|  |  |
| --- | --- |
| D:\UKTC\UKTC tempatets\Logo\UKTC-logo-color.png | **НАЦИОНАЛНА ПРОФЕСИОНАЛНА ГИМНАЗИЯ ПО КОМПЮТЪРНИ ТЕХНОЛОГИИ И СИСТЕМИ**  **ГР.ПРАВЕЦ ПРИ ТЕХНИЧЕСКИ УНИВЕРСИТЕТ – СОФИЯ** |

**ДИПЛОМНА РАБОТА**

ДЪРЖАВЕН ИЗПИТ ЗА ПРИДОБИВАНЕ НА ТРЕТА СТЕПЕН НА ПРОФЕСИОНАЛНА КВАЛИФИКАЦИЯ – ЧАСТ ПО ТЕОРИЯ НА ПРОФЕСИЯТА

на

Ученик: Димо Димитров Димитров

ученик/чка от 18112 (XII) клас

Тема: **Уеб сайт за представяне на статистики за криптовалути**

**ПРОФЕСИОНАЛНО НАПРАВЛЕНИЕ:** 481 КОМПЮТЪРНИ НАУКИ

**ПРОФЕСИЯ:** 481020 СИСТЕМЕН ПРОГРАМИСТ

**СПЕЦИАЛНОСТ:** 4810201 СИСТЕМНО ПРОГРАМИРАНЕ

Ученик: ………………….. Ръководител: ………………………

*Ивелин Михайлов*

Правец, 2022

Съдържание

[Увод 3](#_Toc132920634)

[Глава 1: Литературен обзор 4](#_Toc132920635)

[1.1 Среди за разработка на уеб сайт 4](#_Toc132920636)

[1.1.1 Intellij IDEA 4](#_Toc132920637)

[1.1.2 Eclipse 4](#_Toc132920638)

[1.1.4 Visual Studio Code 5](#_Toc132920639)

[1.2 Методи за разработка на уеб сайт 6](#_Toc132920640)

[1.2.1 Планиране 6](#_Toc132920641)

[1.2.2 Стратегия 6](#_Toc132920642)

[1.2.3 Концепция 7](#_Toc132920643)

[1.3 Основни етапи при изграждане на уеб сайт 7](#_Toc132920644)

[1.3.1 Избор на име, домейн и ключови думи за сайта 7](#_Toc132920645)

[1.4 Предимства и недостатъци при разработка на уеб сайта 8](#_Toc132920646)

[1.4.1 Предимства 8](#_Toc132920647)

[Използвани технологии и работни рамки 9](#_Toc132920648)

[2.1 HTML 9](#_Toc132920649)

[2.1.1 Предимства на HTML 10](#_Toc132920650)

[2.1.2 Недостатъци на HTML 10](#_Toc132920651)

[2.2 CSS 11](#_Toc132920652)

[2.2.1 Предимства на CSS 11](#_Toc132920653)

[2.2.2 Недостатъци на CSS 12](#_Toc132920654)

[2.3 PHP 12](#_Toc132920655)

[13](#_Toc132920656)

[2.3.1 Предимства на PHP 13](#_Toc132920657)

[2.3.2 Недостатъци на PHP 13](#_Toc132920658)

[Глава 3: Инженерно решение на поставената задача 19](#_Toc132920659)

[Глава 4: Анализ на получените резултати, приложимост и изводи 32](#_Toc132920660)

[Заключение 38](#_Toc132920661)

[Използвана литература 39](#_Toc132920662)

[Списък с използваните означения и съкращения 40](#_Toc132920663)

# 

# Увод Настоящата дипломна работа има за цел разработка на уебсайт, който служи за представяне на статистика за най-популярните крипто валути със средствата на езика за програмиране Java Script и по-точно библиотеката му – React.js.

Уебсайтът запазва и маркираните от вас крипто валути за да имате по-лесен достъп до тях. Цялата система е направена с помощта на VisualStudioCode. Системата ползва База Данни чрез MySql-поддръжка.

Налични са всички стандартни функции, които се изискват за такъв тип приложение, а именно :

1. Регистрация на потребители;
2. Log-in форма за вписване на потребителя чрез потребителско име и парола;
3. Възможност за маркирането на крипто валути като „Любими“;
4. Вписване като обикновен потребител или администратор от което зависи и нивото на достъп и вида на операции, които могат да се изпълняват;

След влизането в профила си, данните потребителя ще бъдат запомняни за известно време, което позволява за по-лесен достъп.

Интерфейсът умишлено е минималистки и опростен за да не затруднява работата на потребителя.

# Глава 1: Литературен обзор

## 1.1 Среди за разработка на уеб сайт

Средите за разработка са много, но тези, които се отличават най-много от останалите са:

1. Intellij IDEA
2. Eclipse
3. PyCharm
4. Visual Studio Code

### 1.1.1 Intellij IDEA

**IntelliJ IDEA** е интелигентна среда за разработка. Обикновенно се използва за писане на **Java** код. Тя обаче потдържа по-голямото множество от популярните езици за разработване на уеб приложения и уеб сайтове. **IntelliJ IDEA** е бърза, гъвкава, приятна, лесна за работа и предоставя възможноста за смяна на настройките и цветовата си гама. Тя е годна за търговски разработки.

### 1.1.2 Eclipse

**Eclipse** е многоезична среда за разработване на софтуер. Състои се от интегрирана среда за разработка(IDE) и плъгин система. Използва се главно за писане на **Java** код. Плъгн системата позволява писане на повечето популярни езици за програмиране. Тя е предпочитана за разработката на логическата част от уеб приложенията и използването й за създаването на уеб сайтове е рядкост.

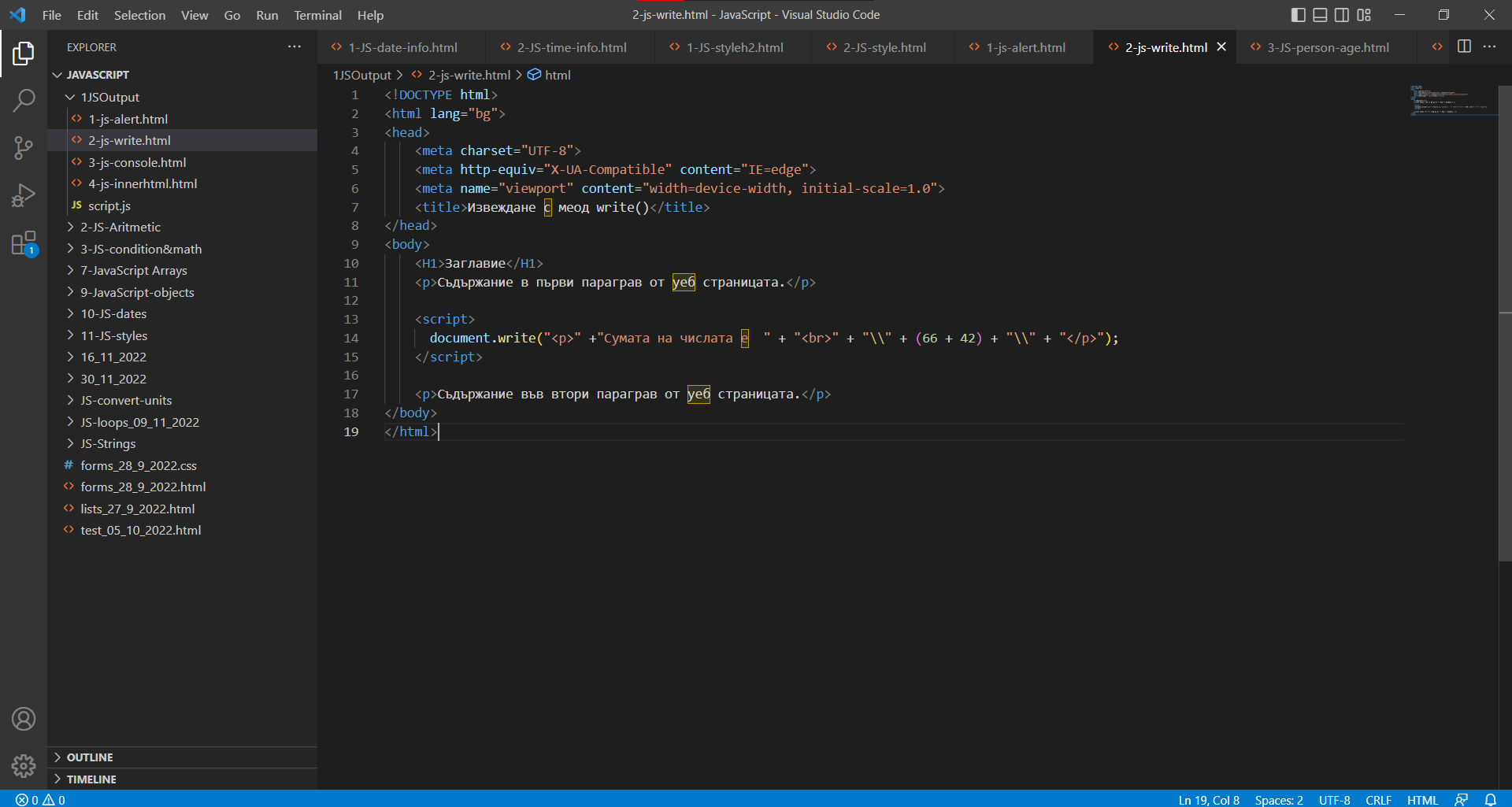
С **Eclipse** се работи по-сложно от колкото с **IntelliJ IDEA**, което я прави неподходяща за разработчици с малко опит и за екипи, работещи по проекти с множество технологии.

1.1.3 PyCharm

**PyCharm IDE** е среда за разработка за уеб приложения и уеб сайтове. Тя е изградена пряко за работа с **Python, JavaScript**. Една от главните й сили за създаване на уеб сайтове е рамката **Django**. Тя позволява и писането на **HTML** и **CSS** в средата за разработка.

## 1.1.4 Visual Studio Code

Visual Studio Code е интегрирана среда за разработка (**IDE**) за всякакъв вид програмни езици от високо ниво. Предназначен е да улесни максимално рабоботата при разработка на приложения. За всеки език са налични автоматични допълнения, които се изтеглят отделно. В средата за разработка могат да се отворят текстови документи, снимки с различни разширения и файлове на потдържаните езици. Има и терминал, който доставя улеснен достъп, не само до проекта над който работим, но и до цялата система. Visual Studio Code може да се използва на Windows, Mac и Linux. Безплатен е и лесен за инсталиране.



**Фигура 6.1**

Примерен изглед на **Visual Studio Code**

***Потдържани езици::***

***Плъгините***, част от Visual Studio Code и тезки, които можем да си изтеглим допълнително потдържат следните главни езици и много други:

* + - * 1. C++
        2. C#
        3. CSS
        4. Dart
        5. Dockerfile
        6. F#
        7. Go
        8. HTML
        9. Java
        10. JavaScript
        11. JSON
        12. Julia
        13. Less
        14. Markdown
        15. PHP
        16. PowerShell
        17. Python
        18. R
        19. Rust
        20. SCSS
        21. T-SQL
        22. TypeScript.

## 1.2 Методи за разработка на уеб сайт

1. Планиране
2. Стратегия
3. Концепция

### 1.2.1 Планиране

Планирането е етапа в който изготвяме базов план на дейностите, които трябва да се извършат по проекта. Разглеждат се всички налични ресурси и се обмислят неравностите, които може да възникнат в екипа.

### 1.2.2 Стратегия

Започваме етапът на стратегия, като анализираме информацията, получена от клиента. Очертаваме основите, целта, времето и ресурсите с които рзполагаме. Разделяме екипите от разработчици и рапределяме типовете задачи.

### 1.2.3 Концепция

Концепцията остава една и съща. Докато разработката тече, дейноста на всички работещи се следи. Софтуера се тества и презентира редовно. С течение на времето, проекта е завършен.

## 1.3 Основни етапи при изграждане на уеб сайт

1. Проучване
2. Анализ
3. Планиране

Преди да се започне работа по проекта, се оточнява всеки детайл от него с клента. Това са: дизайн, насоченост, име на домейн, цветове, функционалност, методи за удостоверяване на потребителите, технологии за изграждане и други.

### 1.3.1 Избор на име, домейн и ключови думи за сайта

Домейнът е едно от най-важните неща характеризиращи добрите и успешни сайтове. Той трябва да се състои от ключова дума, която да показва същността на сайта. Той трябва да е къс и лесен за запомнчне за да се използва по-често от потребителите и да го запомнят без да осъзнаят.

Името на уеб сайта също е от голямо значение. То трябва да е лесно за запомняне, да сочи към главната функционалност на сайта и да се запомня лесно. Това води до по-голям поток от потребители.

Ключовите думи трябва да са съобразени с дейността на сайта. Те са важни при сърфирането в интернет. С по-точни и използвани ключови думи, уеб сайта ще получава по0голям приток на нови потребители.

1.3.2 Софтуерно решение

В този етап се избира софтуерната платформа върху която ще се изгради продукта. Това ще бъде решено, като се вземат предвид нуждите на клиентите и целта на продукта. Тук се изгражда и картата на сайта, по която ще се изгради скелета му.

1.3.3 Дизайн

Дизайнът се определя от целите на продукта, който изграждаме. Цветовата палитра се избира по бизнеса с който ще се занимаваме. Дизайнът винаги трябва да е модерен, динамичен и лесен за ползване. По този начин потребителите се задържат по-дълго време и популяризират продуктите ни повече.

1.3.5 Тестване

По време на изграждането на уеб сайта сме длъжни да имаме постоянно тестване. Така работата върви леко, гладко и няма недоволни разработчици. Тази техника предотвратява големите сривове и губенето на данни. По този начин можем да предоставяме готови части от продукта на поръчителя му.

1.3.6 Валидиране на кода

Преди да бъде завършен проекта, трябва да сме сигурни ,че няма никакви софтуерни грешки и всичко работи безупречно. Така избягваме пробиви в защитата и предотвратяваме бъдещи сривове.

1.3.7 Окончателно завършване

След като всички стъпки по изграждането са готови, преминали сме всички тестове и знаем , че няма проблеми, можем да пуснем продукта в експлоатация. Това обаче далеч не означава, че сме готови. Нужни са постоянно потдържане на сигурност и готовност за оправяне на проблеми в реално време. Този етап не може да бъде избегнат и се явява при всички видове уеб приложения и уеб сайтове.

## 1.4 Предимства и недостатъци при разработка на уеб сайта

### 1.4.1 Предимства

1. Моделиране на софтуера по нуждите на клиента.
2. Лесна промяна на цветове, дизайн и структура.
3. Бърза ориентация на екипа, защото продукта е изработен от тях.
4. Лесно надграждане и добавяне на функционалности.

1.4.2 Недостатъци

1. Не всяка промяна е възможна, поради липса на ресурси и време.
2. Възможно допускане на грешки при изготвяне на продукта и договарянето с клиента.
3. Нужда от постоянното потдържане на сигурност.

Глава 2: Теоретична част

## Използвани технологии и работни рамки

В настоящия проект са използвани следните технологии, езици и среда за разработка /I D E – Integrated Developement Enviroment/, работни рамки /FrameWorks/ и библиотеки/Library/

1. HTML
2. CSS
3. PHP
4. Java Script
5. XAMPP
6. Visual Studio Code

### 2.1 HTML

**Какво е HTML?**

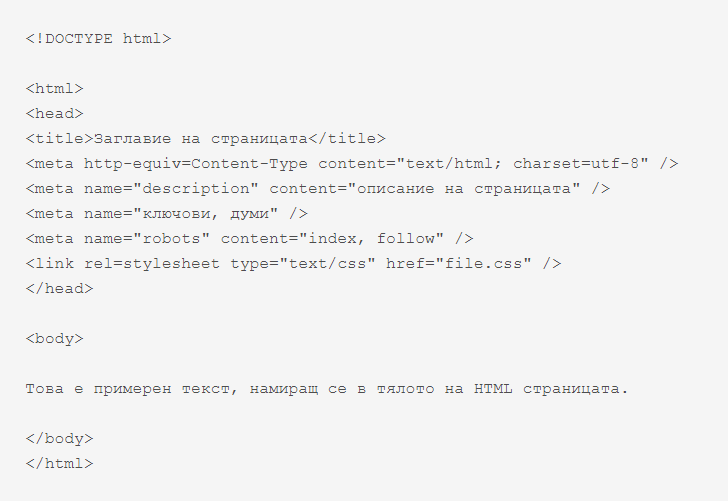
**HTML** е маркиращ език, който служи за скелет на уебстраниците. Използват се тагове, които са вградени и описват условното съдържани на страницата. Езикът е създаден така, че да може да се чете на всеки браузър и да може да се пише на всеки компютър, стига да има изтеглена среда за разработка (IDE). Тъй като той не предоставя желаната красота, **HTML** често се използва с други езици за създаване на уебсайтове.

Почти задължителното използвяне на **HTML** с помощни езици ни доказва, че той е податлив на обработка и ни отдава свободата за моделиране на кода и съдържанието на разработваната уебстраница по желанията на разработчика.

**За какво се използва HTML?**

**HTML** се намира във всички сайтове в свтовната мрежа. След създаването на сайтовете, те се качват на сървър, свъран към **WWW** (World Wide Web). Достъпването до сайта се случва чрез инренет търсачки, а те интерпретират кода и от него се показва страницата.

Начина на операция на интернет търсачките ни позволява да достъпваме уебсайтове от всяко устройство имащо такава и достъп до интернет връзка. Такива устройства са: телефони, лаптопи, настолни компютри, телевизори, таблети в коли и тн.



**Фигура 2.1.1**

Пример за началото на **HTML** страница

### 2.1.1 Предимства на HTML

1. Лесен за ползване.
2. Лесен за учене.
3. Лесен за редактиране.
4. Безплатен.
5. Чете се от всички интернет търсачки.

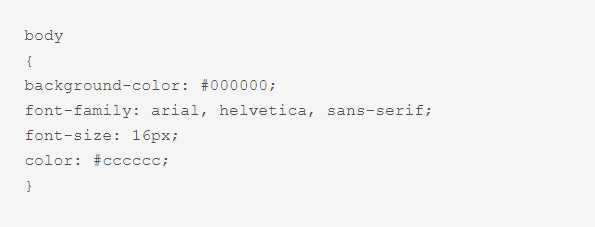
### 2.1.2 Недостатъци на HTML

1. Най-новата версия не се потдържа от старите браъзъри.
2. Не е красив, сам по себе си.
3. Не е гъвкав.

### 2.2 CSS

**CSS** (Cascading Style Sheets) е език за описване презентацията, видимоста и стила на **HTML** елементите. **CSS** е неразделим от **HTML** и рядко се вижда уебстраница без него. При [създаване на **HTML** страница](https://help.superhosting.bg/create-html-page.html), съдържанието й се описва (маркира) с **HTML** код, а презентацията на това съдържание, тоест как ще изглежда то в браузъра, се описва с **CSS** код (стил). **CSS** кодът се поставя във файл с разширение .css. След това този файл може да се използва във всяка една страница от уеб сайта. Така че стилът ще се намира на едно място и при промяна ще се отразява на всички уеб страници. **CSS** може да се приложи по 3 различни начина:

1. Външен – Изготвя се нов файл и в него се записва всичкия код. Поставя се разширение .css
2. Вътрешен – В началото на **HTML** файла се записва <style> таг. **CSS** кодът се записва в него, след което тагът се затваря - </style>.
3. Вграден – CSS стилът се разполага като атрибут директно в HTML тага например **<p>**.



**Фигура 2.2.1**

Примерен **CSS** код

### 2.2.1 Предимства на CSS

1. Ползването му е изключително лесно.
2. Научаването му е лесно.
3. Прибавянето му към работещи сайтове се случва бързо.
4. Редактирането и откриването на греши е лесно.

### 2.2.2 Недостатъци на CSS

1. Нужна е добра структира на **HTML** кода, за да може да е удобно и безпроблемно ползването на **CSS.**
2. Има ограничение на красотата на анимациите, които можем да създаваме с него.
3. При успоредното използване на жъншни библиотеки за дизайн, може да се получат грешки.

### 2.3 PHP

**PHP** е [скриптов език](https://bg.wikipedia.org/wiki/%D0%A1%D0%BA%D1%80%D0%B8%D0%BF%D1%82%D0%BE%D0%B2_%D0%B5%D0%B7%D0%B8%D0%BA) със синтаксис, базиран на [**C**](https://bg.wikipedia.org/wiki/C_(%D0%B5%D0%B7%D0%B8%D0%BA_%D0%B7%D0%B0_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%B8%D1%80%D0%B0%D0%BD%D0%B5)) и [**Perl**](https://bg.wikipedia.org/wiki/Perl). Използва се предимно в интернет среда за изпълнение на широк кръг от услуги. Фактът, че **PHP** се разпространява свободно, го прави удачен избор за изграждане на [уеб сървър](https://bg.wikipedia.org/wiki/%D0%A3%D0%B5%D0%B1_%D1%81%D1%8A%D1%80%D0%B2%D1%8A%D1%80), базиран изцяло на свободни продукти – **Linux**, **MySQL** и др. **PHP** файловете могат да съдържат текст, [**HTML**](https://bg.wikipedia.org/wiki/HTML), [**CSS**](https://bg.wikipedia.org/wiki/CSS), [**JavaScript**](https://bg.wikipedia.org/wiki/JavaScript) и **PHP** код и имат разширение .php. **PHP** може да се прилага и във файлове с различни разширения. При работа с вградено извикване се използва тага <?php(желана фукционалност)?>. Езикът се използва за работа зад завесата (backend), като извършва много функционалности при създаването на уебсайтове: връзка с база данни, обработване на снимки, обработване на низове чрез регулярни изрази, работа с дата и час, работа с принтери, изпращане на имейли, изпълняване на логика и др.

**PHP** може да работи с повечето видове бази данни. Това ни позволява лесен достъп до данни запазени в базата данни и лесната им обработка и валидация. **PHP** се използва често при работа с пароли, потребителски имена и имейли и доставя достатъчна сигурност, стига разработчиците да са изчислили нужната строгост на сигурноста.

### 

**Фигура 2.3.1**

Примерен **PHP** код

### 2.3.1 Предимства на PHP

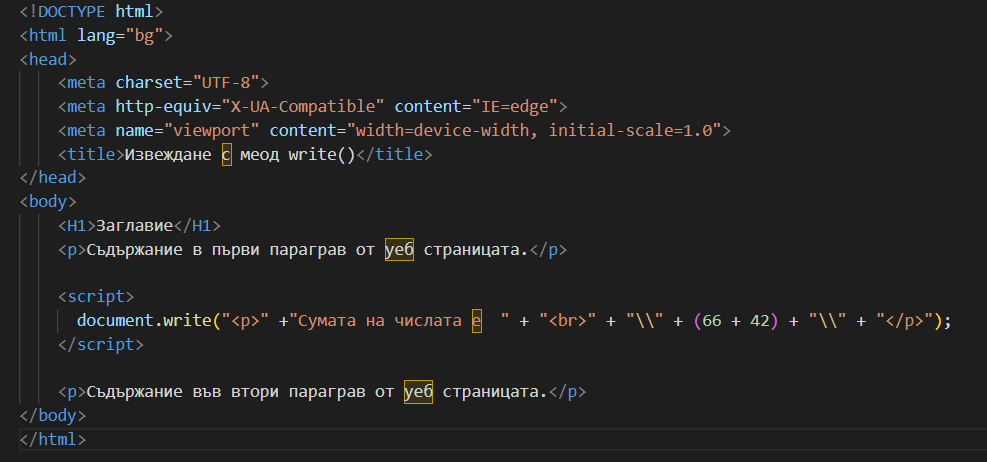
1. Лесен за разбиране логически.
2. Лесен за разбиране визуално.
3. Полезен при връзки с бази дании.
4. Лесна интеграция в **HTML** код.
5. Възможност за обработка на данни изпратени от сървъри и груги езици за профрамиране.

### 2.3.2 Недостатъци на PHP

1. Труден за научаване.
2. При по-малък опит на разработчиците, кодът често става нечетим.
3. Отнема време за построяване на пътищата по които да протичат данните.
4. Трудно се откриват грешки.

2.4 Java Script

**JavaScript** е един от най-популярните и използвани езици за програмиране в света. Той е лек, отворен и междуплатформен език за програмиране. **JavaScrip** е програмен език, който трябва да бъде “превеждан” или интерпретиран. Интерпретатора му е вграден в софтуера на уеб браузърите. **JavaScript** е обектно-базиран скриптов език,създаден от Netscape за използване в **HTML** документи и позволява взаимодействие с уеб страници по различни уникални начини.

 **JavaScript** наподобява **PHP** по аспекта с операциите по логическата част, но ако оставим това, те са коренно различни. **PHP** се използва главно за логика и функционалност, която е скрита от потребителя. **Java** няма нищо общо с **JavaScript**, въпреки използваната част от името му. **JavaScript** се използва осван за логическите елененти и за анимации, движение по екрана и др. **JavaScript** разполлага и с множество библиотеки, които се добавят отделно и предлагат много улеснени и полезни функции. Така работата на разработчика се улеснява значително. **JavaScript** може да се вгражда в **HTML** файлове и да се пише в отделен файл.

**Фигура 2.4.1**

Примерен **JavaScript** код

2.4.1 Предимства на Java Script

1. Може да се използва както за дизайн, така и за логика.
2. Може да се изпълнява във файлове с други разширения.
3. Може да се поставя в код на друг език за програмиране.
4. Има нова, адаптивна структура наречена **JSX**. Тя позволява писането на **HTML** в Java Script обекти и файлове.
5. Работи добре с готови библиотеки.
6. Има изградени библиотеки за почти всичко, което може да ви е нужно.

2.4.2 Недостатъци на Java Script

1. Не може да осъществи директа връзка към база данни.
2. Има свойства, които биха могло да объркат начинаещите.

2.5 Библиотеки на Java Script

2.5.1 React.js

**React.js** е библиотека изградена на **JavaScript**. Той притежава предваритело създадени елементи, които помагат на разработчиците с по-лесното изграждане на дизайнерската и структурната част на уебстраниците. При работата с него, разработчика не е задължен да използва предварително зададена структура на страницата и му предоставя свобода на решения. **React.js** работи лесно и с библютеки от трети страни и предоставя пълна свобода на действията. Това го прави лек за работа и за системата. Той обаче няма да ви помогне със сървърна комуникация, транслации и други подобни, което обаче изобщо не означава, че **React.js** е безполезен.

Пример за библиотека от трето лице е **Ant Design**. Тя е библиотека, която предоставя готов дизайн. Това улеснява и съкращава работата на програмистите в малки екипи и по-неопитните дизайнери.

Гъвкавоста на **React.js** го прави перфектен за създаване на сайтове с повтаряща се структура. Такива сайтове са например: YouTube, Amazon, Facebook и много други. Повтаряемоста в шаблоните се среща почти навсякъде в днешните уебсайтове и това спомага популяризацията на **Rasct.js**.

2.5.1.1 Предимства на React.js

1. Лесен за използване.
2. Лесен за научаване.
3. Четлив.
4. Адаптивен към други езици.
5. Адаптивен към библиотеки от трети страни.
6. Безплатен.

2.5.1.2 Недостатъци на React.js

1. Има свойства, които биха объркали по-неопитните разработчици.
2. Не може да направи директа връзка към база дании или страничен сървър.

2.5.2 Chart.js

**Chart.js** е библиотека построена върху **JavaScript**. Визуализацията на данни е хубава важна практика в програмирането. Тя е нужна почтi винаги и спомага за официялният изглед на уеб приложенията. **Chart.js** се грижи точно за този аспект на програмирането. Графиките, които тя предлага са: решетъчна, линийна и много други. За всяка графика са разработени разновидности, като може да се използват и смесени такива. Библиотеката позволява и манипулация на различните предварително изготвени елементи и дизайна им.

2.5.2.1 Предимства на Chart.js

1. Улеснява работата на потребителите.
2. Лесна за работа и инсталиране, адаптивна и красива на външен вид.
3. Използването й спомага за намаляването на ресурсите, които трябва да се отделят за изработването на проекта и улеснява работата на разработчиците.
4. Анимациите, които са вградени в библиотеката са красиви.
5. Безплатна.

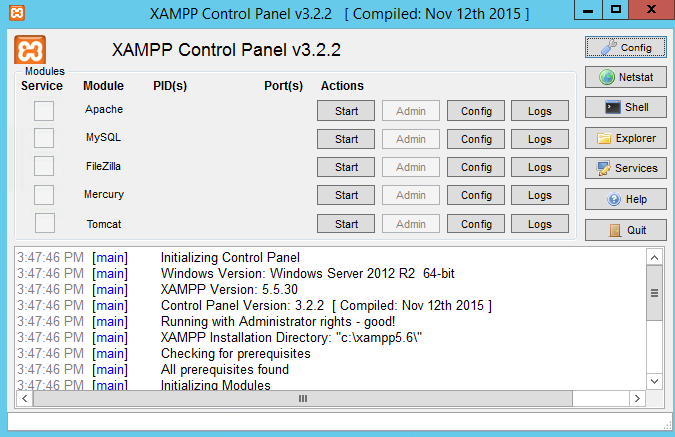
2.5.2.1 Недостатъци на Chart.js

1. Трудно правене на редакция върху готовия код.
2. Ограничен брой графики.

2.6 XAMPP

**XAMPP** е локален сървър служещ за локалното тестване и съзадаване на уебсаутове. Съкратен е от ***Apache***, ***MySQL***, ***PHP*** и ***Perl***. Лек е като система и е перфектно решение за разработването на по-малки проекти. Локалният сървър е нужен на разраотчиците, защото не може да се пусне нова версия на продукта, която не е тествана. Това би счупило системата и би я направило неизползваема, доато се разработи и качи новата логика. Може да се инсталира на Windows, Linux и Mac.

Тъй като не е самостоятелно приложение, **XAMPP** предоставя два основни компонента за неговото инсталиране – **Apache**, който се използва за създаване на локалния сървър, и **MySQL**, който можете да използвате като база данни за вашия уебсайт.



**Фигура 2.6.1**

Контролен панел на **XAMPP**

Тъй като **XAMPP** eсъчетание от няколко различни софтуера, той има две главни части - **Apache**, който се използва за създаване на локалния сървър, и **MySQL**, който може да бъде използван като база днни за уебсайта.

***Компонентите на XAMPP :***

1. **Apache** е най-широко използваният софтуер за сървъри. Надежден е, бърз, удобен, потдържа се от основните операционни системи и може да се ползва и на реални сървъри. Той служи за връзката между логиката на приложението и уебсайта, който вижда потребителя. Информацията минава през логиката и се праща към базата данни. Извършва се и обратната връзка, като се праща информация от базата данни, до логикаъа и след това до уебстраницата.
2. **MySQL** е система за управление на база данни.

Базата данни е структурирана колекция от данни. Такива данни може да е списък с имейли и пароли или нормален списък за пазаруване. Всякакъв вид данни може да бъде представен под формата на таблица. Това представяне е един от най-логичните начини за четливо предтавяне на информация.

Релационното запазване ни позволява да запазваме таблиците в отделни файлове, да има подреденост и да не струпваме информацията на едно място. Така можем да правим заявки и да извличаме информация от таблиците, без да има каквото и да е объркване. Това може да се случи чрез директни заявки в интерфейса на **MySQL** Server, в кода на логическата част на уебсайта и др.

**MySQL** има отворен код. Това ни позволява да го достъпим по всяко време, чрез данните за акаунта, на който принадлежи базата данни. Това може да се случи по всяко време и от всяко устройство, което има инсталиран нужния софтуер за редакция.

**MySQL** е лек, бърз и надежден. Това ни позволява да използваме почти пулната мощност на компютъра, дори базата данни да е включена. Това обаче важи за малките проекти с малко на брой зачвки на веднъж. В обратен случай системата трябва да преоретизира изпълнението на заявките и са нужни ресурси за тази дейност.

**MySQL** Server първоначално е разработен за обработка на големи бази данни много по-бързо от съществуващите решения и успешно се използва в силно взискателни производствени среди от няколко години. Въпреки че е в непрекъснато развитие, MySQL Server днес предлага богат и полезен набор от функции. Неговата свързаност, скорост и сигурност правят MySQL Server изключително подходящ за достъп до бази данни в Интернет.

**MySQL** е широко разпространен и безплатен. Това води до лесния достъп до

нужни знания, защото има лесен достъп до опита на разработчиците. Има и много изработени материали ако се нуждаем от тях.

1. **Pearl** е слабо типизиран език. Най-силните му ехнологии са регулярните изрази и вградените сложни структури от данни. Той може да се пише процедурно, обектно ориентирано и функционално. Изграден е от езиците: C, awk, sed, Lisp и др. В допълнение към **CGI**, Perl 5 се използва за системна администрация , мрежово програмиране , финанси, биоинформатика и други приложения, като например GUI . След написването на кода, той се компилира. Получения реултат е интерпретиран до байткод, кратък и силно компресируем. За разлика от Java, байткодът не е лесно достъпен, благодарение на което са избегнати проблемите с обратната съвместимост.

2.6.1 Предимства на XAMPP

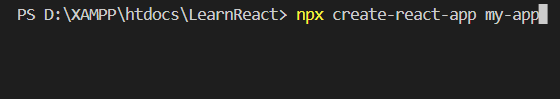
1. Лесен за използване.
2. Лесен за инсталиране.
3. Рядко възникват грешки при използването му.

2.6.2 Недостатъци на XAMPP

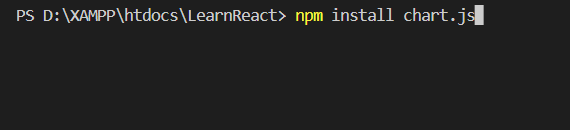
1. Понякога се объркват пътищата на сървърите за различните вградени технологии.

# Глава 3: Инженерно решение на поставената задача

За реализирането на моя сайт използвам Visual Studio Code. Важно е да са изтеглени библиотеките: React.js и Chart.js. В противен случай сайтът няма да работи. За да ги изтеглим ще ни е нужна рамката Node.js. На (Фиг.1) е показана командата за инсталиране на React.js, а на (Фиг.2) е показана командата за инсталиране на Chart.js. И двете команди са валидни само за Windows.

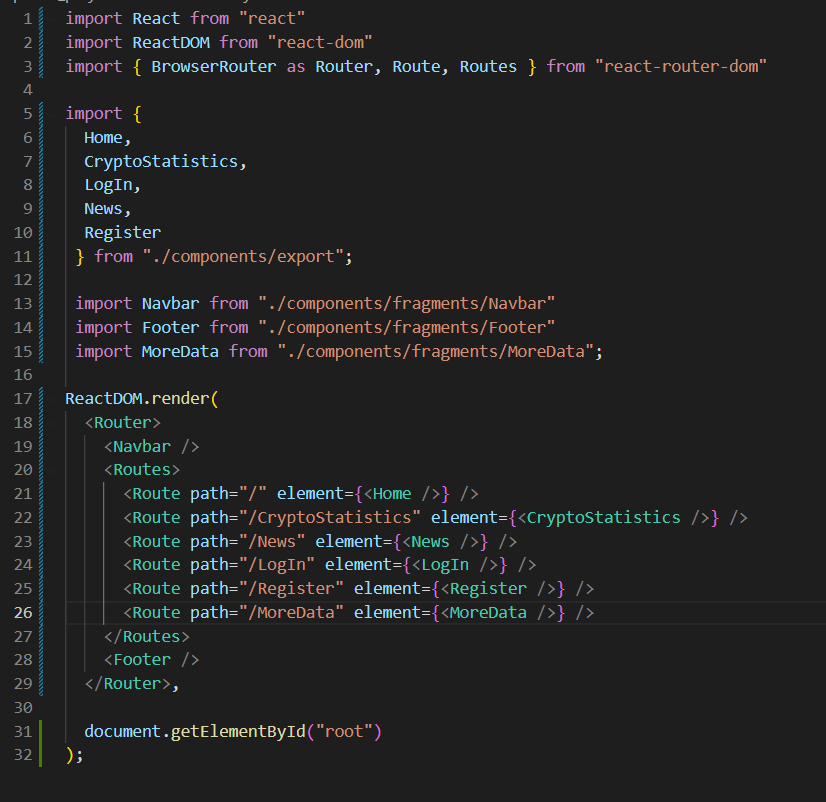
****

**Фигура 3.1** Инсталиране на **React.js**

****

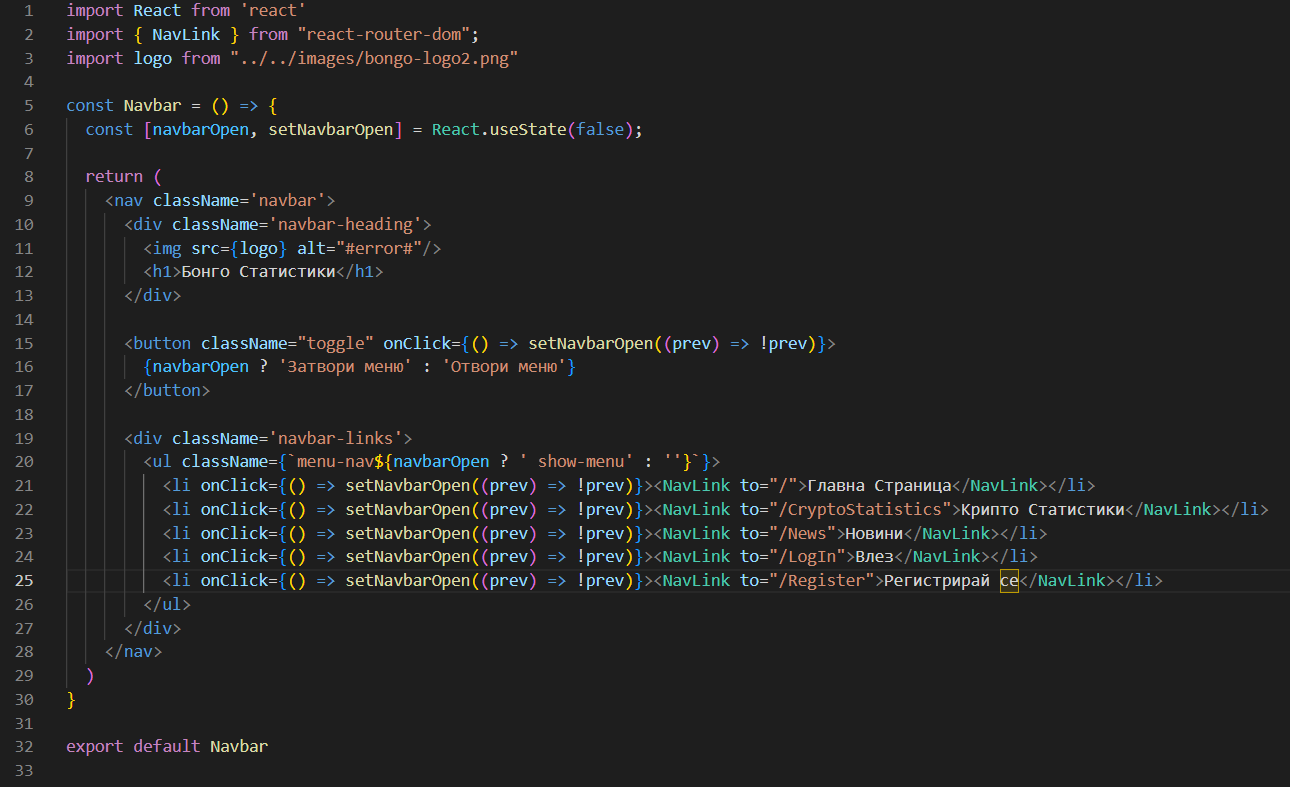
**Фигура 3.2** Инсталиране на **Chart.js**

Всички страници на уеб сайта са изградени от различни компоненти. Всеки компонент отговаря за отделна функционалност. Файлът от (Фиг 3.3) служи за скелет и създава всички пътища на проекта. Елементът „<Routes>“ ни позволява да създадем отделниете пътища и да държим структурата чиста. Всички отделни части са извикани предварително.

****

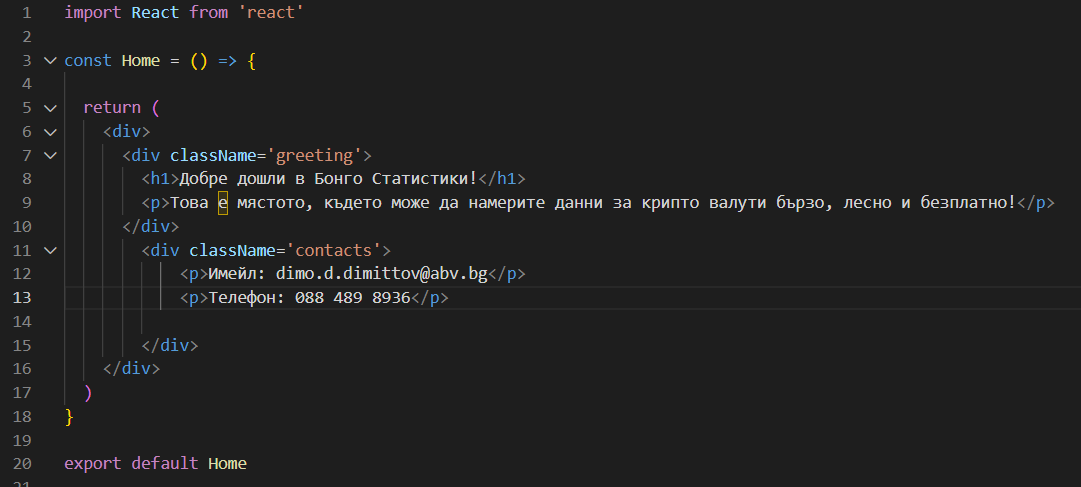
**Фигура 3.3** Код за пътищата на уеб сайта

Менюто на страницата е изградено от един файл и е посичено само веднъж. „<NavLink>“ елементът се грижи да бъдем препратени при натискането на различните бутони на менюто. Цялата структура е показана на (Фиг 3.4)



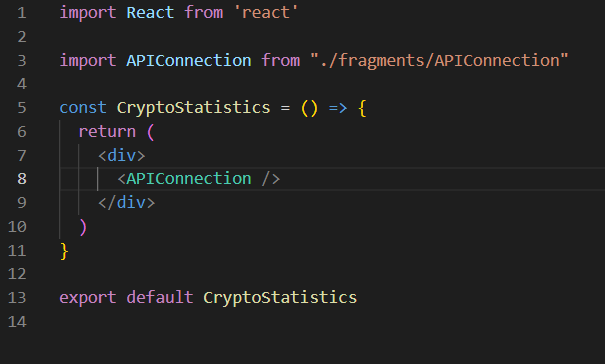
**Фигура 3.4** Меню

Началната страница е изградена от няколко прости елемента, които можем да видим на (Фиг 3.5). Тя е най-семплият файл от всички използвани в уеб сйта.

******

**Фигура 3.5** Начална страница

В следващите редове ще говоря за основната функционалност на проекта, а именно – „Крипто Статистики“. В нея са изобразени статистиките за топ 50-те крипто валути, наредени по популярност. Данните за тях се презареждат на всяка секунда. Първият файл (Фиг 3.6) е структурен и служи само за рамка.

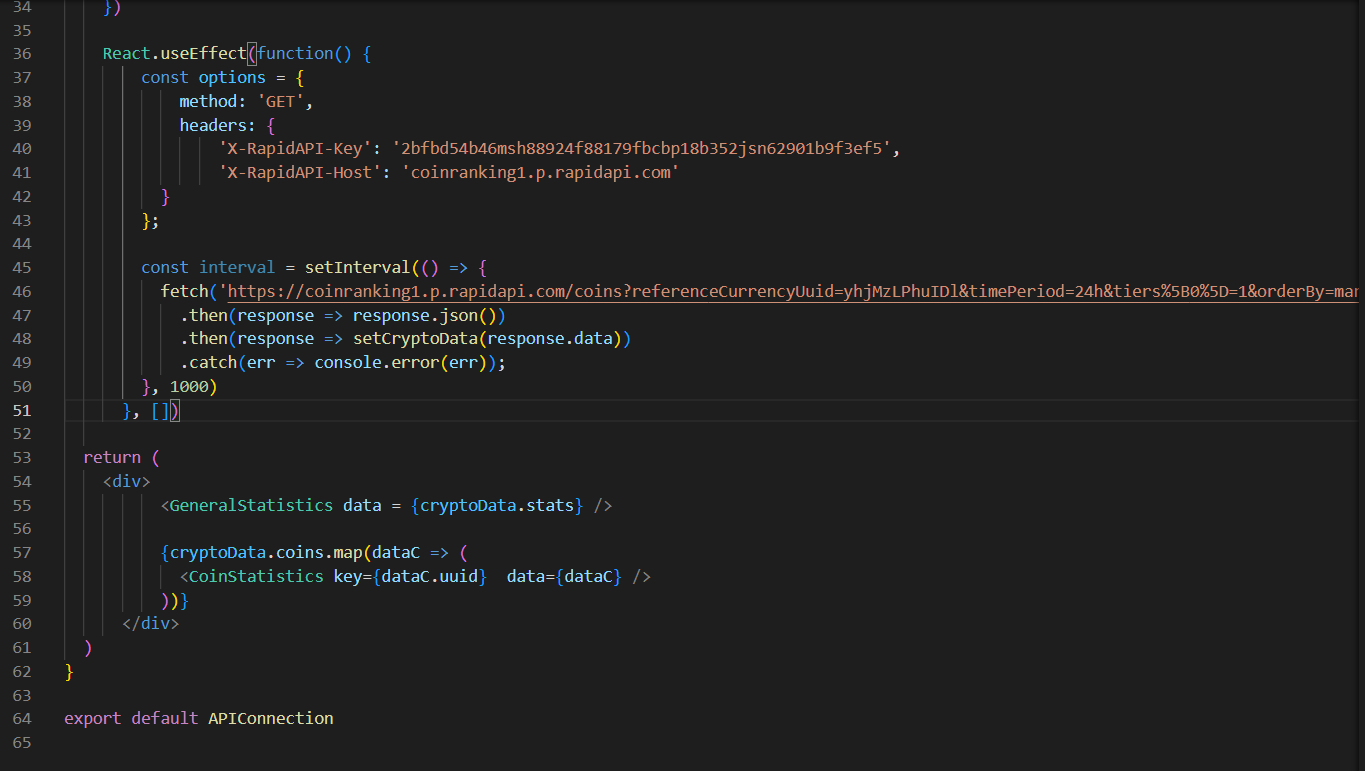
******

**Фигура 3.6** Главен файл на функционалност – „Крипто Статистики“

Следващият файл (Фиг. 3.7 и Фиг 3.8) служи за получаването на данните и дистрибуцията им към дъщерните компоненти. Данните се получават чрез изградената връзка с Coinranking API. Това е софтуер, изграден за бързото изпращане на данни за крипто валути и се използва главно от разработчици. За да запазя данните съм изградил обектна структура. Информацията се подава на двата компонента, които служат за обхождането им и изкарването им на екрана. Вторият извикан компонент – „<CoinStatistics>“, се използва за шаблон, като приема данните за всяка крипто валута и се повтаря докато те не се изчерпат.

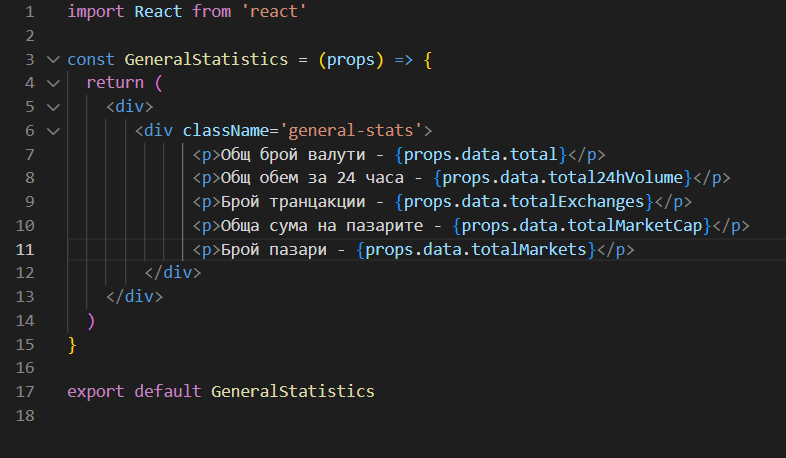
******

**Фигура 3.7** Структура за запазване на данните

******

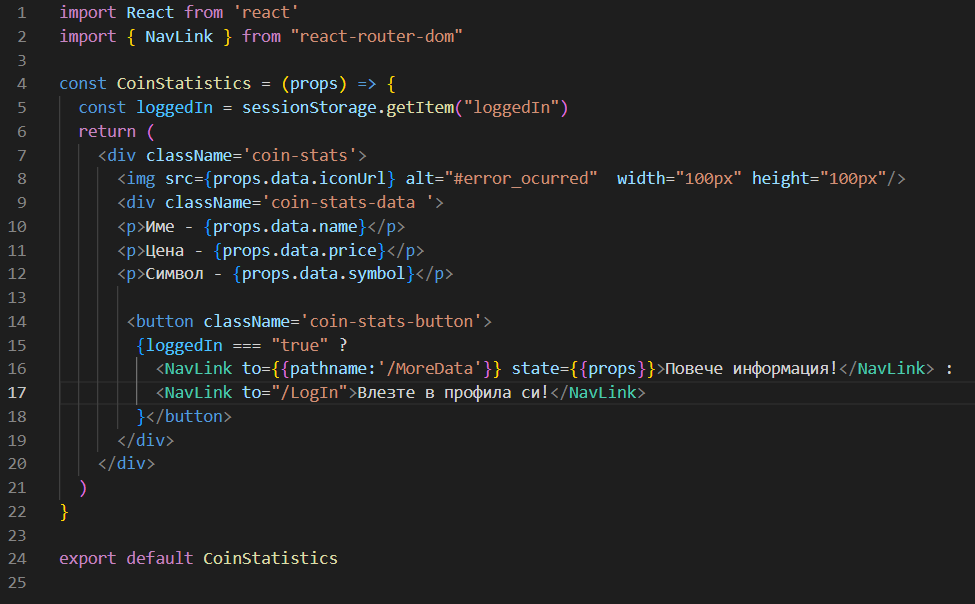
**Фигура 3.8** Връзка с външния софтуер

Компонентът от (Фиг 3.9) се състои от полетата, които изкарват подадените му данни. Той се извиква само веднъж.

******

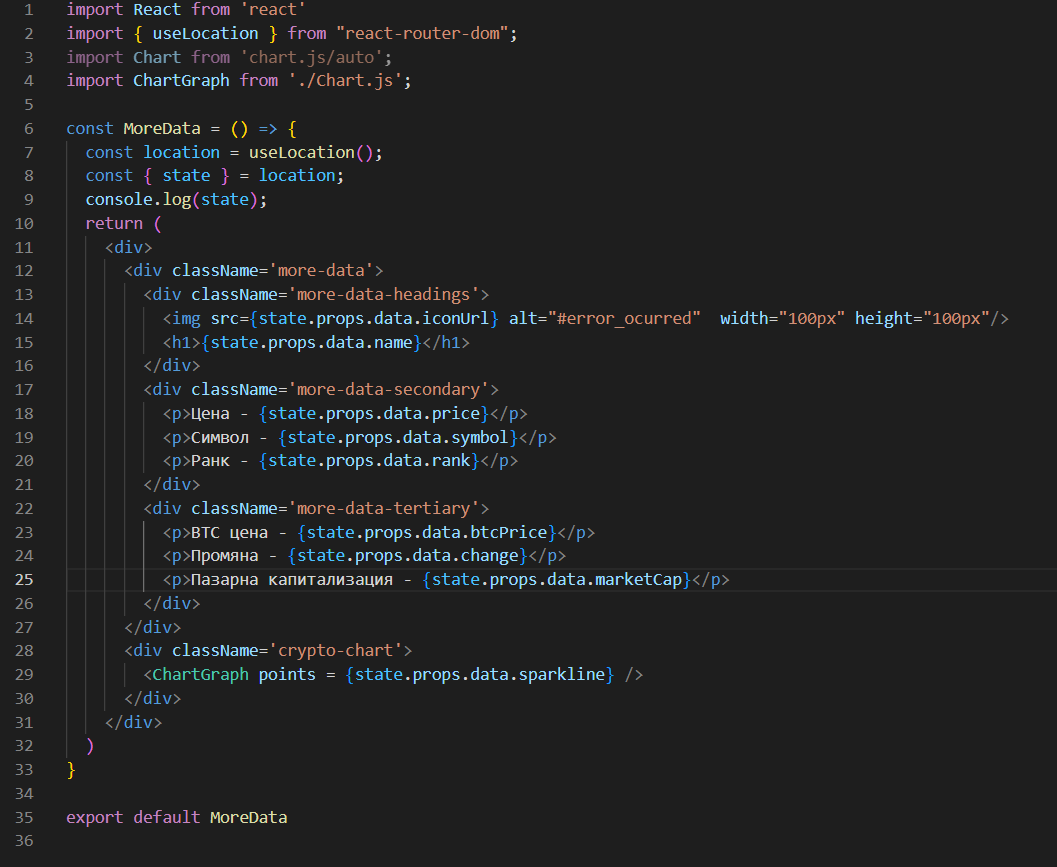
**Фигура 3.8** Компонент работещ с подадени данни

Компонентът на (Фиг. 3.9) не изкарва пълната подадена информация. За да получи достъп до цялата информация, потребителят трябва да влезне в профила си. Ако потребителят няма съществуващ профил, трябва да се регистрира.

******

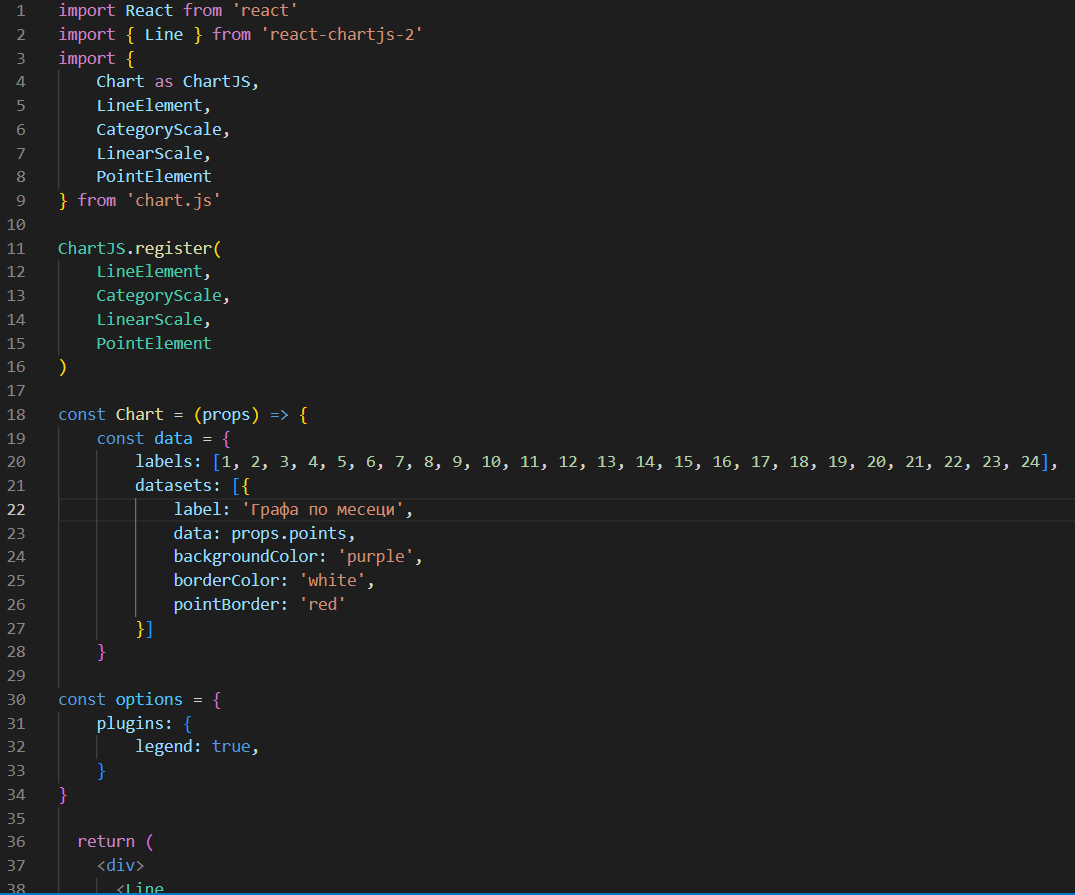
**Фигура 3.9** Компонент представящ съкратени данни.

След като сме влезли в профила си, сайтаът ни отвежда до друга страница. (Фиг 3.10) Тя показва цялата информация за избраната крипто валута. Част от получената информация се предава на градивен компонент.



***Фиг 3.10*** Компонент представящ пълни данни.

Компонентът „<ChartGraph>“, показан на (Фиг 3.11), използва получената информация за изграждането на диаграма. Тя е съставена на базата на цената на посочената крипто валута в рамките на две години. С това функциалноста за статистиките на крипто валути е обходена.



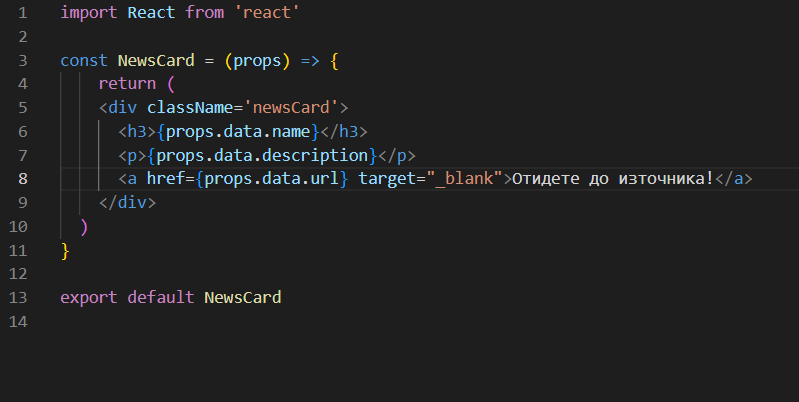
***Фиг 3.11*** Изграждане на диаграма

Преминаваме към друга функционалност на уеб страницата. Компонентът от (Фиг.3.12) прави връзка с Bing News Search API. Това е софтуер предназначен за изпращането на данни за актуални новини. Аз съм използвал новини от бизнес сектора, които са пряко свързани с пазарите и цените на акции и крипто валути.

Получената информация се обхожда и се подават на компонента от (Фиг 3.13). Данните се разделят и подават по отделно за всяка новина. Налична е и опция, която да ви отведе до източника на изпратената новина.

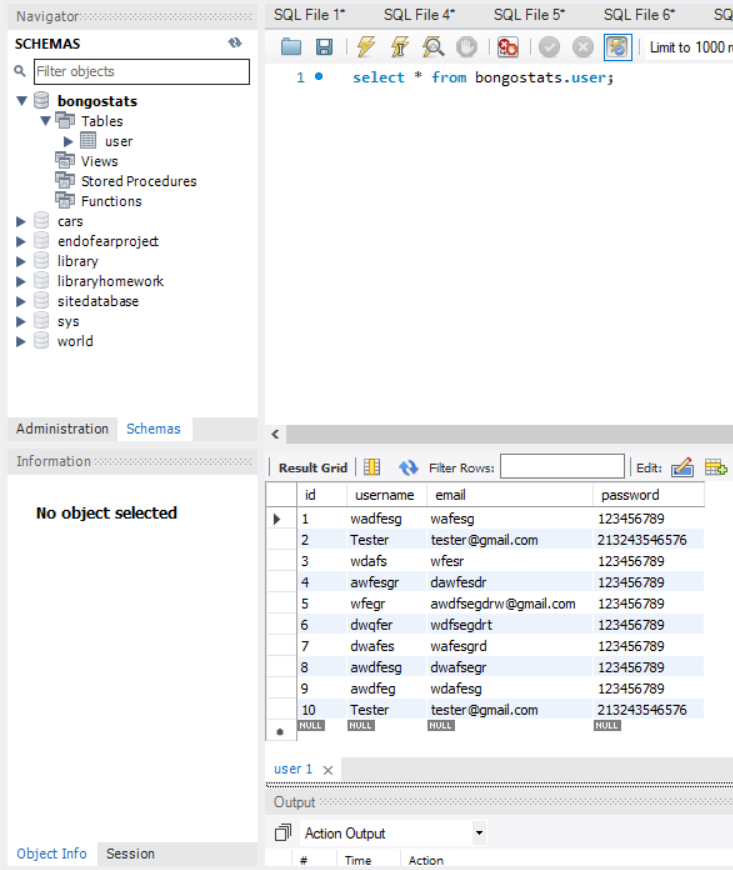


***Фиг 3.12*** Връзка с външен източник за новини

******

***Фиг 3.13*** Визуализиране на новини

Изградил съм база данни в която да се пазят данните за регитриралите се потребители. (Фиг. 3.14) В нея данните са представени под същинстата си форма. Този аспект може да се подобри с времето, като паролите на потребителите минават през хеш функция и се записва само получения хеш код. По този начин информацията на потребителите е предпазена на по-високо ниво и се предотвращават различни видове атаки.



***Фиг 3.14*** База Данни

За приемането и предаването на данни се грижат локалния сървър **XAMPP** и скрипта, който съм написал на езика за програмиране **PHP**. Локалният сървър се грижи за връзката с базата данни. Скрипта се грижи за създаването на връзка между локалния сървър и уеб сайта. (Фиг. 3.15) В кодът на **PHP** се дефинира връзката със сървъра, праща се заявка получаването на всички дании в базата с данни и се праща резултат на кода от (Фиг. 3.16 и Фиг 3.17).

******

***Фиг 3.15* PHP** скрипт за регистрационна форма

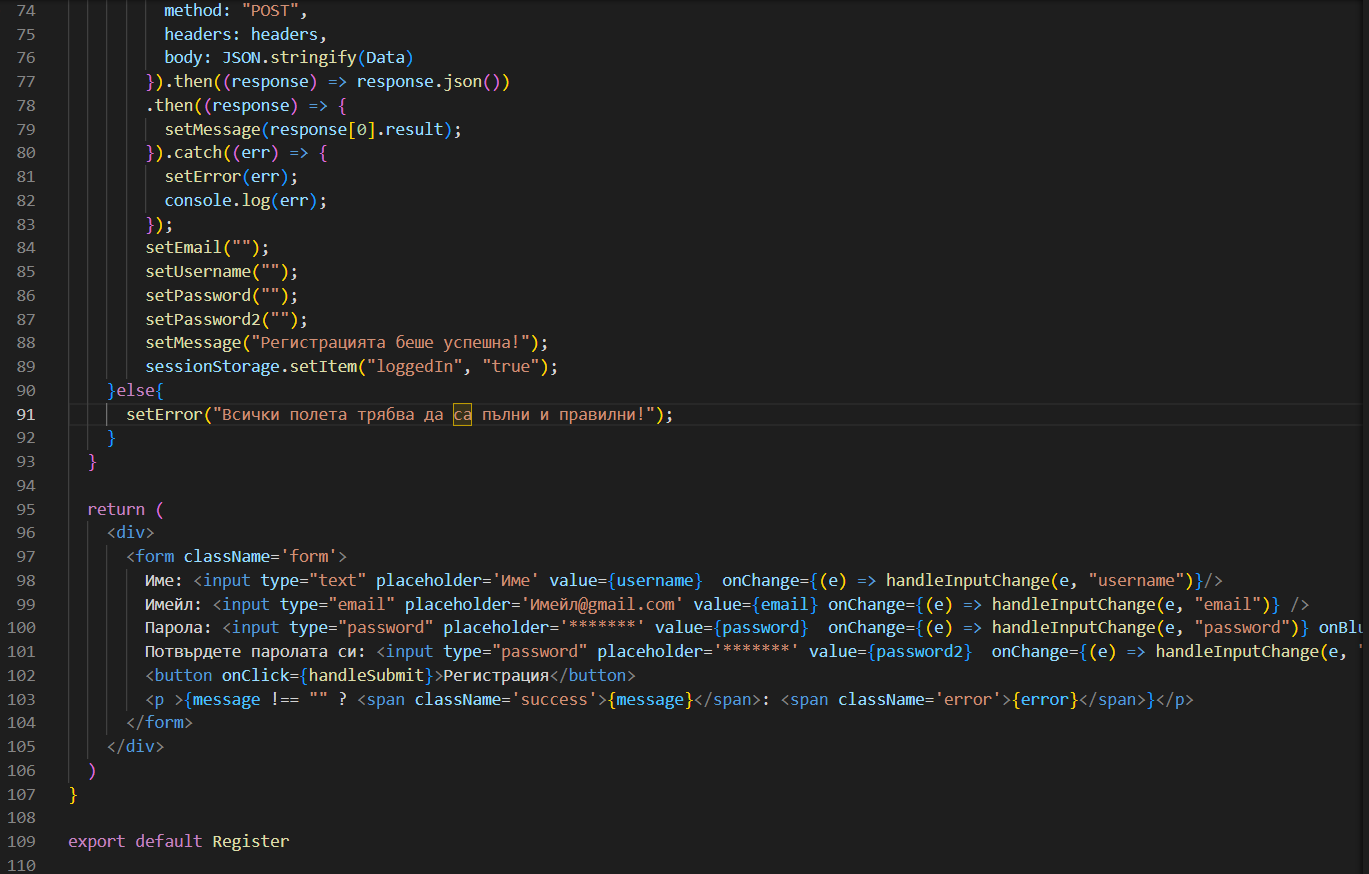
Файлът от (Фиг 3.16 и Фиг 3.17) за приемането на въведените данни на потребителя. Той ги валидира:

1. забранява празни полета;
2. забранява потвърждаващата парола да е различна от първата;
3. забранява

След като информацията е проверена и запазена се изпраща заявка до кода от (Фиг 3.15). Получава се отговор от кода, приключил въвеждането в базата данни и се визуализира съобщение за потребителя***.***

******

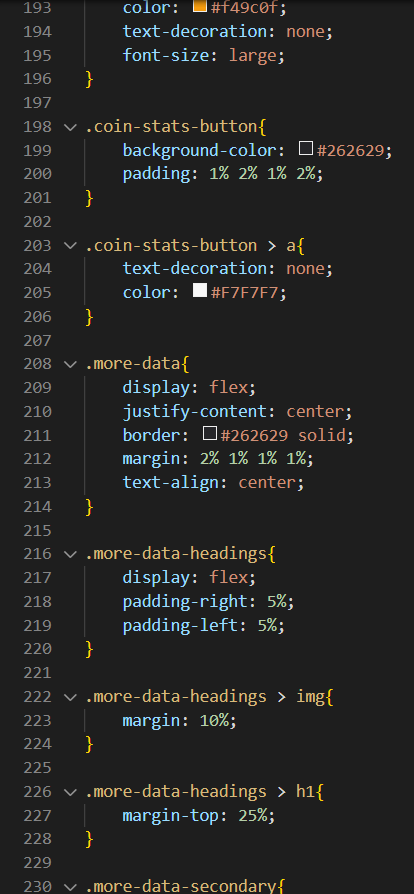
***Фиг 3.16* PHP** Валидиране на въведените полета

******

***Фиг 3.17* PHP** Полета за въвеждане

Логиката за влизането в акаунт е изградена по подобен начин на тази за регистрирането.

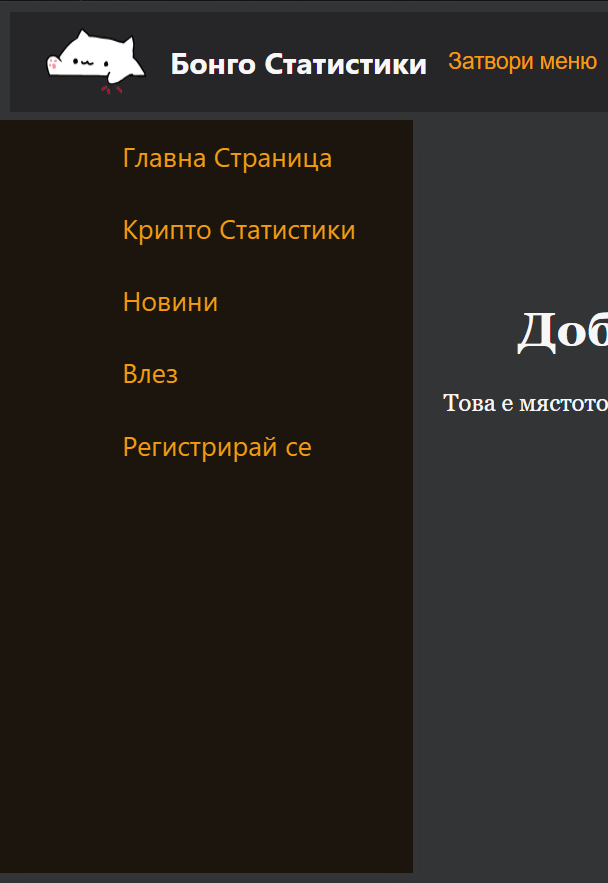
Цялата стилизация на уеб сайта е изградена чрез **CSS**. Той е записан само в един файл, защото не е обемен. За селектирането на различните елементи съм използвал „className“ тагове, които са вградени за **React.js**. Те ми позволяват да променям абсолютно всичко свързано по дизайна и да създавам анимации. Част от кода е представена чрез (Фиг 3.18).



***Фиг 3.18* PHP** Стилизиране на елементи

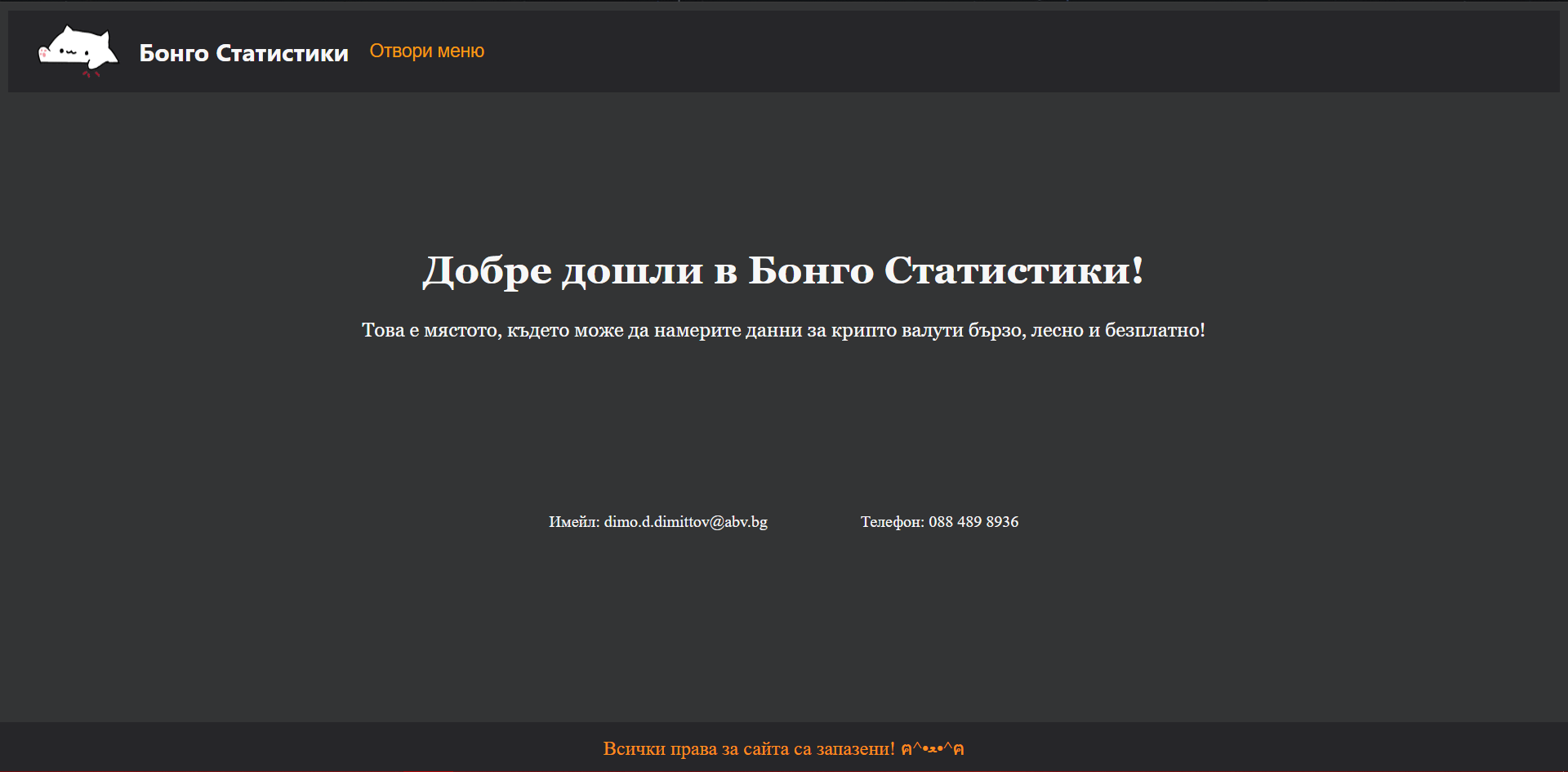
# Глава 4: Анализ на получените резултати, приложимост и изводи

Прилагам скрийншот на менюто на сайта (Фиг. 4.1). Чрез него, потребителите лесно достъпват това, което търсят. Навигацията са състои от пет бутона и всеки води към следните страници.



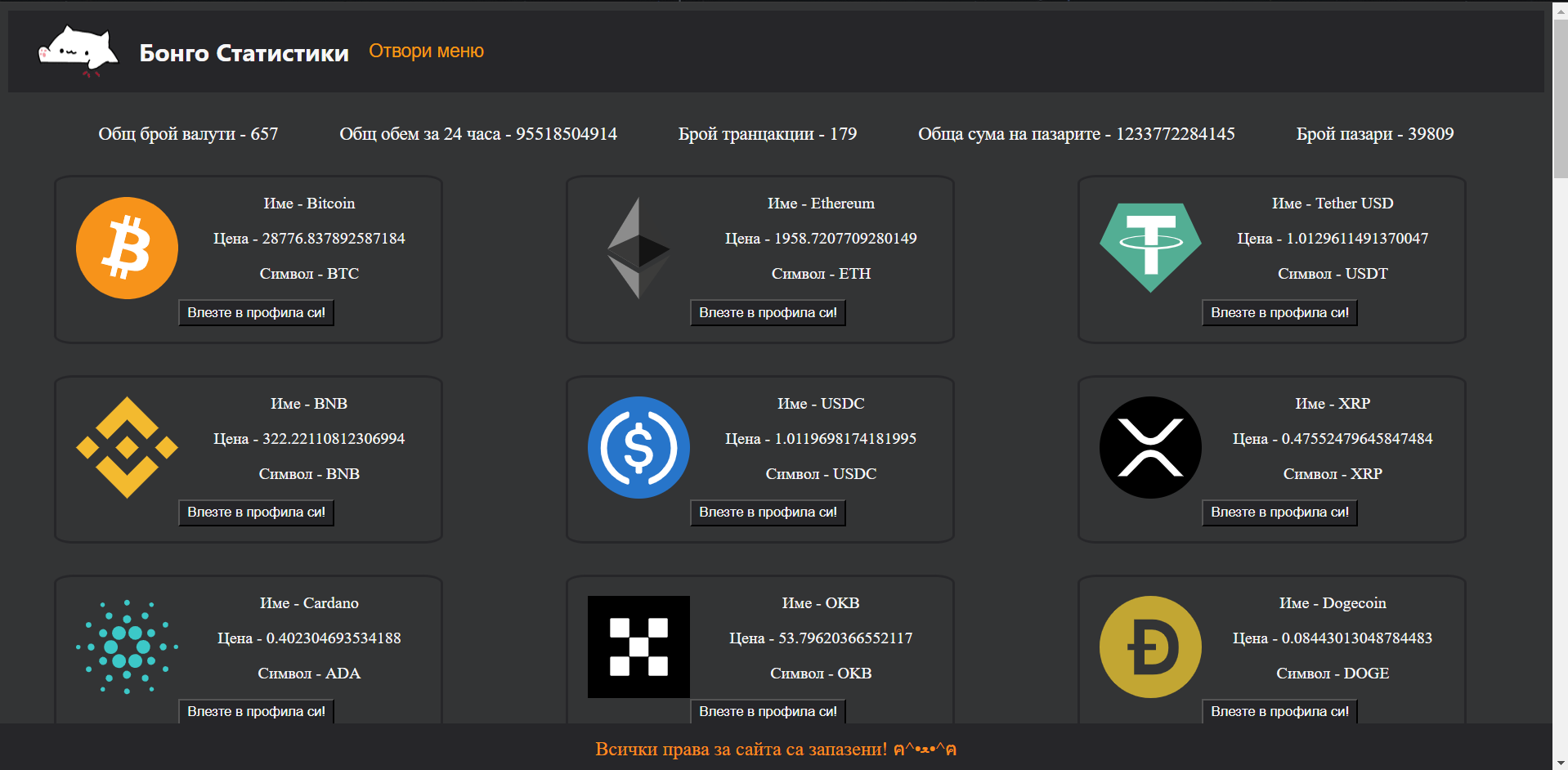
***Фиг 4.1*** Меню

Началната страница на сайта е първото нещо, което излиза при влизане. Тя се състои от поздрав и контакти, при нужда от помощ и докладване на проблем. В най-долната част на страницата е поставено поле. То заявява, че правата са запазени и се появява на всяка част от проекта, заедно с навигацичта. (Фиг. 4.2)

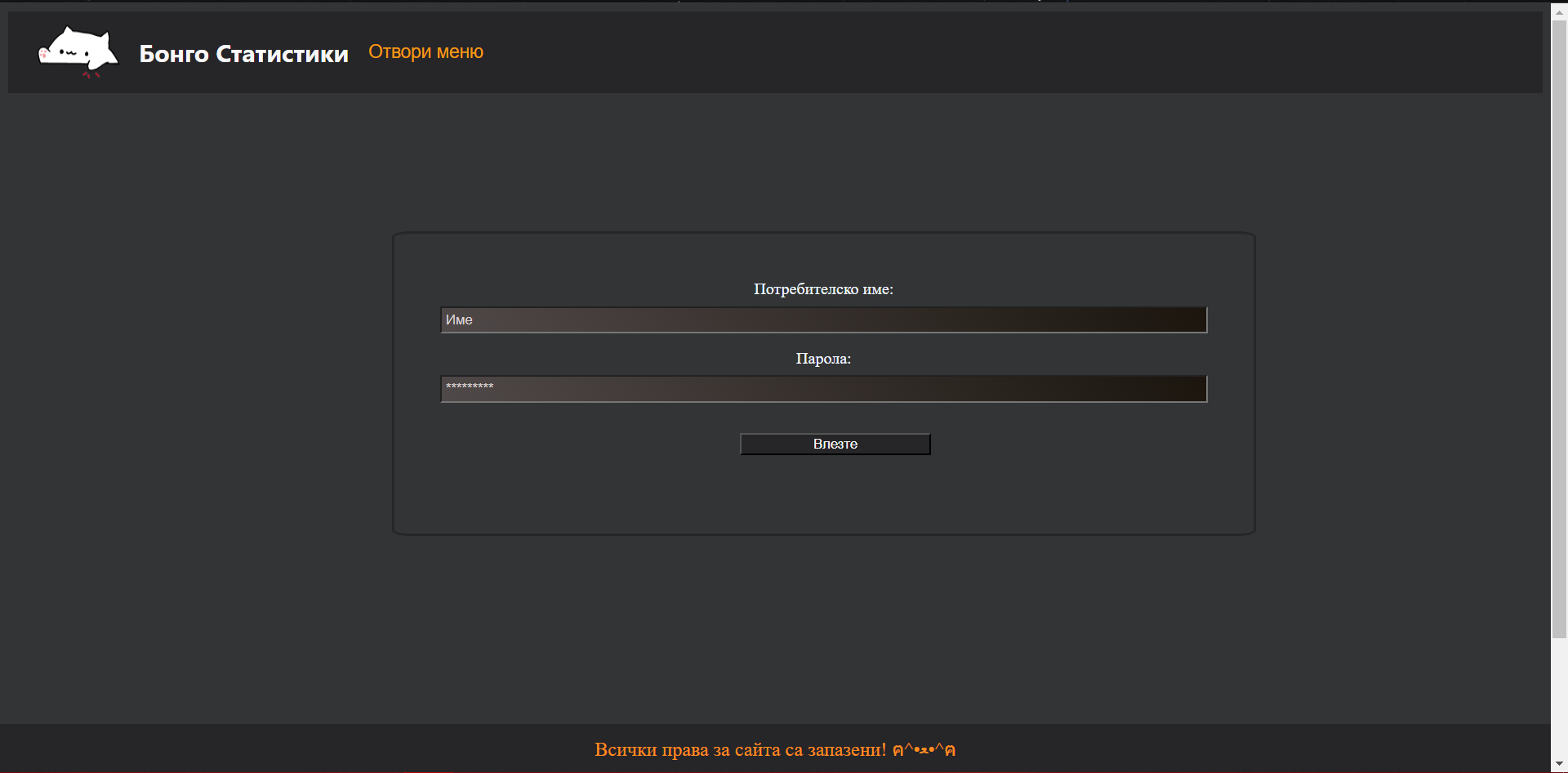


***Фиг 4.2*** Начална страница

Следващата част от уеб сайта са статистиките. (Фиг. 4.3) Първата страница ни показва съкратени данни за всички налични крипто валути. Бутонът в дъното на всяко поле ни подканва да влезнем в профила си, за да видим повече информация. След натискането му сме препратени към формата за влизане в профил. (Фиг. 4.4) При грешка е визуализирано предупреждение в червено, а при успех се показва зелен тексст.

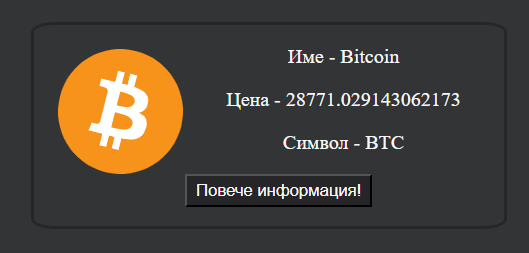


***Фиг 4.3*** Страница със статистики



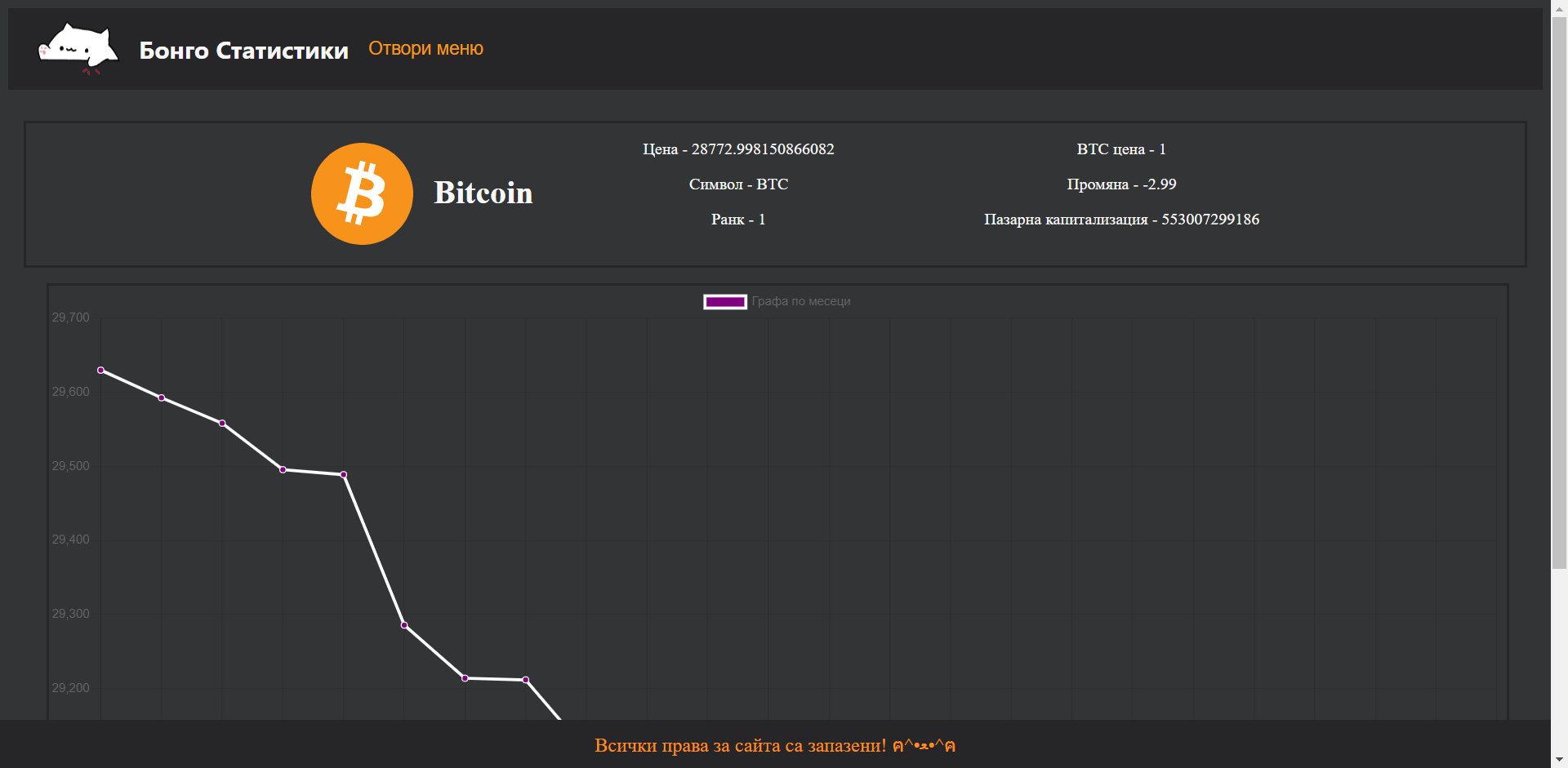
***Фиг 4.4*** Форма за влизане в профил

След като сме влезнали в профила си се връщаме в страницата със статистиките. Бутонът се е променил. (Фиг. 4.5)

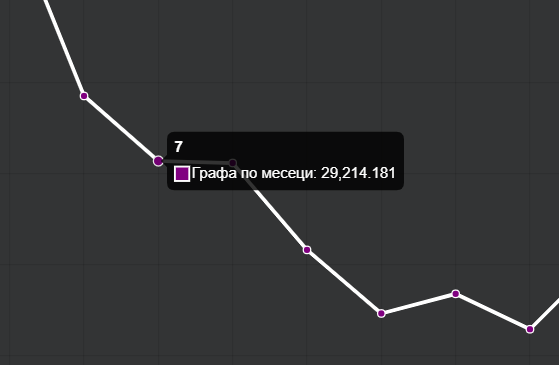


***Фиг 4.5*** Поле с данни

След натискането му сме отведени до (Фиг. 4.6) На страницата виждаме всички данни за крипто валутата и графика. Графиката показва информация за всяка поставена точка. (Фиг 4.7)

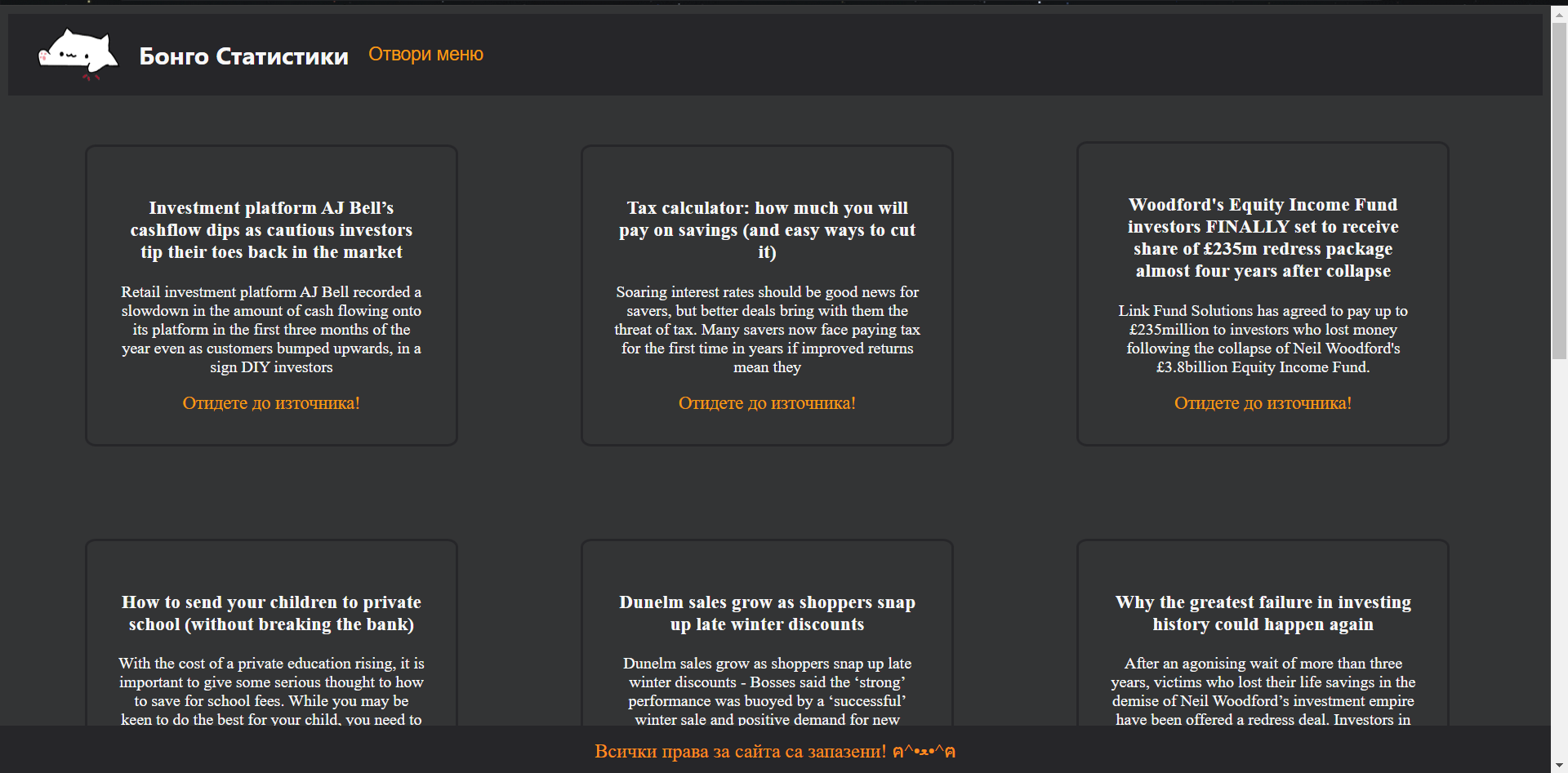


***Фиг 4.6***  Страница с графика

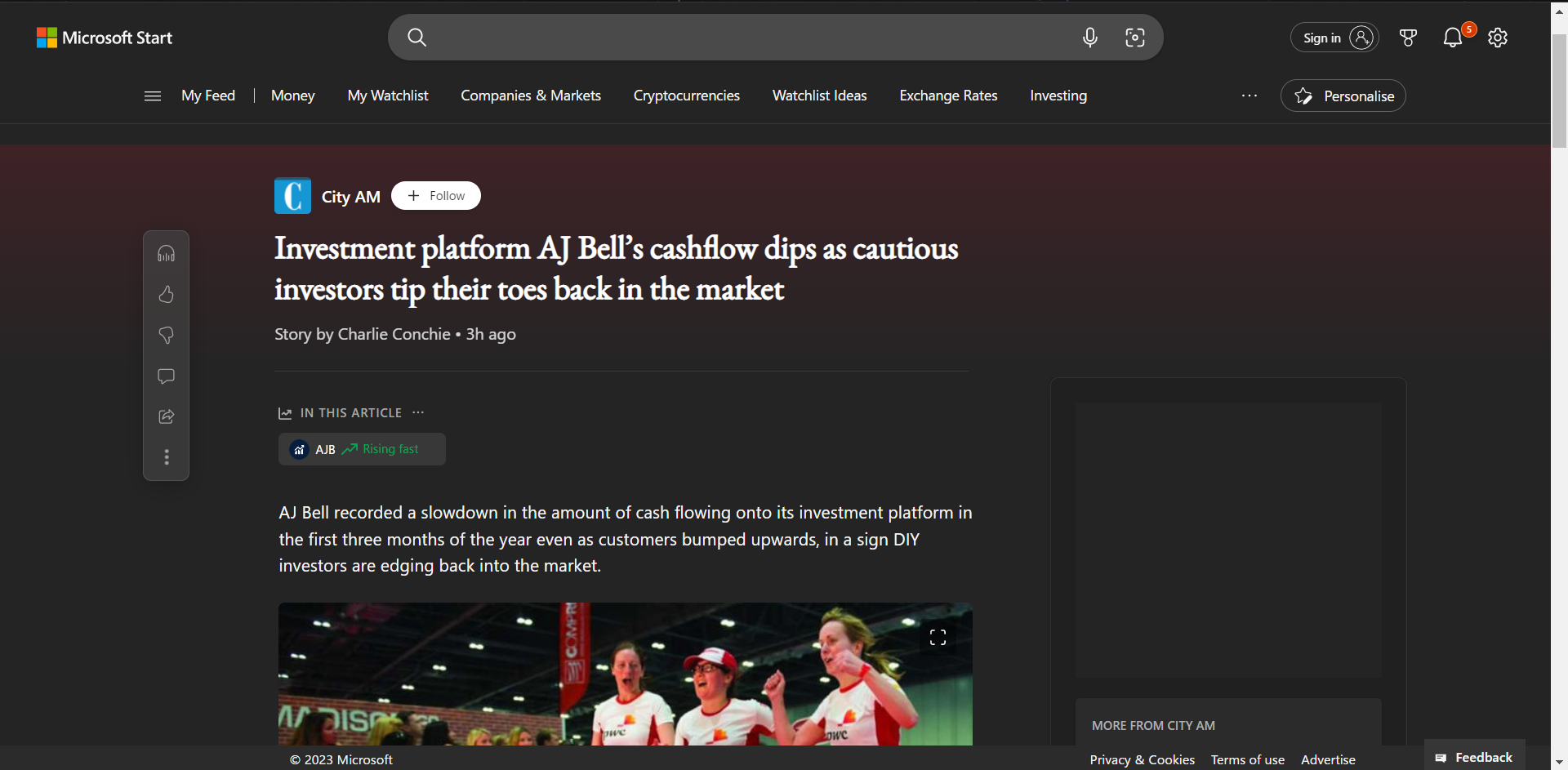


***Фиг 4.7***  Точка на графа

В следващото поле на навигацията натискаме „Новини“. То но отвежда до страница със съкратена информация за наличните новини. (Фиг. 4.8) Бутонът, който стои най-отдолу ни препраща до източника на новината. Всички новини са взето от сайта на Microsoft (Фиг. 4.9) и се обновяват всеки ден.

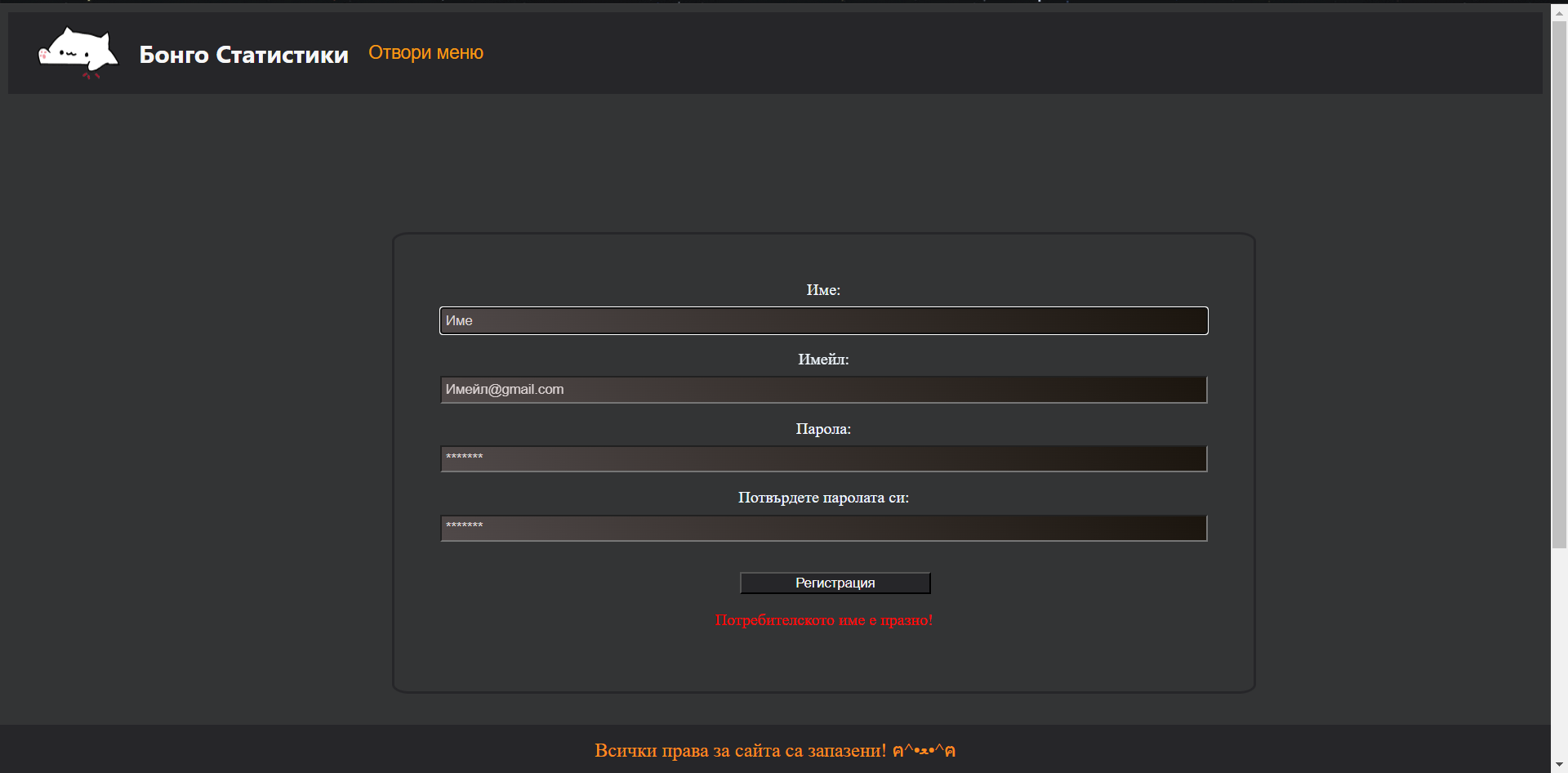


***Фиг 4.8***  Страница с новини



***Фиг 4.9***  Сайт източник

Последната, но не на последно място е страницата съдържаща регистрационната форма. (Фиг. 4.10) При грашка изкарва предупреждение в червено, а при успех, изкарва текст в зелено.



***Фиг 4.10***  Регистрационна форма

# Заключение

Желанието ми е, сайтът да продължи развитието си с течение на времето. Бъдещето му е:

1. Да се подобри дизайна и да се добавят аниации.
2. Да се добавят повече функционалности, като главната от тях е възможността да се търгува.
3. Да се подобри сигурноста и начина на пазене на личните данни на потребителите.

Създаването на уеб сайта беше забавно предизвикателство, което ми донесе много знания. Надявам се да бъде плезен на широка база потребители и да се популяризира.

## Използвана литература

[1] “Eclipse.” *Wikipedia*, <https://en.wikipedia.org/wiki/Eclipse>

[2] “IntelliJ IDEA.” *Wikipedia*, <https://en.wikipedia.org/wiki/IntelliJ_IDEA>

[3] “Visual Studio Code” *Wikipedia,* <https://en.wikipedia.org/wiki/Visual_Studio_Code>

[4] “Methods for building a website”, <https://www.dreamhost.com/academy/ways-to-build-a-site/>

[5] “Advantages and Disadvantages for building a website.”, <https://jellywebsites.com/subscription-websites/advantages-and-disadvantages-of-building-your-own-website-website-design-corona-ca/>

[6] “Java.” *Wikipedia,* <https://en.wikipedia.org/wiki/Java_(programming_language)>

[7] “HTML.” *Wikipedia,* <https://en.wikipedia.org/wiki/HTML>

[8] “HTML.” Disadvantages and advantages, <https://www.aplustopper.com/advantages-and-disadvantages-of-html/>

[9] “CSS.” *Wikipedia*, <https://en.wikipedia.org/wiki/CSS>

[10] “CSS.” *Advantages and Disadvantages,* [*https://bg.eyewated.com/предимства-и-недостатъци-на-инлайн/*](https://bg.eyewated.com/)

[11] React tutorial, [*https://scrimba.com/learn/learnreact*](https://scrimba.com/learn/learnreact)

[12] Project building, [*https://www.youtube.com/watch?v=9DDX3US3kss&t=2319s&ab\_channel=JavaScriptMastery*](https://www.youtube.com/watch?v=9DDX3US3kss&t=2319s&ab_channel=JavaScriptMastery)

[13] Bug fixing, *https://stackoverflow.com/questions/60123984/react-router-pass-props-from-one-component-to-another-using-navlink*

## Списък с използваните означения и съкращения

HTML – Hyper Text Markup Language

CSS - Cascading Style Sheets

PHP – Personal Home Page

IDE – Интегрирана среда за разработка

JSX – Синтаксис, който е разширение на JavaScript

CGI – Computer Graphic Imagery

АPI – Application Programming Interface