

Документация на проект ,,Safedocs”

1. **Тема**

Документация на проект № 195, Категория: Софтуерни приложения

1. **Автори**

* Виктор Йорданов Недев – 0849161426 , адрес: Велико Търново, телефон: 0889533397, email: [viktornedev08@gmail.com](mailto:viktornedev08@gmail.com), училище: ПМГ „ Васил Друмев ” от 10д клас.
* Борис Момчилов Савянов – 0846131460 , адрес: Велико Търново, телефон: 0885212824, email: [bsavyanov@gmail.com](mailto:bsavyanov@gmail.com) , училище: ПМГ „ Васил Друмев ” от 10д клас.

1. **Ръководител**

- Георги Игнатов, старши учител по Информационни технологии и Информатика, е-mаil адрес на ръководителя: [g\_ignatov@mail.bg](mailto:g_ignatov@mail.bg), тел:0889255850

1. **Резюме:**
   1. “Safedocs” е платформа за създаване, управление и удостоверяване на автентичността на сертификати и документи чрез блокчейн технологии. Платформата предлага лесен и сигурен начин за генериране на дигитални сертификати, които могат да бъдат удостоверени в блокчейн мрежата по всяко време. Целевите групи на проекта са образователни институции и компании, които желаят да сигурират автентичността и сигурността на издаваните от тях сертификати и документи.

4.2 Основни етапи в реализацията на проекта

За успешното изпълнение на проекта “Safedocs” бяха изпълнени няколко ключови етапа, които са основоположни за функционалността и продуктивността на платформата. Екипът разпредели силите помежду си, за да постигне високо ниво на качество и ефективност.

Етапи отговорни за Виктор Недев

Виктор Недев беше отговорен за създаването на дизайна на платформата. Неговата работа включваше разработката на удобен и изчистен интерфейс, който да подпомага лесната навигация и разбираемост на функциите на сайта. Неговата роля също така включваше подготвката на презентация, документация и рекламни материали.

Етапи отговорни за Борис Савянов

Борис Савянов се занимаваше с техническите аспекти на платформата, включващи разработката на блокчейн компоненти и интеграцията с Firebase. Той беше отговорен за развитието на функции за генериране, съхранение и удостоверяване на сертификати в блокчейн мрежата. Освен това, Борис разработи и функционалностите за връзка с базата данни, както и създаването на логиката за верификация на сертификати.

4.3 Ниво на сложност на проекта − основни проблеми при реализация на поставените цели

Основни предизвикателства:

* Интеграция на блокчейн и Firebase: Една от основните сложности беше свързването на блокчейн технологията с Firebase. Блокчейнът предоставя децентрализирано и сигурно съхранение на данни, докато Firebase е централизирана база данни в реално време. Координирането на тези две технологии изискваше специални решения за синхронизация и управление на данни.
* Управление на транзакциите в блокчейна: Тъй като всяка операция за издаване на сертификат включва транзакция в блокчейна, трябваше да се справим с въпроси като газ такси, управление на време за потвърждение на транзакции и обработка на евентуални неуспешни транзакции.
* Генериране на PDF сертификати: Генерирането на PDF файлове със сертификати, включително QR кодове за проверка, изискваше интеграция на външни библиотеки и оптимизация за коректно изобразяване на всички детайли.

Решения:

* Асинхронна обработка: Използвахме асинхронни функции за управление на транзакциите в блокчейна, което позволява на потребителите да продължат да използват интерфейса, докато транзакцията се обработва.
* Автоматизирано генериране на PDF: За автоматично генериране на PDF документи с QR кодове използвахме библиотеката pdf-lib, което позволи лесно добавяне на текст и изображения в PDF файловете.

4.4. Логическо и функционално описание на решението

Архитектура на системата:

Системата следва клиент-сървър модел, където клиентската част е реализирана със SvelteKit, а сървърната част използва Firebase и блокчейн контракти в Ethereum. Клиентът изпраща заявки към Firebase за удостоверяване и съхранение на данни. За издаване на сертификати, клиентът комуникира с блокчейн договора чрез MetaMask.

Модули на системата:

* Модул за удостоверяване (Firebase Authentication): Отговаря за влизане на потребителите и управление на потребителските сесии.
* Модул за съхранение на сертификати (Firebase Realtime Database и Firestore): Съхранява част от информацията на издадените сертификати.
* Модул за блокчейн операции (Smart Contract): Управлява издаването на сертификати в блокчейна, подписването на данни и валидацията на сертификати.
* Модул за генериране на сертификати (PDF Generator): Генерира PDF файлове с детайли за сертификатите и вградени QR кодове за лесна верификация.

Взаимодействия между модулите:

* Клиентската част изпраща данни за нов сертификат към блокчейн контракта и Firebase.
* Firebase се използва за съхранение на сертификати в реално време и уведомяване на клиента при актуализация.
* При заявка за верификация, клиентът получава данни от Firebase и проверява подписа чрез блокчейн контракта.

Основни функции на системата:

* Издаване на сертификати: Потребителите могат да издават сертификати чрез блокчейн, след като предоставят необходимите данни.
* Верификация на сертификати: Всеки сертификат може да бъде верифициран чрез сканиране на QR код или директно въвеждане на адреса на сертификата в системата.
* Генериране и изтегляне на сертификати: Генерира се PDF сертификат, който може да бъде изтеглен от потребителите.
  1. – За изработката на проекта използвахме софтуерите:

Visual Studio Code

Photoshop – trial

Git и Github

MetaMask

Ganache – npm модул за виртуализация на eth блокчейн

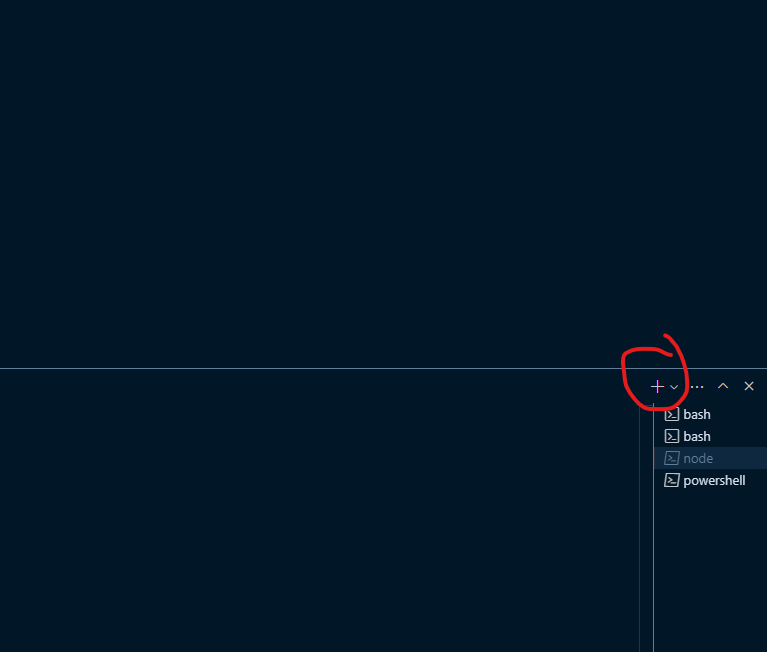
NPM (Node package manager)

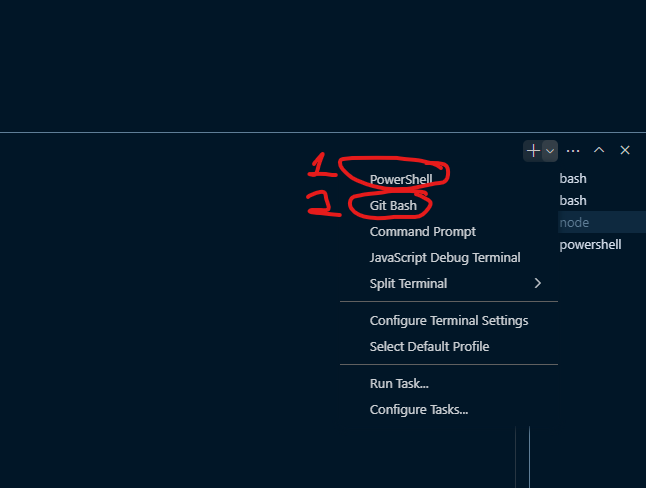
Основните технологии са:

* HTML, CSS, JavaScript, NodeJS, SvelteKit, Firebase и Solidity
  1. Проектът може да се изтегли от ,, <https://github.com/BorisSavianov/Blockchain-certificates>”

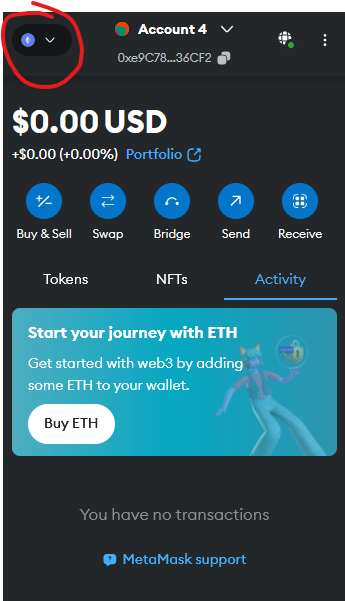
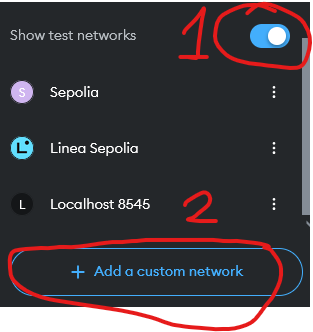
Стъпки за стартиране на проекта:

1. Уверете се че имате инсталирани: Node.js, npm, Visual Sudio, git bash и че имате metamask портфейл.
2. Заредете проектът във Visual Studio.
3. Отворете 2 bash терминала и 1 powershell терминал.

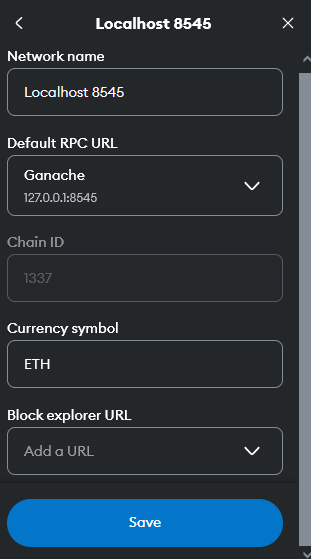
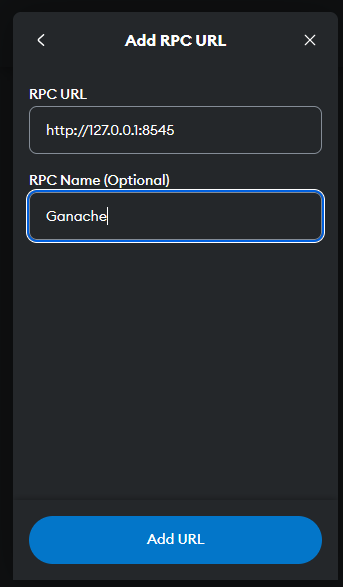




1. Във първият bash терминал напишете: npm install ganache –global –force
2. В същият терминал напишете: ganache
3. Отворете metamask и отидете на добавяне на мрежа и тикнете show test networks и натсинете add custom network.

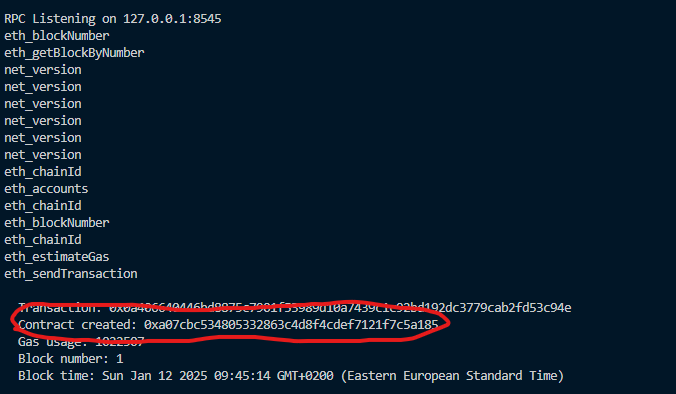
Попълнете бланката по следният начин и натиснете save:

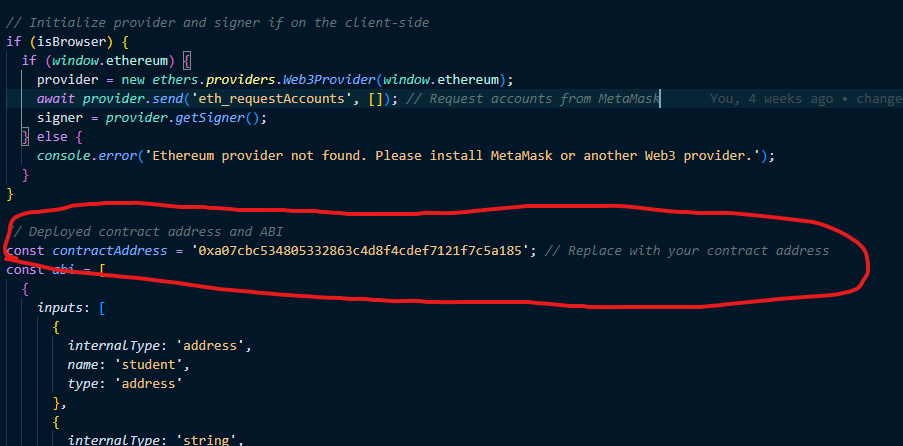
1. Във вторият bash терминал напишете: npx hardhat run scripts/deploy.cjs --network localhost



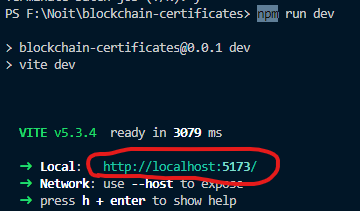
1. Отворете първият bash терминал и копирайте адресът на договора.



И го поставете в src/lib/eth.js

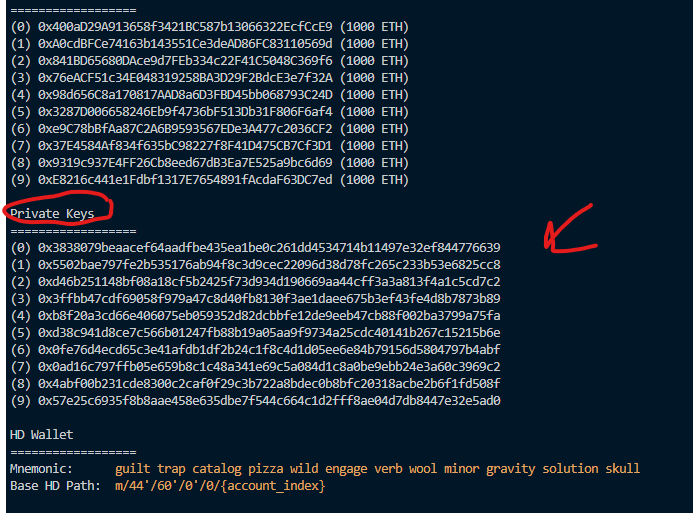


1. След това отворете powershell терминала и напишете: npm run dev

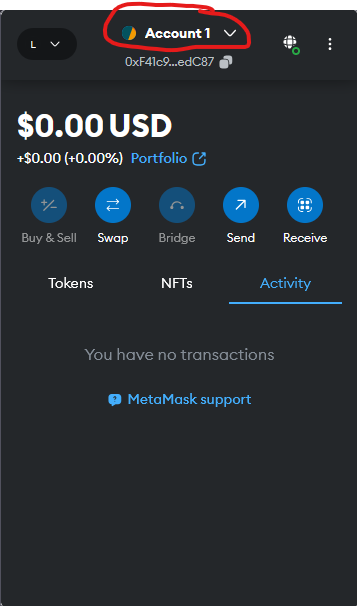
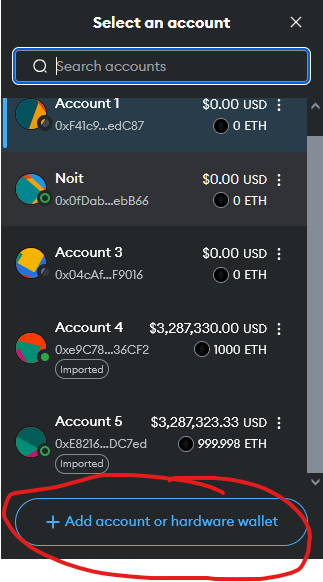


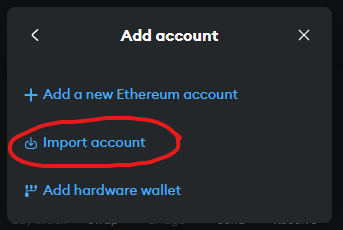
И заредете посоченият линк за платформатa.

1. След като се регистрирате в платформата отворете отново bash терминалът, в който написахте ganache и отидете на секцията private keys и копирайте един от тях

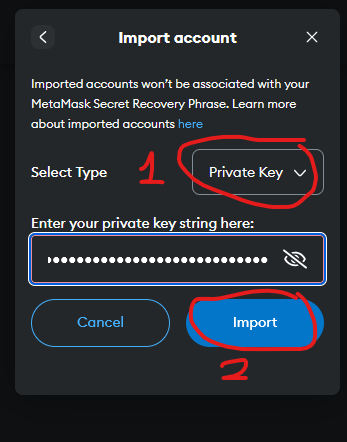


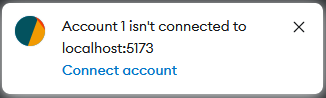
Използвайки някой от тези ключове добавете акаунт в metamask по следният начин:



Уверете се, че сте избрали private key, поставете го и натиснете import:



Ако излезе подобно съобщение, изберете connect account: 

1. Вече сте заредели изцяло платформата на вашето устройство.
   1. Заключение: Уеб приложението предлага иновативен и сигурен подход за управление на сертификати и документи, използвайки предимствата на блокчейн технологиите. То осигурява надеждност, ефективност и улеснение при създаването и удостоверяването на дигитални сертификати и е насочено за гарантиране на безопасността в съвременната дигитална среда.