField('Снимка/графика:', upload\_to='article\_pics')  
 author = models.CharField('Автор', max\_length=60)  
  
 def \_\_str\_\_(self):  
 return self.title  
  
 class Meta:  
 verbose\_name = 'Рисунка'  
 verbose\_name\_plural = 'Рисунки'

Общо в моя случай са създадени следните модели:

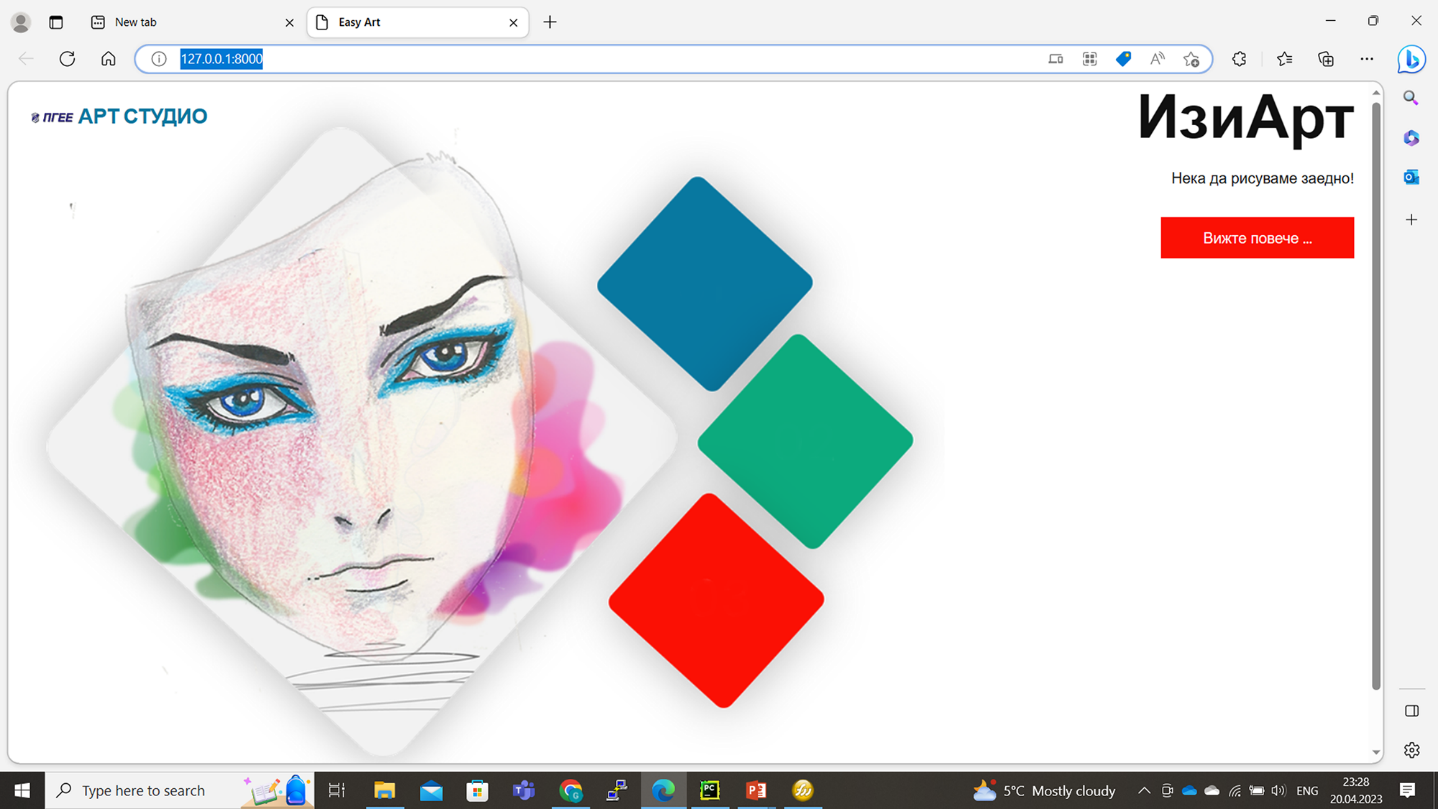
* Picture - за рисунките
* Lesson – за уроците
* Item – за отделните стъпки на всеки урок

### Изгледи

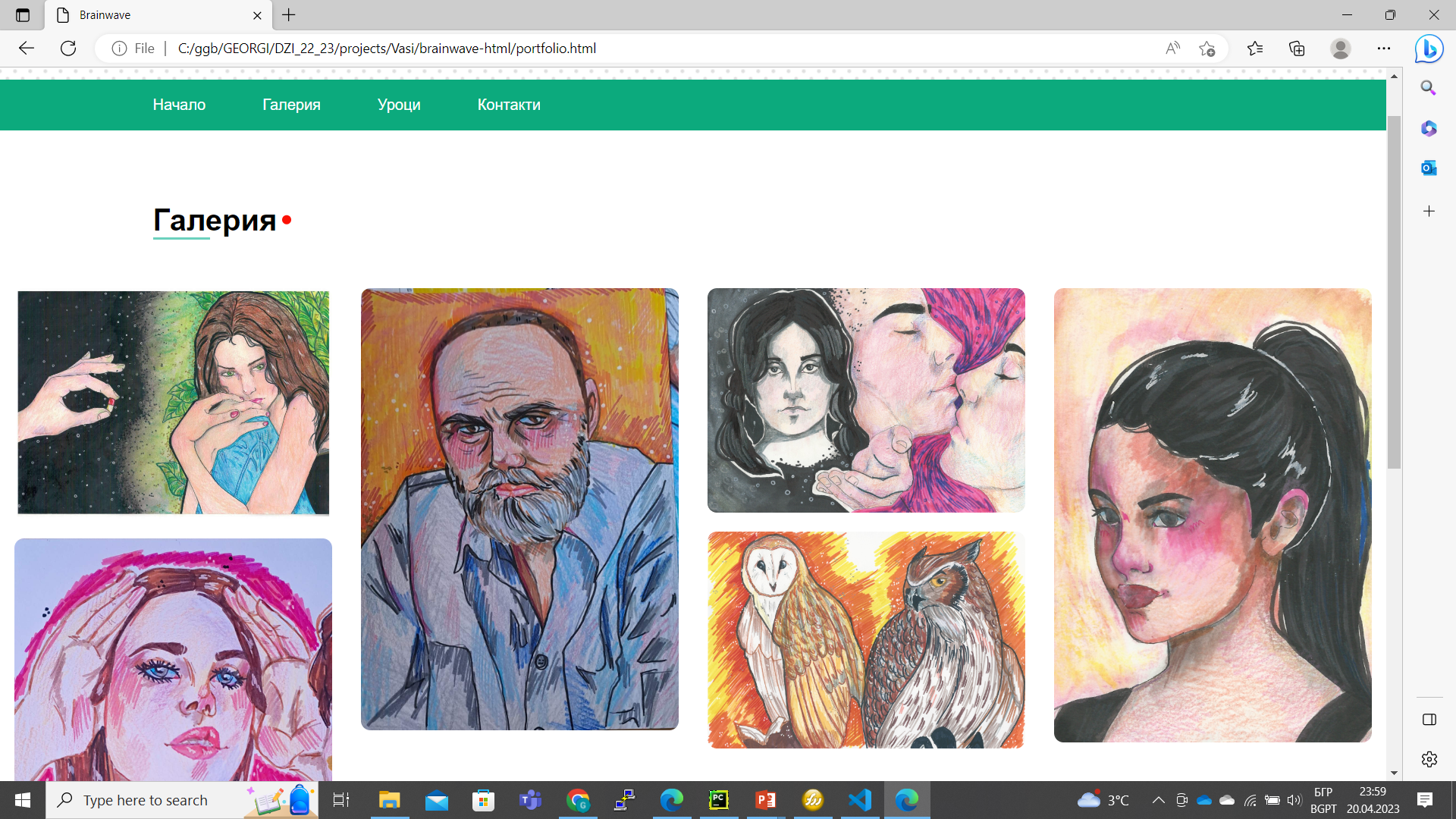
В Django изгледите (views) реализират бизнес-логиката на приложението. Представени са в едноименния файл. Всяка уеб-страница в рамките на сайта е реализирана от отделен изглед. Например, изгледът index създава началната страница на приложението:

# Начална страница.  
**def** index(request):  
 **return** render(request, **'main/index.html'**)

В конкретният изглед не са използват данни от БД, но когато са неоходими – те се извличат посредством съответния модел. Изгледът „извиква“ определн HTML шаблон, в който се вмъкват сответните динамични данни и той се подава на клиента за визуализиране. Изгледът index, визулизиран в браузър изглежда така:



Друг изглед – gallery, формира страницата „За нас“ която представлява галерия (слайд шоу) от всички рисунки от таблицата Picture в БД. За клиента това изглежда така:



### Заявки

Заявките са визуализиране на определна страница Django подава към файла urls.py, който според полученият URL определя кой изглед трябва да се активира:

urlpatterns = [  
 path(**''**, views.index, name=**'home'**),  
 path(**'menu'**, views.menu, name=**'menu\_page'**),  
 path(**'lesson'**, views.lesson(), name=**'lesson\_page'**),  
 path(**'gallery'**, views.gallery, name=**'gallery\_page'**),  
 path(**'contact'**, views.contact, name=**'contact\_page'**),  
]

# Заключение

Самият сайт на художествено ателие, подпомагащо начинаещи артисти е за всеки начинаещ художник който винаги е искал да се потопи в чисуването, но не знае от къде да започне или просто за всеки ценител на доброто изкуство. Сайта ни предлага лесни за използване итерактивни уроци за начинаещи и пълната ни динамична галерия, каято използва база дани с красиви рисунки лично нарисувани от мен самата.

# Списък на използваната литература

1. https://www.javatpoint.com/
2. https://www.datacamp.com/
3. https://developer.mozilla.org/
4. https://www.tutorialspoint.com/
5. https://www.techopedia.com/
6. https://www.w3schools.com/
7. https://hackr.io/blog/what-is-frameworks
8. https://bg.wikipedia.org/wiki/HTML
9. https://intellipaat.com/
10. https://www.jump.bg/blog/
11. https://www.itrobes.com/what-is-dynamic-website/