Research

# Рішення для реалізації адмін-панелі.

## А) Пошук відомих рішень:

За запитом google “admin panel express” було знайдено наступні рішення:

1. AdminJS
2. ForestAdmin
3. KeystoneJS
4. ApostpropheCMS
5. AdminBro
6. Strapi
7. Express.js + coreUi Admin Dashboard Template

## Б) Аналіз знайдених рішень

При порівнянні знайдених інструментів їх буде оцінено за наступними параметрами: функціональність, легкість використання, інтеграція з Express.js, підтримка баз даних, спільнота, вартість ліцензії, розширюваність, відгуки. Для порівняння буде створено описові таблиці з суб’єктивною оцінкою якості кожного з інструментів та коротким описом кожного критерію(від 1 до 3, де 1 – дуже погано, 3 – дуже добре).

|  |  |  |
| --- | --- | --- |
| AdminJs | | |
| Критерій | Опис | Оцінка |
| Функціональність | Має підтримку карт, heatmaps, категорій, графіків, ролей, локалізації, управління контектом, валідіції даних, генерацію PDF звітів, логування, вбудовані методи для CRUD потрібних елементів, їх пошуку, виведення списку та показу окремих елементів. | 3 |
| Легкість використання | Встановлюється за допомогою менеджера пакетів та запускається за допомогою коду, який можна знайти в документації | 3 |
| Інтеграція з Express.js | Має офіційний плагін для Express.js від розробника | 3 |
| Підтримка баз даних | Підтримка TypeORM, Sequelize, Prisma, MikroORM, Objection, SQL та Mongoose, що дає підтримку багатьох популярних SQL та NoSQL баз даних | 3 |
| Спільнота | Активна участь спільноти у вирішенні різноманітних проблем, а також учать у розробці користувацьких плагинів та адаптерів, має 2500+ користувачів у slack | 3 |
| Вартість ліцензії | 0$ для open source, 12$/міс для розміщення у хмарі(в ціну включено хостинг та персональну технічну підтримку) | 3 |
| Розширюваність | Є документація по написанню власних компонентів, стилів, декораторів, зміні представлень форм | 3 |

Підсумок: має великий функціонал та широке поле для кастомізації, є безкоштовним рішенням, яке має широку підтримку.

|  |  |  |
| --- | --- | --- |
| Forest Admin | | |
| Критерій | Опис | Оцінка |
| Функціональність | Має підтримку предналаштувань в залежності від продукту, який треба адмініструвати. В безкоштовній версії має обмежений функціонал, який включає: CRUD, пошук, сортування, custom actions, Dev, Staging і Production середовища. Має підтримку карт, категорій, графіків.  В платній версії має дозволи для ролей, Single Sign-On, нотатки, двофаторну автентифікацію, білий список ip-адрес, логування, | 2 |
| Легкість використання | Не очевидно написана документація, яка до того ж містить биті посилання, які уповільнили пошук вкладинки з інструкціями для встановлення. Саме встановлення відбувається легко, але інструкція містить тільки фрагменти коду для встановлення | 2 |
| Інтеграція з Express.js | Має підтримку Express.js та документацію на сайті | 3 |
| Підтримка баз даних | Підтримка MariaDB, Microsoft SQL Server, MongoDB, MySQL, PostgreSQL, SupraBase, Metabase | 2 |
| Спільнота | Участь спільноти у вирішенні проблем у вкладинці community на сайті Forest Admin | 3 |
| Вартість ліцензії | 0$ per user per month з обмеженим функціоналом, для повнного доступу ціна надається тільки після реєстрації та анкетування | 1 |
| Розширюваність | відсутня | 1 |

Підсумок: має обмежений функціонал для безкоштовної версії, не має підтримки кастомізації, обмажена, але достатня підтримка баз даних, незрозуміла політика цін для бізнес-клієнтів.

|  |  |  |
| --- | --- | --- |
| KeystoneJS | | |
| Критерій | Опис | Оцінка |
| Функціональність | Має підтримку CRUD, GraphQL API, менеджера сесій та ролей та доступів, фільтрації, пошуку та міграцій баз даних, аутентифікацій, та списків. Не знайдено інформації про графіки, карти, діаграми та інші способи представлення даних | 2 |
| Легкість використання | Встановлення початкового проєкту відбувається однією командою | 3 |
| Інтеграція з Express.js | KeystoneJS – фреймворк, що створено на основі Express.js. | 3 |
| Підтримка баз даних | Підтримка Postgres, MySQL та SQLite, але SQLite не рекомандується до використання | 1 |
| Спільнота | Участь спільноти у вирішенні проблем у slack, що має 2500+ користувачів | 3 |
| Вартість ліцензії | 0$ | 3 |
| Розширюваність | Можливість створити свої поля | 2 |

Підсумок: Має невеликий функціонал загалом, погано кастомізується, та підтримує малу кількість баз даних, підтримка noSQL баз даних відсутня.

|  |  |  |
| --- | --- | --- |
| ApostpropheCMS | | |
| Критерій | Опис | Оцінка |
| Функціональність | Має підтримку CRUD, локалізації, користувачів, форм, імпорту/експорту даних,інструментів SEO, SSO, публікації медіа, відкладеної публікації. Також має платні функції, такі як мультисайт, розширені досзволи для користувачів і груп, а також авторизація | 2 |
| Легкість використання | Встановлення початкового проєкту відбувається однією командою | 3 |
| Інтеграція з Express.js | Має модуль ініціалізації для Express.js | 3 |
| Підтримка баз даних | Підтримка MongoDB | 1 |
| Спільнота | Участь спільноти у вирішенні проблем у Discord, GitHub, StackOverflow та на форумі компанії. | 3 |
| Вартість ліцензії | Community edition – 0$, pro – 199$/міс., assembly – 499$/міс. | 1 |
| Розширюваність | Має додаткові пакети розширення, але деякі з них потребують наявність ліцензії | 2 |

Підсумок: Має невеликий функціонал загалом, погано кастомізується, та підтримує тільки одну базу.

|  |  |  |
| --- | --- | --- |
| AdminBro | | |
| Критерій | Опис | Оцінка |
| Функціональність | Має підтримку CRUD, локалізації, користувачів, файлів, ролей користувачів. | 2 |
| Легкість використання | Встановлюється за допомогою менеджера пакетів та запускається за допомогою коду, який можна знайти в документації | 3 |
| Інтеграція з Express.js | Має офіційний модуль для Express.js від розробника | 2 |
| Підтримка баз даних | Підтримка MongoDB та PostgreSQL | 1 |
| Спільнота | Неробоче посилання на канал у Slack | 1 |
| Вартість ліцензії | 0$ | 3 |
| Розширюваність | Є документація по написанню власних компонентів, стилів, зміні представлень форм | 3 |

Підсумок: Інструмент все ще у розробці на стадії beta-версії, тому погано підійде для використання у реальному проєкті.

|  |  |  |
| --- | --- | --- |
| Strapi | | |
| Критерій | Опис | Оцінка |
| Функціональність | Має підтримку CRUD, локалізації, користувачів, файлів, ролей користувачів, категорій, content-type builder`а, плагінів, відстеження версій документу, пошуку, фільтрації, лейблів, бібліотеки медіа та інших функцій. Додаткові функції, такі як логування та SSO для платної версії | 3 |
| Легкість використання | Встановлення початкового проєкту відбувається однією командою | 3 |
| Інтеграція з Express.js | Використання Express із Strapi має бути дуже складним і не має сенсу. Strapi заснований на Koa | 1 |
| Підтримка баз даних | Підтримка MySQL, MariaDB, PostgreSQL, SQLite | 2 |
| Спільнота | Широка підтримка спільноти у github, discord, twitch, youtube, та на форумі компанії | 3 |
| Вартість ліцензії | 0$ для self-hosted, спеціальна ціна для enterprice рішень | 3 |
| Розширюваність | Є документація по написанню власних полів, кастомізації back-end частини, встановленню плагінів розширень | 3 |

Підсумок: Великий спектр інструментів, легке користування, але немає підтримки noSQL баз даних, а також інструмент не працює з Express.js, тому не підходить для використання за умовою.

|  |  |  |
| --- | --- | --- |
| Express.js + coreUi Admin Dashboard Template | | |
| Критерій | Опис | Оцінка |
| Функціональність | Надає тільки візуальний шаблон для створення андмін-панелі, весь функціонал треба розробляти самому. Має платну ліцензію для отримання додаткових функцій, таких як Starter Project (Boilerplate) | 1 |
| Легкість використання | Встановлення початкового проєкту відбувається шляхом надсилання на пошту посилання для завантаження проєкту шаблону, а потім встановлення додаткових бібліотек для підтримки UI. | 1 |
| Інтеграція з Express.js | Можна використовувати із Express.js, але весь функцонал для інтеграції треба писати самому | 1 |
| Підтримка баз даних | Підтримка тих баз даних, підтримку яких пропише розробник | 1 |
| Спільнота | Вкрай мала | 1 |
| Вартість ліцензії | 0$ для порожнього шаблону, 127$, 197$ або 979$ в залежності від функціоналу | 1 |
| Розширюваність | Документація по кастомізації UI компонентів | 2 |

Підсумок: Не підходить, як швидке рішення для створення адмін-панелі

## В) Вибір рішення

У попередньому блоці було встановлено, що Strapi не підтримує Express.js, а coreUi Admin Dashboard Template є дуже неефективним рішенням, тому їх в подільшому розглянуто не буде.

Нижче наведено таблицю з загальними оцінками знайдених підходящих рішень, на основі якої буде обрано кінцевий інструмент.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Критерій | Інструмент | | | | |
|  | AdminJs | ForestAdmin | KeystoneJS | ApostpropheCMS | AdminBro |
| Функціональність | 3 | 2 | 2 | 2 | 2 |
| Легкість використання | 3 | 2 | 3 | 3 | 3 |
| Інтеграція з Express.js | 3 | 3 | 3 | 3 | 2 |
| Підтримка баз даних | 3 | 2 | 1 | 1 | 1 |
| Спільнота | 3 | 3 | 3 | 3 | 1 |
| Вартість ліцензії | 3 | 1 | 3 | 1 | 3 |
| Розширюваність | 3 | 1 | 2 | 2 | 3 |
| **Середнє значення** | **3** | **2** | **2,43** | **2,14** | **2,14** |

Можна побачити, що найкращим інструментом для реалізації адмін-панелі є AdminJs. Цей інструмент пропонує найбільш широкий спектр функціоналу та найбільш вигідну цінову політику.

У якості додаткових інструментів можна використовувати:

1. Express.js Middleware для додаткової обробки запитів перед тим, як вони потраплять до AdminJs
2. Passport.js для реалізації автентифікації
3. База даних для зберігання даних до адмін-панелі
4. WebSocket для налаштування сповіщень у реальному часі
5. Бібліотеки для завантаження та обробки медіа
6. Інструменти кешування для зменшення навантаження на БД

# Рішення для для роботи з медіа-файлами.

## А) Пошук відомих рішень:

1. Cloudinary
2. AWS S3
3. Firebase Storage
4. Multer
5. Busboy
6. Formidable
7. Sharp

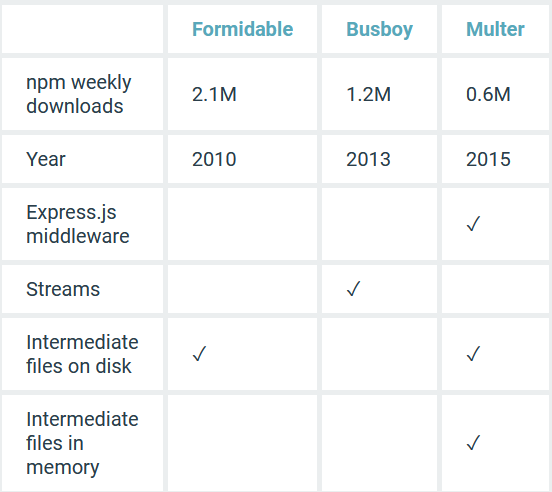
## Б) Аналіз знайдених рішень

При аналізі даних рішень треба враховувати, що дані рішення неможна порівнювати так, як було зроблено з рішеннями для реалізації адмін-панелі, адже а цьому випадку інструменти можуть змінюватись в залежності від потреб, а також комбінуватися, якщо в цьому є необхідність.

Розглянемо Cloudinary, AWS S3 та Firebase Storage: всі три сервіси є хмарними сховищами, що мають SDK для завантаження файлів у хмару. Всі три сервіса мають SDK для Node.js, тому можна реалізувати завантаження медіафайлів з Express.js серверу.

* 1. Cloudinary має потужні інструменти для маніпулювання медіа-файлами, такі як зміна розміру, оптимізація, фільтри тощо. Також є додаткові бібліотеки і інструменти для оптимізації зображення та відео. Cloudinary має бескоштовний обмежений тарифний план та багато планів, які можна обрати в залежності від потреб. Також, суб’єктивно, має найкращу документацію серед трьох сервісів
  2. AWS S3 надає масштабоване сховище, але робота з ним може бути складнішою, особливо при реалізації розширеного функціоналу. Також має сітку тарифних планів, серед яких є і безкоштовний
  3. Firebase Storage може бути зручним для розробників, що користуються і іншими firebase службами. Він має простий інтегрований інтерфейс, що може спростити розробку у порівнянні з Cloudinary або AWS S3, але може бути менш потужним

Розглянемо Multer, Busboy та Formidable. Ці три утиліти дозволяють читати та обробляти файли на боці node.js серверу. У процесі дослідження було знайдено наступну таблицю:



1. Multer є middleware для express.js серверу, та допомагає зберігати файли у пам’яті та на диск. Також Multer дозволяє визначити конфігурацію завантаження файлів, включаючи обмеження типів файлів та розмірів.
2. Busboy — це потоковий парсер на основі подій. Замість зберігання проміжних файлів він забезпечує потік до вхідного файлу. Він не прив’язаний до express.js, але цілком може бути використаний для обробки файлів.
3. Formidable зберігає файли у тимчасовому каталозі на жорсткому диску та також не є прив’язаним до express.js.

Розглянемо sharp. Цей інструмент дозволяє обробляти зображення: змінювати розмір, обрізати та форматувати. Він стане при нагоді, якщо зображення можна стиснути або обрізати аби зберігти місце на сервері.

## В) Вибір рішення

Для того, щоби обрати певний спосіб зберігання файлів(хмарний або локальний), треба провести аналітику навантаження на сервіс, відмовостійкості кожного з рішень, ціну на використання та можливість масштабування за необхідності.

На мою думку, якщо є обмежений обсяг файлів та не виникає вимог до великої швидкодії та високої доступності, то локальне зберігання може бути зручним вибором. А хмарне сховище підійде для забезпечення швидкодії та доступності, якщо є така необхідність.

Аналізуючі хмарні сховища, я би обрав Cloudinary через найбільш наповнену, на мою думку, документацію та можливість обробки медіа-файлів.

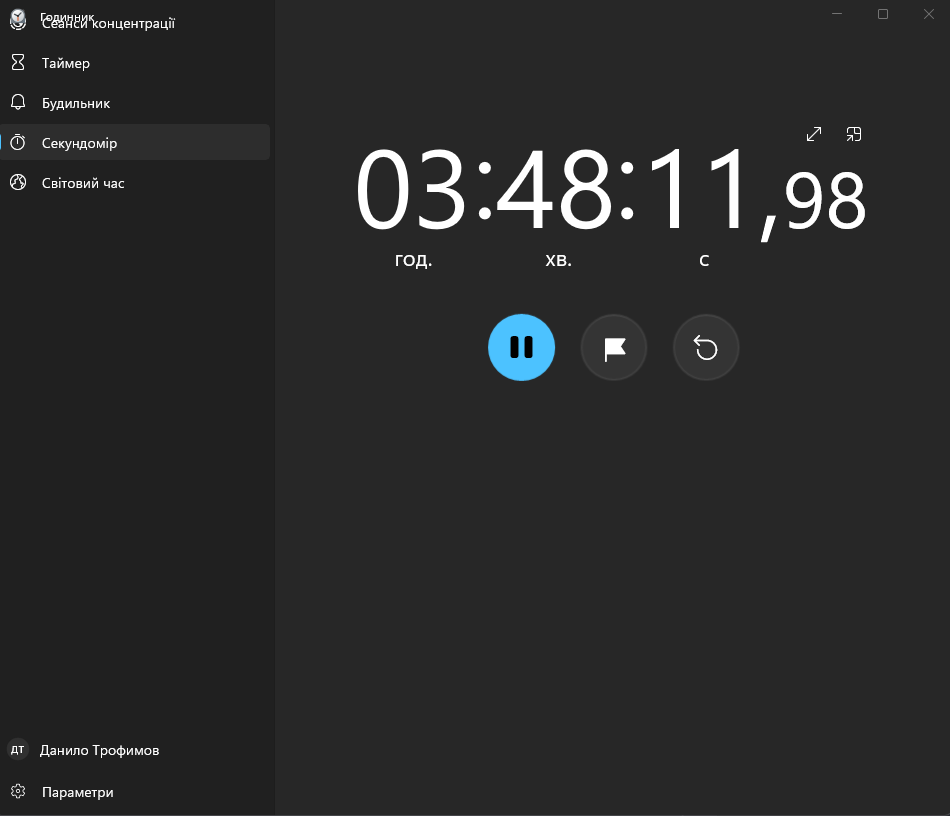
Для локального збереження файлів, я би обрав Multer, якщо немає потреби у роботі з потоками. Це простий інструмент, який розроблений для express.js і є простим у використанні. Але через відсутність опцій для редагування медіа-файлів, я би додатково використовував Sharp.

# Час:

Рішення для реалізації адмін-панелі: 2 години 45 хвилин

Рішення для для роботи з медіа-файлами: 1 година 3 хвинили

Всього: 3 години 48 хвилин



Розробка ПЗ: 5 годин 35 хвилин

