

Технічне завдання на розробку веб-додатка **SolSpy**

ЧОРНЕТКА

Версія документа: 1.10
Дата: 23 липня 2025

Замовник:

Виконавець:

1. Вступ	3
1.1. Мета проєкту	3
1.2. Цільова аудиторія	3
1.3. Аналоги	3
2. Функціональні вимоги	4
2.1. Основні вимоги	4
2.2. Головна сторінка	4
2.3. Сторінка валідатора	6
2.4. Особистий кабінет	6
2.5. Сторінка порівняння	7
2.6. Адміністративна частина	7
2.7. Текстові сторінки	7
2.8. Додаткові функції	8
3. Технічні вимоги	9
3.1. Frontend	9
3.2. Backend	9
3.3. Інтеграції	9
3.3.1. PRC Solana	9
3.3.2. Локалізація	10
3.3.3. Discord API	10
4. UI/UX дизайн	11
4.1. Основні вимоги	11
4.2. Стиль, кольорова схема та шрифти	11
4.3. Логотип - SolSpy	12
4.4. Карта сайту	13
4.5. Wireframes	13
4.4. Figma-макет	20
5. Розширення	22
5.1. Мобільний додаток	22
5.2. Додаткові інтеграції	22
6. Нефункціональні вимоги	23
7. Умови підтримки та навчання	24
7.1. Технічна підтримка	24
7.2. Контент-підтримка	24
7.2. Навчання	24
8. Кошторис	25
9. Етапи реалізації	26
Додаток 1 - Ризики	27
Додаток 2 - Налаштування хостингу та деплой	28
Додаток 3 - Розрахунок нагород Awards	29
Додаток 4 - Методи і розрахунки	30

A.4.1. Номер поточної епохи, прогресс та час до кінця епохи.....	30
A.4.2. Поточний курс SOL до USD	31
A.4.3. Отримати список валідаторів.....	32
A.4.4. Метадані валідатора	33
TypeScript налаштування:.....	33
Аватар	33
Назва/Ім'я	34
Location (країна) + City	34
Website	35
Скріншот сайту.....	35
Description.....	36
ASN (Autonomous System Number) + IP	36
A.4.5. Identity Key (обрізаний до 4+...+4).....	37
A.4.6. Vote Key та Withdrawer Key	37
A.4.7. Статус (active / delinquent)	38
A.4.8. TVC Score (місце за stake).....	38
A.4.9. Stake Pools (іконки пулів)	39
A.4.10. Inflation Commission.....	39
A.4.11. MEV Commission.....	40
A.4.12. Uptime	41
A.4.13. Client (з version)	41
A.4.14. Статус SFDP	41
A.4.15. Vote Rate.....	42
A.4.16. Jito Score.....	43
A.4.17. Leader Slots.....	43
A.4.17. Time Next Slot	44
A.4.18. Skipped Slots / Produced Slots	44
A.4.19. Account Assets	45
A.4.20. Сервер (CPU, RAM, SSD).....	45
A.4.21. Обчислення Spy Rank	46
A.4.22. Налаштування RPC на ноді в mainnet.....	48

1. Вступ

1.1. Мета проєкту

Створення інформативного, візуально-привабливого та інтерактивного веб-додатка (веб-сайта) для моніторингу актуальної інформації валідаторів Solana.

Користувачі повинні мати змоги фільтрувати інформацію, додавати до обраних, порівнювати, бачити історію змін, сповіщень, стейкінгу (Kiwi) та функції аналітики (графіки). Кожен валідатор може отримувати нагороди (зірочки).

1.2. Цільова аудиторія

- Валідатори
- Інвестори (стейкери)
- Аналітики блокчейн
- Розробники (власне API з JSON-відповідями)

1.3. Аналоги

- <https://stakewiz.com/>
- <https://solanabeach.io/>
- <https://www.validators.app/>
- <https://topvalidators.app/>
- <https://1000x.sh> - параметри валідатора та вибір стовпців;
- <https://marinade.finance/> - графіки, як тут.

2. Функціональні вимоги

2.1. Основні вимоги

- Актуалізація даних кожні - **1 секунд** (або 5 сек, 60 сек, 5 хв, 15 хв, 30 хв)
- Фільтрація валідаторів
- Сортування, змінення порядку стовбців
- Додавання валідатора до обраних (профіль користувача, localStorage)
- Порівняння валідаторів
- Стейкінг (перенаправлення на Kiwi)
- Вказання пулів кожного валідатора та сортування по пулам (від більшого)
- **Pool eligibility checker & validator simulator (stake pool eligibility, requirements, and future tooling to assist validators) - на майбутнє**

2.2. Головна сторінка

- Вибір мережі валідаторів (Mainnet/Testnet)
- Панель поточної епохи (epoch):
 - Epoch
 - Epoch progress
 - Час до кінця (Epoch time remaining)
- Поточний курс SOL/\$
- Іконки:
 - Порівняти обраних валідаторів (без авторизації користувач має змогу порівняти тільки двох - зберігається в localStorage)
 - Додати до обраних (без авторизації користувач може додати до обраних не більше 5 - зберігається в localStorage)
 - Сповіщення сайту (для авторизованих користувачів)
- Кнопка Login / Register (попап-вікно)
- Банер-карусель з карточками топ-10 валідаторів (власний score на основі [Stakewiz](#)) або обраними в адміністративній панелі)
- Банер-карусель з новинами з Discord (API-інтеграція) або із адмінки
- Пошук валідатора за ключем або ім'ям
- Таблиця валідаторів:
 - Колонки (приклад вибору колонок як на сайті [1000x](#)):
 - Checkbox для групових дій (порівняти, додати до обраних, налаштувати сповіщення, додати до blacklist)
 - Spy Rank (додаток A.4.21)

- Аватар (якщо є)
- Назва/Ім'я та Identity Key (образований до 4 символа на початку і 4 символа наприкінці)
- Статус (active/delink)
- TVC Score (місце в поточну епоху)
- Stake Pool (набір пулів у вигляді іконок, як на validators.app)
- Inflation Commission
- MEV Commission
- Uptime
- Client (з version)
- Статус SFDP (none, pending, onboard, retired, rejected)
- Location (country)
- Awards (Додаток 3)
- Vote Rate (як в додатку SolCirl)
- Website
- City
- ASN
- IP
- Jito Score (30 епох - [місце в рангу Jito](#))
- Кнопки дій:
 - Відкрити сторінку валідатора
 - Додати до порівняння
 - Додати до обраних
 - Сповіщення (попап-вікно з таблами Telegram/Email - для незареєстрованих користувачів доступні не всі сповіщення (Inf%, MEV%, Status) - зберігається в localStorage)
 - Stake
 - Заблокувати (не відображати в таблиці)
- Візуальна підсвітка валідатора за певними умовами (наприклад, з адміністративної панелі)
- Опції відображення (множинний вибір):
 - Всі
 - Тільки обрані
 - Тільки ті, що обрані для порівняння
 - Заблоковані
 - Не з рашки (не релізі це треба приховати, але в майбутньому треба мати можливість увімкнути)
 - Тільки з іменем
 - Тільки ті, що мають сайт
 - Тільки ті, що валідують (operated) більше року
 - Тільки ті, що мають inflation commission та MEV com. - 0
- Колонки можна міняти містами (drag-&-drop) - **порядок повинен зберігатись**, додавати або прибирати необхідні (попап-вікно), міняти вигляд виводу даних (строчки або картки).

2.3. Сторінка валідатора

- Вся інформація, що відображається в таблиці на головній сторінці з повним її функціоналом, включно з полями, що не відображаються (якщо можна отримати доступ до певної інформації, наприклад, Uptime). Всюди поточний час користувача сайтом.
 - Сповіщення про зміну (зміна або оновлення)
 - Awards (Додаток 3)
- LeaderSlots, Time Next Slot, Skipped, Produced
- Account Assets (Identity, Vote, Withdrawer)
<https://pro-api.solscan.io/pro-api-docs/v2.0> - як тут
- Мониторинг серверів:
 - Локалізація (ASN Lookup/GeoIP)
 - CPU (якщо доступно)
 - RAM (якщо доступно)
 - SSD (якщо доступно)
 - Хронологія змін (з часом, через пів-року, у вигляді таблиці)
 - Візуалізація на карті
- Графіки (історичні дані за 30 епох):
 - Active Stake (гістограма)
 - Skip Rate (гістограма)
 - Uptime (кольорові кластери)
 - Commission Change (таблиця)

2.4. Особистий кабінет

- Реєстрація/авторизація (Email + 2FA / OAuth / Google, Apple)
- Таби:
 - Список обраних валідаторів у вигляді таблиці (як на головній сторінці з аналогічним функціоналом)
 - Список заблокованих валідаторів (blacklist)
- Налаштування сповіщень (Email/Telegram-бот)
 - Вибір, які сповіщення надсилати у Telegram-бот, а які на Email
 - Збереження Email-адреси
- Додаткові функції
 - Порівняння вибраних валідаторів
 - Експорт даних (CSV/JSON)

2.5. Сторінка порівняння

- Таблиця з порівнянням обраних валідаторів (можливість видалення валідатора з таблиці)
- Повинна бути можливість перетягування валідаторів
- Експорт даних
- Порівняння у вигляді графіків:
 - Uptime (лінійний графік)
 - Skip rate (лінійний графік)
 - Active Stake (лінійний графік)
 - Commission Change (лінійний графік)

2.6. Адміністративна частина

- Перегляд/редагування зареєстрованих користувачів
- Визначення топ-валідаторів (checkbox)
- Визначення яких валідаторів треба підсвічувати (візуально виділяти) в таблиці (checkbox або алгоритм)
- Налаштування сповіщень
 - Рекламні сповіщення в Telegram-бот
- Моніторинг/завантаження новин у карусель
- Статистика відвідувань
- Перегляд логів

2.7. Текстові сторінки

- About Project
- Security
- FAQ (акордеон з питаннями та відповідями)
- API (опис API)
- Contacts

Дані надає замовник.

2.8. Додаткові функції

- Зберігання налаштувань користувача в браузері (localStorage)
- Підтримка мобільної версії (адаптивний дизайн)
- API

3. Технічні вимоги

Розробник сам обирає технології, якими він буде користуватись у розробці. Рекомендовані (не обов'язкові) технології надані у розділі нижче. Розробка Frontend+Backend розпочинається тільки після погодження макету Figma.

3.1. Frontend

- Стек:
 - [React.js](#)
 - [Next.js](#)
 - [Node.js](#)
 - TypeScript
- Графіки:
 - [Rechart](#)
 - [Chart.js](#)
 - D3.js
- Стили:
 - Tailwind CSS
- Адаптивність:
 - Підхід Mobile-first

3.2. Backend

- [Node.js](#) + Express або Python Fast API
- Cron/Worker для періодичного опитування RPC
- Кешування: [Redis](#) або [PostgreSQL](#)
- База даних: SQL або PostgreSQL
- API: REST + WebSocket

3.3. Інтеграції

3.3.1. PRC Solana

Інтеграція з RPC Solana реалізується завдяки запитам до власного RPC замовника, який розташований на окремому спеціально облаштованому під цю задачу сервері. Всі методи

взаємодії з RPC Solana описані в офіційній документації - <https://solana.com/uk/docs/rpc/http>

Загальна інформація:

- getEpochInfo - поточна епоха, слоти
- getBlockHeight - висота блоку
- getVersion - версія Solana
- getLeaderSchedule - розклад лідерів
- getEpochSchedule - довжина епохи
- Сторонні сервіси (дивись додатки).

Інформація про валідатора:

- getVoteAccounts - список валідаторів з комісією, lastVote, rootSlot
- getInflationReward - нагороди валідаторів
- getStakeActivation - активність стейків (активний/неактивний стейк)
- getBlockProduction - активність валідатора
- getIdentity - з блокчейн або Jito
- Сторонні сервіси (дивись додатки).

3.3.2. Локалізація

Геолокація серверів визначається через IP-аналіз або ручне внесення.

- IP валідатора - GeoIP (через сторонні API)
- Визначення чи сервер в рашке (по ASN)

3.3.3. Discord API

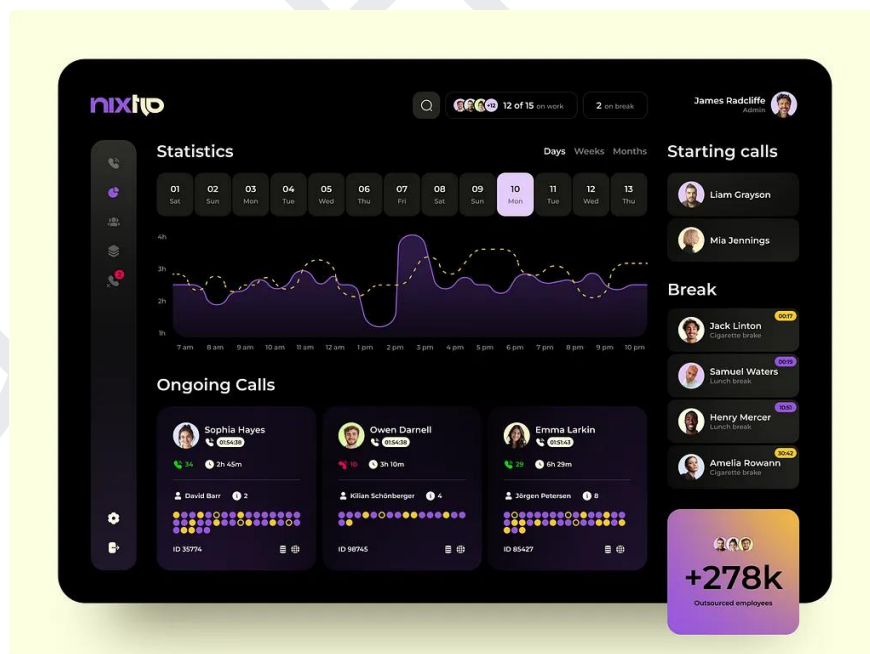
- Парсинг Discord через API або RSS

4. UI/UX дизайн

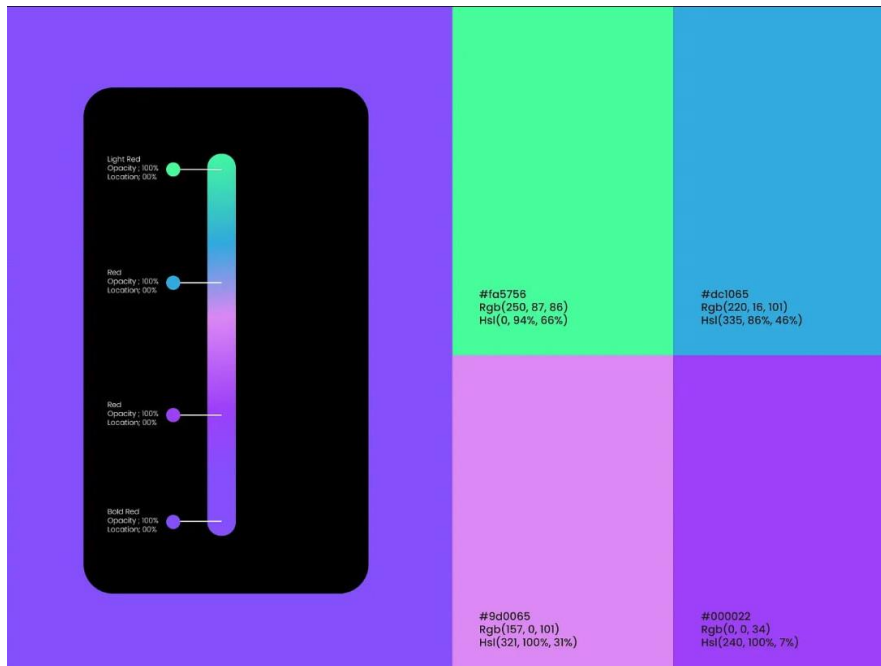
4.1. Основні вимоги

- Інформативний і мінімалістичний дизайн
- Приємна типографія
- Акценти на таблиці та даних
- Темна і світла теми
- Анімації при наведенні
- Банери з ефектом автослайду (карусель)
- Мінімалістичні графіки
- Мобільна адаптивність (Mobile-first)
- Референси з дизайну:
 - <https://stakewiz.com/>
 - <https://solanabeach.io/>
 - <https://topvalidators.app/>

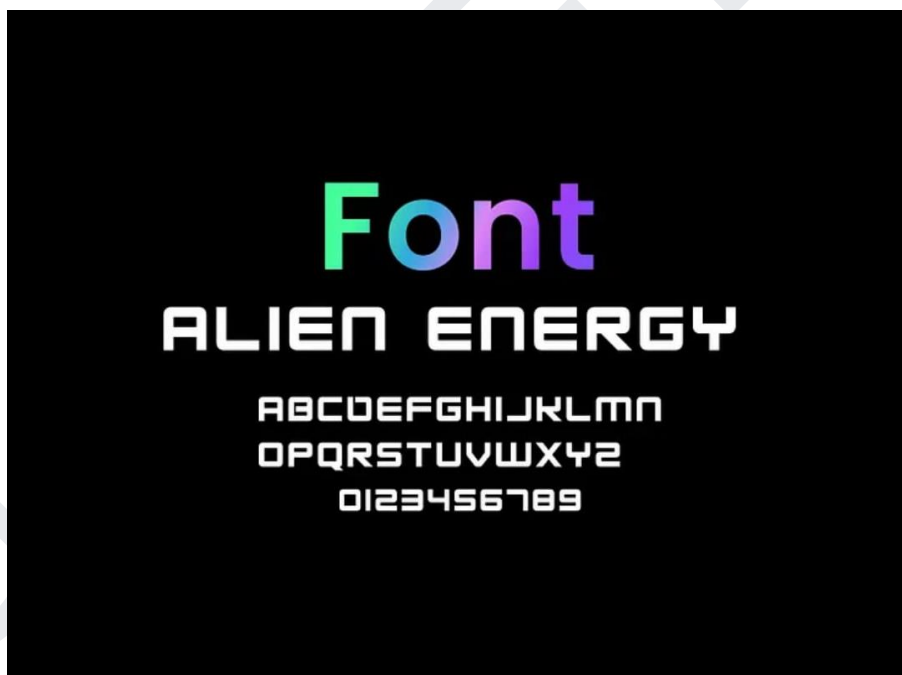
4.2. Стиль, кольорова схема та шрифти



[Dribbble](#)



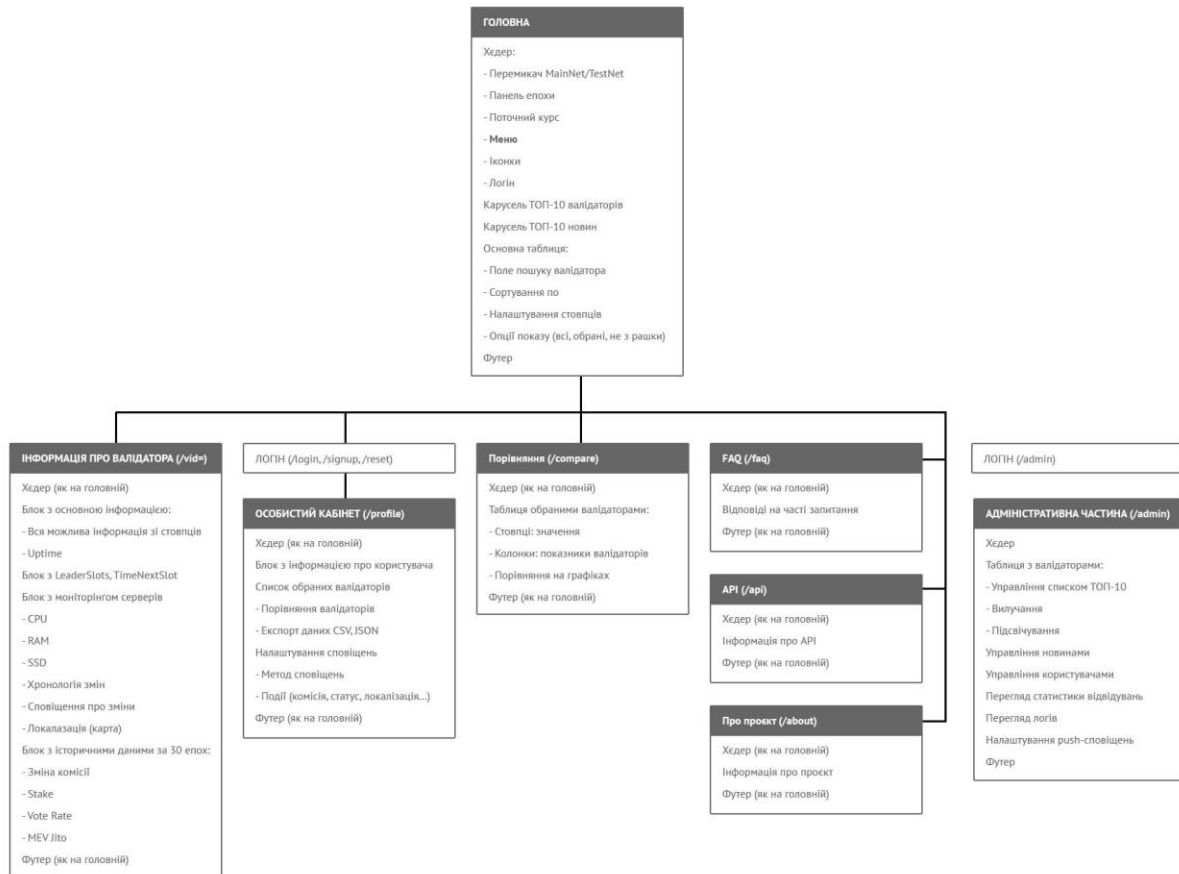
Ярче ніж офіційні кольори Solana



4.3. Логотип - SolSpy

Розробка логотипу обговорюється окремо.

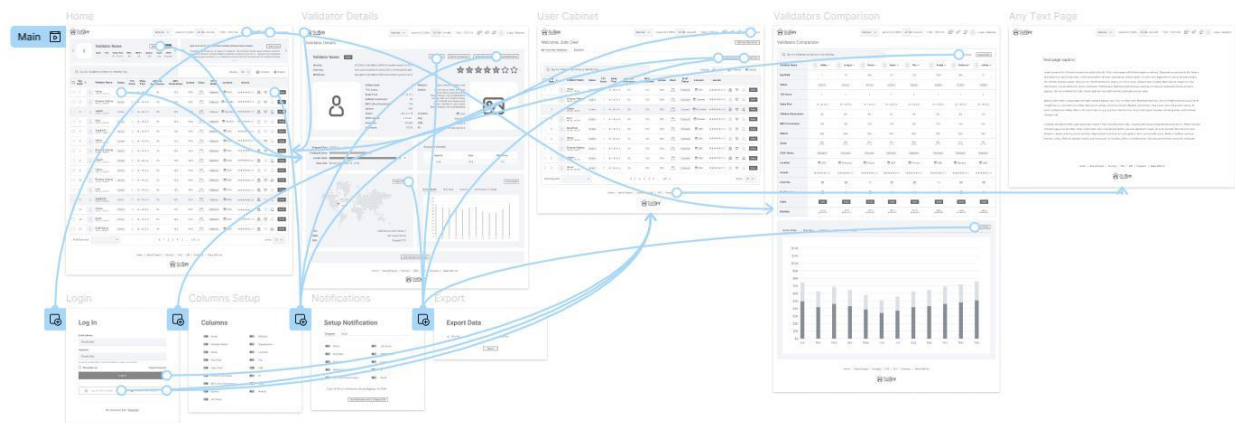
4.4. Карта сайта



4.5. Wireframes


Каркас (wireframe) структури веб-додатку узгоджується окремо перед початком роботи над прототипом веб-додатку у Figma:

- Головна
- Сторінка валідатора
- Сторінка порівняння
- Особистий кабінет
- Текстова сторінка



Wireframes в Figma

4.5.1. Wireframe - Головна сторінка



Mainnet Epoch 813 (28%) 2d 18h 12m left 1 SOL = \$151.53 👤 🔖 🔔 👤 Login / Register

<

👤

Validator Name

Details Stake

Rank

TVC

Stake Pool

INF%

MEV%

Uptime

Client

SFDP

1

1

0%

5%

90%

Jito

Onboard

>

tigarcia 26.06.25, 19:15 @Testnet Validator @Mainnet Beta Validator

Read more

Operators may choose to run Agave or Firedancer. The minimum Testnet Agave software version is 2.3.1. The minimum Testnet Firedancer software version is 0.603.20216. Operators are required to be running Agave version 2.3.1 or Firedancer version 0.603.20216 by the start of Testnet epoch 805.

Q Spy for Validator by Name or Identity Key...

Display: All Columns Output

<input type="checkbox"/>	Spy Rank	Validator Name	Status	TVC Score	Stake Pool	Inflation Commission	MEV Commission	Uptime	Client	SFDP Status	Location	Awards
<input type="checkbox"/>	1	<div><div>👤</div><div>Helius</div><div>HEL1JK...YEe2TU</div></div>	Active	1		0%	10%	90%	Jito 2.2.16	Onboard	🇺🇸 USA	★★★★★☆☆ <div>👤🔖🔔</div> Stake
<input type="checkbox"/>	2	<div><div>👤</div><div>Binance Staking</div><div>HEL1JK...YEe2TU</div></div>	Active	1		0%	10%	90%	Jito 2.2.16	Onboard	🇨🇦 Canada	★★★★★☆☆ <div>👤🔖🔔</div> Stake
<input type="checkbox"/>	3	<div><div>👤</div><div>Jupiter</div><div>HEL1JK...YEe2TU</div></div>	Active	1		0%	10%	90%	Jito 2.2.16	Onboard	🇩🇪 Germany	★★★★★☆☆ <div>👤🔖🔔</div> Stake
<input type="checkbox"/>	4	<div><div>👤</div><div>Klin1</div><div>HEL1JK...YEe2TU</div></div>	Active	1		0%	10%	90%	Jito 2.2.16	Onboard	🇺🇦 Ukraine	★★★★★☆☆ <div>👤🔖🔔</div> Stake
<input type="checkbox"/>	5	<div><div>👤</div><div>Shpillivilli</div><div>HEL1JK...YEe2TU</div></div>	Active	1		0%	10%	90%	Jito 2.2.16	Onboard	🇺🇸 USA	★★★★★☆☆ <div>👤🔖🔔</div> Stake
<input type="checkbox"/>	6	<div><div>👤</div><div>Helius</div><div>HEL1JK...YEe2TU</div></div>	Active	1		0%	10%	90%	Jito 2.2.16	Onboard	🇺🇸 USA	★★★★★☆☆ <div>👤🔖🔔</div> Stake
<input type="checkbox"/>	7	<div><div>👤</div><div>Binance Staking</div><div>HEL1JK...YEe2TU</div></div>	Active	1		0%	10%	90%	Jito 2.2.16	Onboard	🇺🇸 USA	★★★★★☆☆ <div>👤🔖🔔</div> Stake
<input type="checkbox"/>	8	<div><div>👤</div><div>Jupiter</div><div>HEL1JK...YEe2TU</div></div>	Active	1		0%	10%	90%	Jito 2.2.16	Onboard	🇺🇸 USA	★★★★★☆☆ <div>👤🔖🔔</div> Stake
<input type="checkbox"/>	9	<div><div>👤</div><div>Helius</div><div>HEL1JK...YEe2TU</div></div>	Active	1		0%	10%	90%	Jito 2.2.16	Onboard	🇺🇸 USA	★★★★★☆☆ <div>👤🔖🔔</div> Stake
<input type="checkbox"/>	10	<div><div>👤</div><div>Binance Staking</div><div>HEL1JK...YEe2TU</div></div>	Active	1		0%	10%	90%	Jito 2.2.16	Onboard	🇺🇸 USA	★★★★★☆☆ <div>👤🔖🔔</div> Stake
<input type="checkbox"/>	11	<div><div>👤</div><div>Klin1</div><div>HEL1JK...YEe2TU</div></div>	Active	1		0%	10%	90%	Jito 2.2.16	Onboard	🇺🇸 USA	★★★★★☆☆ <div>👤🔖🔔</div> Stake
<input type="checkbox"/>	12	<div><div>👤</div><div>Shpillivilli</div><div>HEL1JK...YEe2TU</div></div>	Active	1		0%	10%	90%	Jito 2.2.16	Onboard	🇺🇸 USA	★★★★★☆☆ <div>👤🔖🔔</div> Stake
<input type="checkbox"/>	13	<div><div>👤</div><div>Zoloto</div><div>HEL1JK...YEe2TU</div></div>	Active	1		0%	10%	90%	Jito 2.2.16	Onboard	🇺🇸 USA	★★★★★☆☆ <div>👤🔖🔔</div> Stake
<input type="checkbox"/>	14	<div><div>👤</div><div>Grunt</div><div>HEL1JK...YEe2TU</div></div>	Active	1		0%	10%	90%	Jito 2.2.16	Onboard	🇺🇸 USA	★★★★★☆☆ <div>👤🔖🔔</div> Stake
<input type="checkbox"/>	15	<div><div>👤</div><div>B2BFinance</div><div>HEL1JK...YEe2TU</div></div>	Active	1		0%	10%	90%	Jito 2.2.16	Onboard	🇺🇸 USA	★★★★★☆☆ <div>👤🔖🔔</div> Stake

With Selected:

▼


< 1 2 3 4 5 ... 100 >





Items: 15 ▼

[Home](#) | [About Project](#) | [Security](#) | [FAQ](#) | [API](#) | [Contacts](#) | [Stake With Us](#)

15

4.5.2. Wireframe - Сторінка валідатора




Mainnet Epoch 813 (28%) 2d 18h 12m left 1 SOL = \$151.53     Login / Register


Validator Details


Validator Name **Stake**


Export Data Add to Comparison Add to Favorites Setup Notifications

Identity: H1SZKAL2odpNBj2oCjffnFGaYwmbGmyewGv1e2TU
Vote Key: he1iusunGwqrNtafDtl.dhsUQDFvo13z9sUa36PauBtk
Withdriver: pduiaKAL2odpNBj2oCjffnFGaYwmbGmyewGv1e2TU



SolSpy Rank: 1
TVC Score: 1
Stake Pool: 
Inflation Commission: 0%
MEV (Jito) Commission: 10%
Uptime: 90%
Client: Jito, 2.2.16
SFDP Status: Onboard
Vote Rate: 99.42%
Jito Score: 97.5%

Website: https://www.validator.com
Details: Lorem ipsum dolor sit amet, consectetur adipiscing elit. Etiam tempor, nisi eu bibendum viverra, sem mi fringilla metus, non blandit arcu massa non massa. Nullam et urna turpis Etiam tempor, nisi eu bibendum
Location:  USA
City: New-York
ASN: AS25673
IP: 37.234.431.212

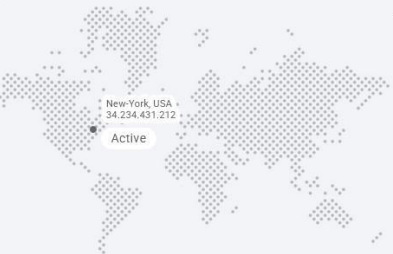


★★★★★

Skipped Slots: 4 14.29%
Produced Slots: 36
Leader Slots: 68 84
Next Slot: 68 min, Mon, July 18, 15:54

Account Assets

Identity	Vote	Withdriver
5.56	0.06	100

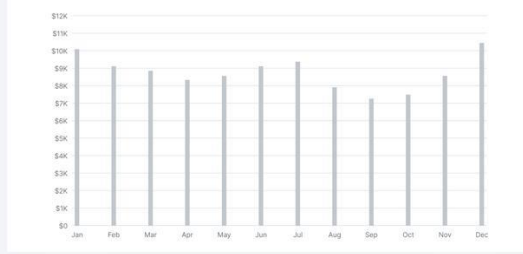


New-York, USA
34.234.431.212
Active

CPU: AMD Ryzen 5000 Series 7
RAM: Samsung 320 Gb
SSD: Seagate 2Tb

Export Data


Active Stake Skip Rate Uptime Commission Change



Export Data

Add Validator to Blacklist

4.5.3. Wireframe - Особистий кабінет







Mainnet

Epoch 813 (28%)

2d 18h 12m left

1 SOL = \$151.53



Logout

Welcome, John Doe!














































My Favorites Validators

Blacklist

Setup Notifications

Export Data

Display: AllColumnsOutput

<input type="checkbox"/>	Spy Rank	Validator Name	Status	TVC Score	Stake Pool	Inflation Commission	MEV Commission	Uptime	Client	SFDP Status	Location	Awards
<input type="checkbox"/>	1	 Helius HEL1JK...YEe2TU	Active	1		0%	10%	90%	Jito 2.2.16	Onboard	USA	★★★★★☆☆    Stake
<input type="checkbox"/>	2	 Binance Staking HEL1JK...YEe2TU	Active	1		0%	10%	90%	Jito 2.2.16	Onboard	Canada	★★★★★☆☆    Stake
<input type="checkbox"/>	3	 Jupiter HEL1JK...YEe2TU	Active	1		0%	10%	90%	Jito 2.2.16	Onboard	Germany	★★★★★☆☆    Stake
<input type="checkbox"/>	4	 Klin1 HEL1JK...YEe2TU	Active	1		0%	10%	90%	Jito 2.2.16	Onboard	Ukraine	★★★★★☆☆    Stake
<input type="checkbox"/>	5	 Shpillivilli HEL1JK...YEe2TU	Active	1		0%	10%	90%	Jito 2.2.16	Onboard	USA	★★★★★☆☆    Stake
<input type="checkbox"/>	6	 Helius HEL1JK...YEe2TU	Active	1		0%	10%	90%	Jito 2.2.16	Onboard	USA	★★★★★☆☆    Stake
<input type="checkbox"/>	7	 Binance Staking HEL1JK...YEe2TU	Active	1		0%	10%	90%	Jito 2.2.16	Onboard	USA	★★★★★☆☆    Stake
<input type="checkbox"/>	8	 Jupiter HEL1JK...YEe2TU	Active	1		0%	10%	90%	Jito 2.2.16	Onboard	USA	★★★★★☆☆    Stake
<input type="checkbox"/>	9	 Helius HEL1JK...YEe2TU	Active	1		0%	10%	90%	Jito 2.2.16	Onboard	USA	★★★★★☆☆    Stake

With Selected:

< 1 2 3 4 5 ... 100 >

Items: 15

[Home](#) | [About Project](#) | [Security](#) | [FAQ](#) | [API](#) | [Contacts](#) | [Stake With Us](#)



405

4.5.4. Wireframe - Сторінка порівняння

SolSpy

MainnetEpoch 813 (28%)2d 18h 12m left1 SOL = \$151.53Login / Register

Validators Comparison

Q Spy for Validator by Name or Identity Key...

RowsExport Data

Validator Name	Helius ×	Longus ×	Penis ×	Basis ×	Vita ×	Avada ×	Kedavra ×	Helius ×
Spy Rank	1	14	586	18	135	1024	586	21
Status	Active	Active	Active	Active	Active	Active	Active	Active
TVC Score	1	1	1	1	1	1	1	1
Stake Pool	▲▲▲▲▲	▲▲▲▲▲	▲▲▲▲▲	▲▲▲▲▲	▲▲▲▲▲	▲▲▲▲▲	▲▲▲▲▲	▲▲▲▲▲
Inflation Commission	0%	0%	0%	0%	0%	0%	0%	0%
MEV Commission	10%	10%	10%	10%	10%	10%	10%	10%
Uptime	90%	90%	90%	90%	90%	90%	90%	90%
Client	Jito 2.2.16	Jito 2.2.16	Jito 2.2.16	Jito 2.2.16	Jito 2.2.16	Jito 2.2.16	Jito 2.2.16	Jito 2.2.16
SFDP Status	Onboard	Onboard	Onboard	Onboard	Onboard	Onboard	Onboard	Onboard
Location	USA	Germany	Poland	USA	Ukraine	USA	Canada	USA
Awards	★★★★☆	★★★★☆	★★★★☆	★★★★☆	★★★★☆	★★★★☆	★★★★☆	★★★★☆
Favorites	♡	♡	♡	♡	♡	♡	♡	♡
Notifications	🔔	🔔	🔔	🔔	🔔	🔔	🔔	🔔
Stake	Stake	Stake	Stake	Stake	Stake	Stake	Stake	Stake
Blacklist	Add to BLACKLIST	Add to BLACKLIST	Add to BLACKLIST	Add to BLACKLIST	Add to BLACKLIST	Add to BLACKLIST	Add to BLACKLIST	Add to BLACKLIST

Active StakeSkip RateUptimeCommission Change

Export Data

Month	Active Stake (\$K)	Skip Rate (\$K)
Jan	5.0	2.5
Feb	4.2	2.0
Mar	4.8	2.2
Apr	4.5	2.0
May	3.8	2.0
Jun	3.5	1.5
Jul	3.8	1.8
Aug	4.2	2.0
Sep	4.5	2.0
Oct	4.8	2.2
Nov	4.8	2.2
Dec	5.0	2.5

Home | About Project | Security | FAQ | API | Contacts | Stake With Us

SolSpy

4.5.2. Wireframe - Текстова сторінка



Mainnet

Epoch 813 (28%)

2d 18h 12m left

1 SOL = \$151.53



Login / Register

Text page caption

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Proin scelerisque sollicitudin magna ac ultrices. Suspendisse eget purus elit. Mauris id aliquam est, eget luctus urna. Lorem ipsum dolor sit amet, consectetur adipiscing elit. Ut enim velit, dignissim at mauris vel, porta mollis leo. Aenean id porta sapien, vel porta leo. Sed id elementum magna, et viverra lacus. Aliquam erat volutpat. Nam ultrices neque nec odio ullamcorper, non condimentum ipsum commodo. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Sed ac condimentum ante. Donec eget orci non nibh facilisis commodo nec non urna.

Mauris nulla lorem, congue eget odio eget, sodales dapibus nisi. Cras vel dolor sem. Maecenas facilisis, felis et fringilla euismod, turpis ante fringilla leo, ac commodo eros tellus vitae ipsum. Integer eu lectus vel nunc dapibus ullamcorper vitae a erat. Cras id dignissim purus, sit amet condimentum tellus. Mauris elit nisl, ornare id ex at, porttitor lobortis dolor. Fusce ante ligula, faucibus vel elit gravida, condimentum rhoncus felis.

Curabitur nec blandit diam, quis accumsan mauris. Etiam vitae dignissim odio. In porta justo turpis, id blandit lectus porta non. Etiam volutpat volutpat augue non faucibus. Etiam sollicitudin, dolor vitae aliquet finibus, ante leo dignissim massa, sit amet suscipit felis tortor id justo. Aliquam a neque at lectus viverra porttitor. Mauris lacinia tincidunt est sed egestas. Nunc sed convallis justo. Morbi at eleifend nisl, non interdum purus. Nulla ac dapibus mauris, a ultrices quam. In faucibus, tellus ac condimentum ultricies, purus metus varius elit, eu feugiat

[Home](#) | [About Project](#) | [Security](#) | [FAQ](#) | [API](#) | [Contacts](#) | [Stake With Us](#)

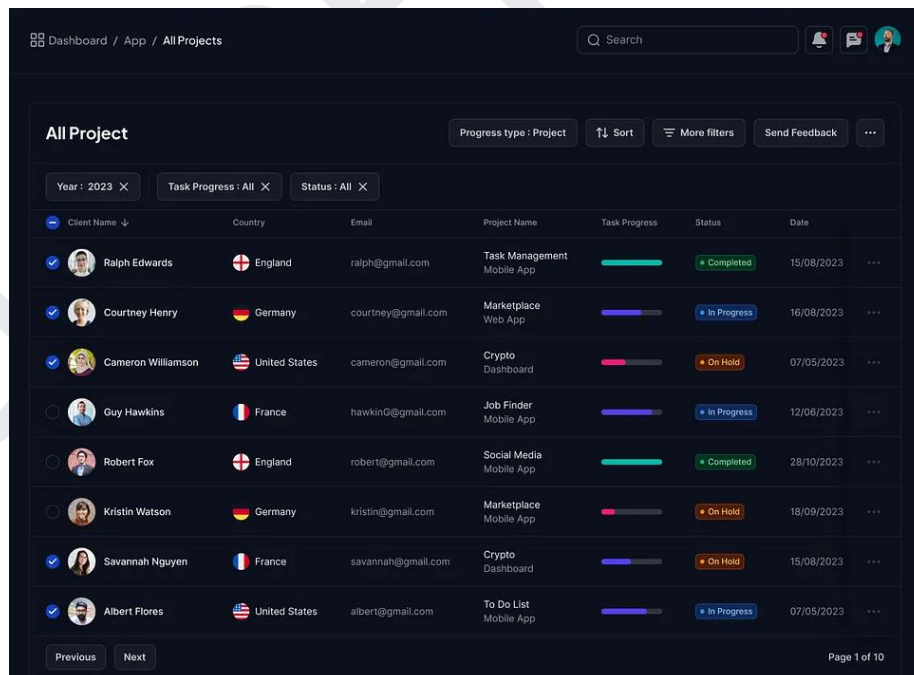


4.4. Figma-макет

Референси щодо дизайну та елементів:



Сторінка на [Dribbble](#).



Сторінка на [Dribbble](#).



Сторінка на [Dribbble](#).



Сторінка на [Dribbble](#).

Тут будуть зображення узгодженого Figma-макету з посиланням на макет.

5. Розширення

5.1. Мобільний додаток

Це окремий проєкт на майбутнє..

- React Native або Flutter
- Повторює ключову функціональність:
 - Таблиця, пошук, сторінка валідатора і т.і.
 - Обрані валідатори, порівняння тощо
 - Push-нотифікації
- Підтримка iOS/Android
- Отримання даних через єдиний Backend

5.2. Додаткові інтеграції

- Telegram-бот (Python) - на першому етапі просто отримує сповіщення (як у Stakewiz)
- API для зовнішнього користування (JSON)

5.2.1. Telegram-бот

Телеграм-бот призначений для отримання обраних користувачем сповіщень з сайту про зміну конфігурації обраних валідаторів. Відписка від сповіщень реалізується через бот (кнопка "Unsubscribe for this notifications").

Вимоги:

- Python
- Сервер для розміщення бота (Heroku, DigitalOcean, AWS, Google Cloud або хостинг сайту)

6. Нефункціональні вимоги

- Швидкість завантаження (до 2 - 5 секунд)
- Мінімальна затримка оновлень (до 10 - 30 секунд)
- Push-нотифікації для мобільного додатка (Firebase)

ЧОРНЕТКА

7. Умови підтримки та навчання

7.1. Технічна підтримка

- Гарантія технічної підтримки - 6 місяців з моменту здачі проекту (1-3 години на тиждень). У підтримку входить виправлення багів і помилок роботи веб-додатку згідно узгодженого технічного завдання.
- Оновлення коду веб-додатка або додавання нового функціоналу виконується за додатковою угодою (окреме технічне завдання, окремі терміни, окрема оплата).
- Написання технічної документації.
- Засоби комунікації:
 - Telegram
 - Email
 - GoogleMeet

7.2. Контент-підтримка

Підтримка контенту виконується менеджером, який слідкує за правильним функціонуванням сайту, статистикою та наповнює/змінює новини.

7.2. Навчання

- Написання короткого мануалу (PDF/Docx)
- Онлайн-навчання через (GoogleMeet) - 2 години
- Якщо необхідно відеоінструкції (YouTube/GoogleDrive)

8. Кошторис

Орієнтовно.

#	Назва роботи	Години	Ставка \$	Сума \$
1	Технічне завдання + UI/UX дизайн	40		
2	Frontend розробка	150		
3	Backend розробка	100		
4	Налаштування RPC сервера	12		
5	Інтеграція з RPC + логіка	80		
6	Налаштування сервера та деплой	8		
7	Реалізація мобільного додатку	-		
8	Тестування	50		
9	Технічна підтримка *	72		
10	Контент-менеджмент	-		
11	Навчання	8		
12	Резерв (домен, хостинг та інше)	-	-	500
	Разом	520		

* В залежності від потреби

9. Етапи реалізації

Орієнтовно.

#	Етап	Тривалість (робочих днів)
1	Підготовка дизайну	7-10
2	Розробка Backend + Frontend	30-35
3	Налаштування RPC сервера	1-2
4	Інтеграція з RPC + логіка	7
5	Налаштування сервера та деплой	1-2
6	Розробка мобільного додатку	
7	Тестування	3-5
8	Навчання персоналу	1
	Загалом	50-62

Реліз - 2.5 місяці.

Повний реліз - 4 місяці.

Додаток 1 - Ризики

1. Нестабільність RPC Solana - необхідність резервних серверів.
2. API Discord може змінити формат.
3. Блокування IP для визначення геолокації серверів.
4. Форс-мажори у зв'язку з війною в Україні.
5. Використання сторонніх сервісів.

ЧОРНЕТКА

Додаток 2 - Налаштування хостингу та деплой

- Хостинг та домен надає замовник (а також логін/пароль)
- Бажані платформи:
 - Vercel
 - Digital Ocean
 - AWS
 - GoDaddy
 - або інші за рекомендацією розробника
- Протоколи:
 - SSH
 - FTP
 - HTTPS
 - SSL

Додаток 3 - Розрахунок нагород Awards

Зірочки нараховуються за:

- 1** — Skipped Slots Score – якщо 5 епох немає пропуску слотів, тоді зірочка
- 2** — Software Version Score – якщо версія клієнта $>$ або $=$ мінімальній дозволений версії від SFDP <https://discord.com/channels/428295358100013066/895740485140906054>
- 3** — Uptime Score – якщо відсоток аптайму за 30 епох більше або дорівнює 98%
- 4** — Inflation Commission Score – якщо комісія валідатора 0% (якщо валідатор збільшив її і потім повернув на нуль то зірочка не зараховується впродовж пів року)
- 5** — MEV Commission Score – якщо комісія валідатора 0% (якщо валідатор збільшив її і потім повернув на нуль то зірочка не зараховується впродовж пів року)
- 6** — Published Information Score – бали за наповнення інформацією (імя, про ноду, картинка)
- 7** — Site Score – бали за нормальний сайт (це зірочка отримується тільки за нашою згодою. Валідатор має написати нам на пошту з проханням перевірити його сайт, і якщо ми визнаємо його сайт достатньо інформативним для валідатора, ми тоді даємо йому цю зірочку)

На сторінці валідатора при наведенні на зірку повинно висвічуватись пояснення.

Додаток 4 - Методи і розрахунки

А.4.1. Номер поточної епохи, прогрес та час до кінця епохи

Метод [getEpochInfo RPC](#):

```
curl https://api.devnet.solana.com -s -X \
  POST -H "Content-Type: application/json" -d '{
    {
      "jsonrpc": "2.0",
      "id": 1,
      "method": "getEpochInfo",
      "params": [
        {
          "commitment": "finalized"
        }
      ]
    }
  },
```

Повертає:

```
{
  "jsonrpc": "2.0",
  "result": {
    "absoluteSlot": 395530735,
    "blockHeight": 383485388,
    "epoch": 915,
    "slotIndex": 250735,
    "slotsInEpoch": 432000,
    "transactionCount": 16142346412
  },
  "id": 1
}
```

epoch: номер поточної епохи

slotIndex: номер поточного слота в епосі

slotsInEpoch: загальна кількість слотів у цій епоці
absoluteSlot: абсолютний номер слота

Прогрес в процентах:

```
const progressPercent = (slotIndex / slotsInEpoch) * 100;  
console.log(`Прогрес епохи: ${progressPercent.toFixed(2)}%`);
```

Час до кінця епохи (середня тривалість одного слота ~ 400 мс):

```
const progress = slotIndex / slotsInEpoch;  
const slotsLeft = slotsInEpoch - slotIndex;  
const timeLeftSeconds = slotsLeft * 0.4; // час до кінця епохи в секундах  
  
const days = Math.floor(timeLeftSeconds / (24 * 3600));  
const hours = Math.floor((timeLeftSeconds % (24 * 3600)) / 3600);  
const minutes = Math.floor((timeLeftSeconds % 3600) / 60);  
const seconds = Math.floor(timeLeftSeconds % 60);  
  
console.log(`Залишилось: ${days} дн, ${hours} год, ${minutes} хв,  
${seconds} сек`);
```

A.4.2. Поточний курс SOL до USD

Метод:

```
curl -s  
"https://api.coingecko.com/api/v3/simple/price?ids=solana&vs_currencies=us  
d"
```

Повертає:

```
{  
  "solana": {  
    "usd": 123.45  
  }  
}
```


А.4.3. Отримати список валідаторів

Метод: [getVoteAccounts RPC](#)

В майн-нет:

```
curl https://api.mainnet-beta.solana.com -s -X POST -H "Content-Type: application/json" -d '{
  "jsonrpc": "2.0",
  "id": 1,
  "method": "getVoteAccounts"
}'
```

В тест-нет:

```
curl https://api.testnet.solana.com -s -X POST -H "Content-Type: application/json" -d '{
  "jsonrpc": "2.0",
  "id": 1,
  "method": "getVoteAccounts"
}'
```

Відповідь містить два масиви:

- **current** — валідатори, які зараз активні та голосують.
- **delinquent** — валідатори, які не голосують (відстали або неактивні).

Кожен елемент містить:

```
{
  "votePubkey": "Vote111...",
  "nodePubkey": "Node111...",
  "activatedStake": 1234567890,
  "commission": 10,
  "epochVoteAccount": true,
  "epochCredits": [
    [123, 4567, 4000],
    [124, 5000, 4567]
  ],
  "lastVote": 12345678,
  "rootSlot": 12345000,
```

```
"credits": 789012
}
```

- `votePubkey` — публічний ключ `vote account` валідатора (`vote key`).
- `nodePubkey` — публічний ключ вузла валідатора (`validator identity key`).
- `activatedStake` — кількість активного стейку (у лампортах).
- `commission` — комісія валідатора у відсотках (0–100).
- `epochVoteAccount` — чи є акаунт активним у поточній епосі (`true/false`).
- `epochCredits` — масив, що містить інформацію про кількість голосів у кожній епосі: `[epochNumber, credits, previousCredits]`
- `lastVote` — номер останнього слота, за який валідатор проголосував.
- `rootSlot` — останній `root slot`, який повідомив валідатор (може бути `null`).
- `credits` — загальна кількість кредитів, зароблених цим акаунтом (відображає активність голосування).

А.4.4. Метадані валідатора

TypeScript налаштування:

```
import { Connection, PublicKey } from "@solana/web3.js";
const connection = new Connection("https://api.mainnet-beta.solana.com", "confirmed");
```

Аватар

Джерело: <https://raw.githubusercontent.com/solana-labs/validators/main/validators/mainnet/validators.json>

Метод: GitHub JSON конфіг

```
const validatorAvatar = `https://raw.githubusercontent.com/solana-labs/validators/main/validators/mainnet/validators.json` // поле "avatarUrl"
```

Альтернатива:

`https://raw.githubusercontent.com/solana-labs/validators/main/validators/mainnet/avatars/<identity>.png`

Назва/Ім'я

Джерело: `getVoteAccounts` або `validators.app` GitHub JSON

Метод (`nodePubkey` → metadata mapping):

```
const voteAccounts = await connection.getVoteAccounts();
console.log(voteAccounts.current[0]?.nodePubkey); // identity
```

Потім - назву по мапінгу з GitHub-JSON.

Альтернатива:

```
curl -X POST https://api.mainnet-beta.solana.com \
-H "Content-Type: application/json" \
-d '{
  "jsonrpc": "2.0",
  "id": 1,
  "method": "getValidatorInfo"
}'
```

Поле: `.result[].info.name`

Location (країна) + City

Метод: на основі IP (`getClusterNodes`) → IP → GeoIP API (наприклад, `ipinfo.io` або `maxmind`)

```
const ip = nodes[0].gossip;
fetch(`https://ipinfo.io/${ip}?token=YOUR_TOKEN`)
```

Альтернатива:

Без RPC дізнатись через IP-геолокацію з `getClusterNodes`

```
# витягуємо IP з `gossip` або `tpu`

curl ipinfo.io/<ip>?token=<your_api_token>
# or
curl http://ip-api.com/json/<ip>
```

Website

Метод: RPC getValidatorInfo

```
const validatorInfo = await connection.getValidatorInfo();
console.log(validatorInfo[0]?.info.website);
```

Альтернатива:

Поле в `getValidatorInfo().result[].info.url` або `website`

Скріншот сайта

Метод: TS Puppeteer

```
npm install puppeteer
npm install --save-dev @types/node

import puppeteer from 'puppeteer';
import fs from 'fs/promises';

async function takeScreenshotWithTypeScript(url: string, outputPath:
string): Promise<void> {
  const browser = await puppeteer.launch();
  const page = await browser.newPage();
  await page.setViewport({ width: 1280, height: 800 });
  await page.goto(url, { waitUntil: 'networkidle2' });

  await page.screenshot({
    path: outputPath,
    fullPage: true,
    type: 'png',
  });

  await browser.close();
}

takeScreenshotWithTypeScript('https://solana.com',
'./screenshots/solana.png')
  .then(() => console.log('Скріншот зроблено!'))
  .catch(err => console.error('Помилка:', err));
```

Альтернатива - Playwright:

```
import { chromium } from 'playwright';
import fs from 'fs/promises';

async function captureWithPlaywright(url: string, path: string) {
  const browser = await chromium.launch();
  const page = await browser.newPage();
  await page.setViewportSize({ width: 1280, height: 800 });
  await page.goto(url);
  await page.screenshot({ path, fullPage: true });
  await browser.close();
}

captureWithPlaywright('https://solana.com', './screenshots/solana-playwright.png');
```

Description

Метод: RPC getValidatorInfo

```
console.log(validatorInfo[0]?.info.details);
```

ASN (Autonomous System Number) + IP

Метод:

- IP: через RPC getClusterNodes()
- ASN: через сторонний API (<https://ipinfo.io> або <https://ip-api.com>)

```
const ip = nodes[0].gossip;
fetch(`http://ip-api.com/json/${ip}?fields=as`)
```

Альтернатива:

- IP: .gossip з getClusterNodes
- ASN: через IP API: `curl http://ip-api.com/json/<ip>?fields=as`

A.4.5. Identity Key (обрізаний до 4+...+4)

Метод: [RPC getVoteAccounts](#)

```
const identityKey = validator.nodePubkey;
const shortIdentity = identityKey.slice(0, 4) + "..." +
identityKey.slice(-4);
```

A.4.6. Vote Key та Withdrawer Key

Метод: [RPC getVoteAccounts](#)

```
const voteAccounts = await connection.getVoteAccounts();
const voteAccount = voteAccounts.current.find(v => v.nodePubkey ===
identity);
console.log(voteAccount.votePubkey, voteAccount.withdrawer); // vote key +
withdrawer
```

Альтернатива:

```
curl -X POST https://api.mainnet-beta.solana.com \
-H "Content-Type: application/json" \
-d '{
  "jsonrpc": "2.0",
  "id": 1,
  "method": "getVoteAccounts"
}'
```

Поля: `.result.current[]` | `votePubkey`, `withdrawer`, `nodePubkey`

A.4.7. Статус (active / delinquent)

Метод: [RPC getVoteAccounts](#)

```
const activeValidators = voteAccounts.current;  
const delinquentValidators = voteAccounts.delinquent;
```

Це дві частини: `.current[]` (active) та `.delinquent[]` із `getVoteAccounts`.

```
# якщо є у .current → active  
# якщо .delinquent → delinquent
```

A.4.8. TVC Score (місце за stake)

Метод:

- Список валідаторів через `getVoteAccounts`
- Обчислюється рейтинг на основі `activatedStake`

```
const sorted = [...voteAccounts.current].sort((a, b) =>  
  Number(b.activatedStake) - Number(a.activatedStake));
```

Альтернатива:

```
# Отримати весь список активних валідаторів:  
curl -X POST https://api.mainnet-beta.solana.com \  
-H "Content-Type: application/json" \  
-d '{  
  "jsonrpc": "2.0",  
  "id": 1,  
  "method": "getVoteAccounts"  
}'
```

#Ранжування за `activatedStake`

```
const sorted = result.current.sort(  
  (a, b) => Number(b.activatedStake) - Number(a.activatedStake)  
);  
const tvcRank = sorted.findIndex(v => v.votePubkey === yourVoteKey) + 1;
```

A.4.9. Stake Pools (іконки пулів)

Через RPC це не видно напряму.

Потрібні власні іконки.

Джерело:

- https://api.validators.app/api/v1/* (сторонній)
- Solana Foundation delegation program registry

Альтернатива (<https://validators.app/api/v1>):

`https://api.validators.app/api/v1/validators/<voteKey>`

A.4.10. Inflation Commission

Метод: [RPC getVoteAccount](#)

```
curl https://api.devnet.solana.com -s -X \
  POST -H "Content-Type: application/json" -d '
  {
    "jsonrpc": "2.0",
    "id": 1,
    "method": "getVoteAccounts",
    "params": [
      {
        "commitment": "finalized",
        "votePubkey": "i7NyKBMJCA9bLM2nsGyAGCKHECuR2L5eh4GqFciuwNT"
      }
    ]
  }
  ,
```

Відповідь:

```
{
  "jsonrpc": "2.0",
  "result": {
```



```

"current": [
  {
    "activatedStake": 38263229364446900,
    "commission": 95,
    "epochCredits": [
      [902, 1383125544, 1376213656],
      [903, 1390037304, 1383125544],
      [904, 1396949288, 1390037304],
      [905, 1403861272, 1396949288],
      [906, 1406766600, 1403861272]
    ],
    "epochVoteAccount": true,
    "lastVote": 391573587,
    "nodePubkey": "dv2eQHeP4RFrJZ6UeiZWoc3XTtmtZCUKxxCApCDcRNV",
    "rootSlot": 391573556,
    "votePubkey": "i7NyKBMJCA9bLM2nsGyAGCKHECuR2L5eh4GqFciuwNT"
  }
],
"delinquent": []
},
"id": 1
}

```

Альтернатива:

```

const voteAccount = await connection.getVoteAccounts();
console.log(voteAccount.current[0]?.commission);

```

A.4.11. MEV Commission

Джерело: Jito API (наприклад, <https://api.jito.network> або Jito Dashboard GraphQL)

A.4.12. Uptime

Непрямо через skipped slots / produced slots:

- збираються дані через getConfirmedBlocks

Альтернатива: сторонні сервіси: solanabeach.io API

A.4.13. Client (з version)

Метод: [RPC getClusterNodes](#)

```
const nodes = await connection.getClusterNodes();  
console.log(nodes[0].version); // наприклад solana-core 1.16.18
```

Альтернатива:

```
curl -X POST https://api.mainnet-beta.solana.com \  
-H "Content-Type: application/json" \  
-d '{  
  "jsonrpc": "2.0",  
  "id": 1,  
  "method": "getClusterNodes"  
}'
```

Поле: .version

A.4.14. Статус SFDP

Джерело: [Foundation SFP list](#) (CSV мапиться “вручну”)

Альтернатива: з GitHub або CSV файлу з Solana Foundation (<https://github.com/solana-foundation/validator-keypairs>)

A.4.15. Vote Rate

Потрібно рахувати вручну: кількість голосів від валідатора / кількість лідер-слотів за епоху.

Приклад на TS (Node.js + fetch):

```
const fetch = require("node-fetch");

async function getVoteRate(votePubkey, identityPubkey) {
  const rpcUrl = "https://api.testnet.solana.com";

  // Отримуємо голоси
  const voteResp = await fetch(rpcUrl, {
    method: "POST",
    headers: { "Content-Type": "application/json" },
    body: JSON.stringify({
      jsonrpc: "2.0",
      id: 1,
      method: "getVoteAccounts"
    })
  });
  const voteData = await voteResp.json();
  const voteAccount = voteData.result.current.find(v => v.votePubkey ===
votePubkey);
  const [ , current, previous ] = voteAccount.epochCredits.slice(-1)[0];
  const voteCount = current - previous;

  // Отримуємо лідер-слоти
  const leaderResp = await fetch(rpcUrl, {
    method: "POST",
    headers: { "Content-Type": "application/json" },
    body: JSON.stringify({
      jsonrpc: "2.0",
      id: 1,
      method: "getLeaderSchedule",
      params: [null, { identity: identityPubkey }]
    })
  });
  const leaderData = await leaderResp.json();
  const leaderSlots = leaderData.result[identityPubkey].length;
```

```

// Обчислюємо
const voteRate = voteCount / leaderSlots;
console.log(`Vote Rate: ${ (voteRate * 100).toFixed(2)}%`);
}

getVoteRate("G1z...ABC", "9xy...XYZ");

```

A.4.16. Jito Score

Витягується через Jito API (<https://jito.network/>) або Jito Validators Leaderboard

Альтернатива: GraphQL - leaderboard за 30 epoch

A.4.17. Leader Slots

Щоб отримати всі, треба обходити:

```
connection.getLeaderSchedule(undefined, identityKey)
```

Альтернатива:

```

curl -X POST https://api.mainnet-beta.solana.com \
-H "Content-Type: application/json" \
-d '{
  "jsonrpc": "2.0",
  "id": 1,
  "method": "getLeaderSchedule",
  "params": [ null, { "identity": "<validator_identity>" } ]
}'

```

A.4.17. Time Next Slot

Обчислюється на базі поточного слоту:

```
const currentSlot = await connection.getSlot();
const blockTime = await connection.getBlockTime(currentSlot);
```

Альтернатива:

- Поточний слот: `getSlot`
- Час слота: `getBlockTime`

```
curl -X POST https://api.mainnet-beta.solana.com \
  -H "Content-Type: application/json" \
  -d '{
    "jsonrpc": "2.0",
    "id": 1,
    "method": "getSlot"
  }'
```

Або:

$$\text{time_to_next_slot} = (\text{next_leader_slot} - \text{current_slot}) * \text{slot_duration}$$

A.4.18. Skipped Slots / Produced Slots

Метод: Через `getLeaderSchedule` + `getConfirmedBlocks` → Перевірка, чи блок існує для слота лідера.

Складне завдання — треба зберігати історію.

A.4.19. Account Assets

Метод: [getAccountInfo RPC](#)

```
// Identity
const accInfo = await connection.getAccountInfo(new
PublicKey(identityKey));
console.log(accInfo.lamports);
// Так само Votes/Withdrawer
```

Альтернатива:

```
curl -X POST https://api.mainnet-beta.solana.com \
-H "Content-Type: application/json" \
-d '{
  "jsonrpc": "2.0",
  "id": 1,
  "method": "getAccountInfo",
  "params": [
    "IDENTITY_PUBKEY",
    { "encoding": "jsonParsed" }
  ]
}'
```

Поля: .lamports, .data.parsed.info

A.4.20. Сервер (CPU, RAM, SSD)

Вимірюється або через власного агента на ноді або crowd-даними (типу Telemetry), якщо доступно.

Альтернатива:

- Користуватись дашбордом на основі Prometheus + Node Exporter
https://github.com/prometheus/node_exporter

A.4.21. Обчислення Spy Rank

Наші показники	Як ми рахуватимемо	Як рахує ці показники Stakewiz	Коментарі
Skipped Slots Score	10%	Value 0% + 15%	
Software Version Score	10%		
Uptime Score	(30 epochs) 100% - 20%	(30 days) 100% - 20%	
Inflation Commission Score	10	Up to 5% score for commission 0%, no score for 10% commission and above	
MEV Commission Score	10		
Published Information Score	3 out of 3 – 5%	5 out of 5 – 10%	
Site Score	approved 5%	-	
Stake Weight	Від 100k до 400k – 10% від 400k і далі 5% від 0k до 100k – 0%	Up to 15%, 0% for any stake that is >= 10% of the largest validator stake	
Operation History	5%	75 epoch + 10%	
TVC (5 epoch)	5%		

APY ($APY \approx (1 + R)^N - 1$)	10%		R — середній прибуток (reward rate) за одну епоху N — кількість епох у році (приблизно 134 епохи) ^ — піднесення до степеня
Version Penalty			
Superminority Penalty			
Withdraw Authority Penalty		Hawing the vote account withdraw authority set to validator's identity keypair is a bad security practice and incurs a -20% penalty	Differs from validator identity (good)

Очікується приклад розрахунку.

A.4.22. Налаштування RPC на ноді в mainnet

Замість <https://api.mainnet-beta.solana.com> (або dev та test) потрібно використовувати власний RPC-сервер.

1. В solana.service видалити:

```
--private-rpc
```

і додати (якщо нема):

```
--rpc-port 8899 \  
--rpc-bind-address 0.0.0.0 \  
--full-rpc-api \  

```

2. Налаштувати фаєрвол, щоб RPC приймала запити на порт 8899 тільки з конкретного IP:

```
sudo ufw allow ssh
```

```
sudo ufw enable
```

```
sudo ufw allow from IP to any port 8899 // вказати IP, де буде  
розташований сайт
```

```
sudo ufw deny 8899
```

```
sudo ufw status numbered
```

3. Перезавантажити сервіси і солану:

```
sudo systemctl daemon-reexec
```

```
sudo systemctl daemon-reload
```

```
sudo systemctl restart solana
```

4. Перевірка статусу:

```
sudo systemctl status solana
```

5. Перевірка RPC:

```
ss -tulnp | grep 8899
```

повинен повернути:

```
tcp    LISTEN 0          1024          0.0.0.0:8899      0.0.0.0:*
users: ("agave-validator",pid=1093848,fd=437864)
```

6. Локальний запит (після того, як пішов catchup):

```
curl http://127.0.0.1:8899 -X POST -H "Content-Type: application/json" -d
'{"jsonrpc":"2.0","id":1,"method":"getHealth"}'
```

// замість <http://127.0.0.1:8899> - вказати адресу RPC-сервера

повинен повернути:

```
{"jsonrpc":"2.0","result":"ok","id":1}
```