Метод розробки API підвищеної швидкодії

Виконав: ст. гр. 1КІ-16м
Маховик М.Ф.
Науковий керівник:
К.т.н., доц. Савицька Л.А.

Предмет та мета роботи

- **Предметом дослідження** є сучасні методи розробки API.
- **Метою роботи** є розробка методу розробки API, який дозволить підвищити ефективність та швидкість роботи API, а також його здатність до масштабування та адаптації до змін.

Актуальність роботи

- API дозволяє розробникам використовувати готові блоки для побудови додатків. З розвитком інтернет технологій та суттєвим збільшенням об’єму мобільного трафіку збільшується потреба у створенні додаткових клієнтів веб-додатків та інтеграції веб-сервісів.
- Завдяки використанню API користувачі можуть знайти нові сфери застосування додатку, покращити його відвідуваність та якість за рахунок інтеграції у власні веб-сервіси.

Технології та інструменти для розробки:

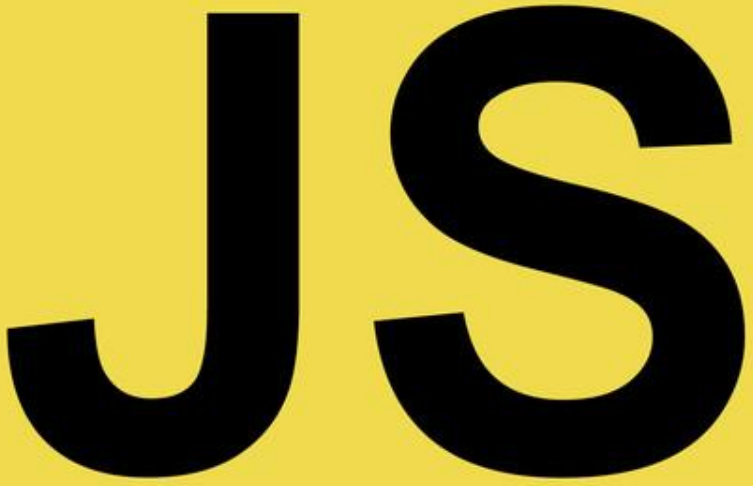
- JavaScript у якості серверної мови

- Програмна платформа Node.js

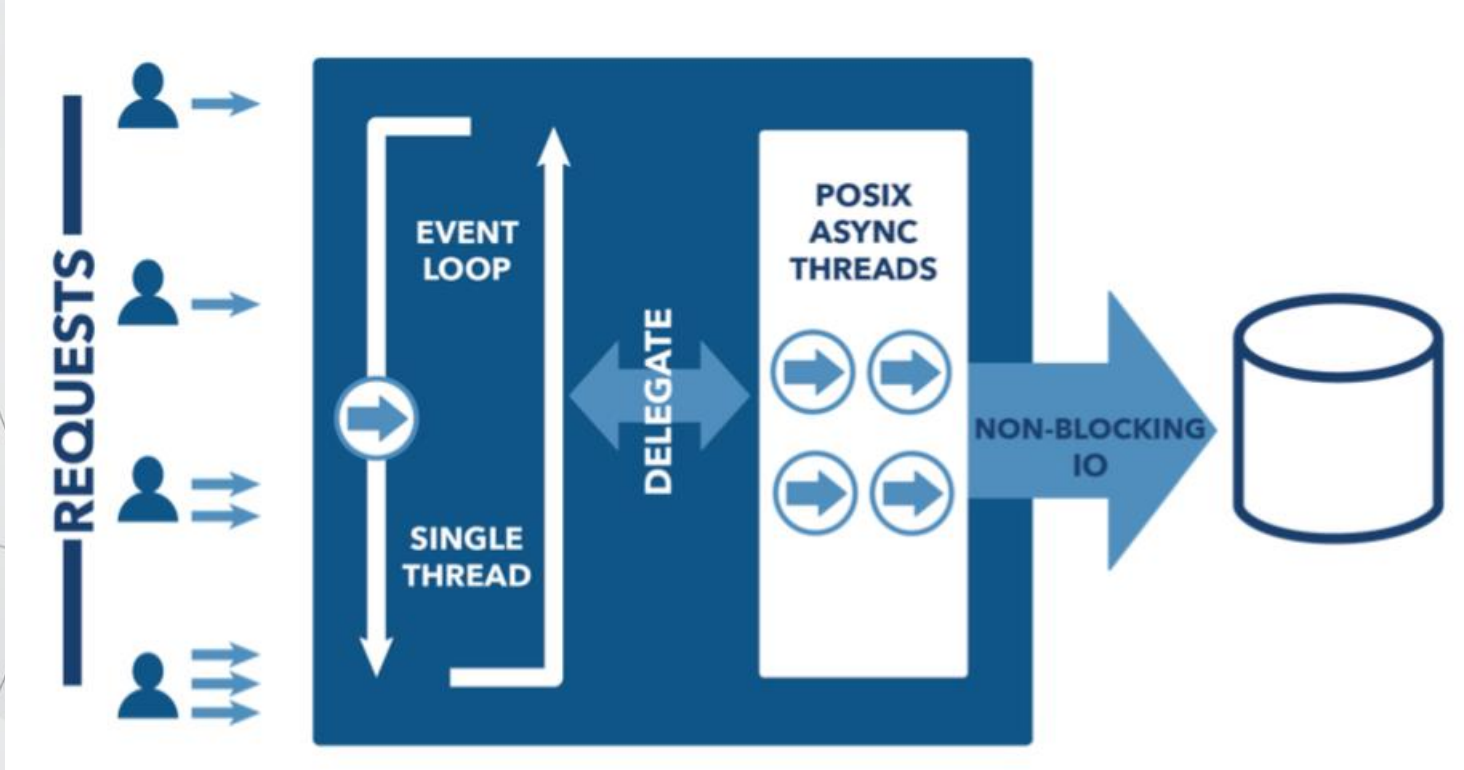
- NoSQL база даних MongoDB

Переваги використання JavaScript як серверної мови:

- ✓ асинхронність;
- ✓ серверна та клієнтська частини програми пишуться на одній мові;
- ✓ швидкодія



Програмна платформа Node.js використовує неблокуючу модель введення/виведення завдяки асинхронності JavaScript. Це дозволяє обробляти величезну кількість з’єднань одночасно, забезпечуючи масштабованість системи.

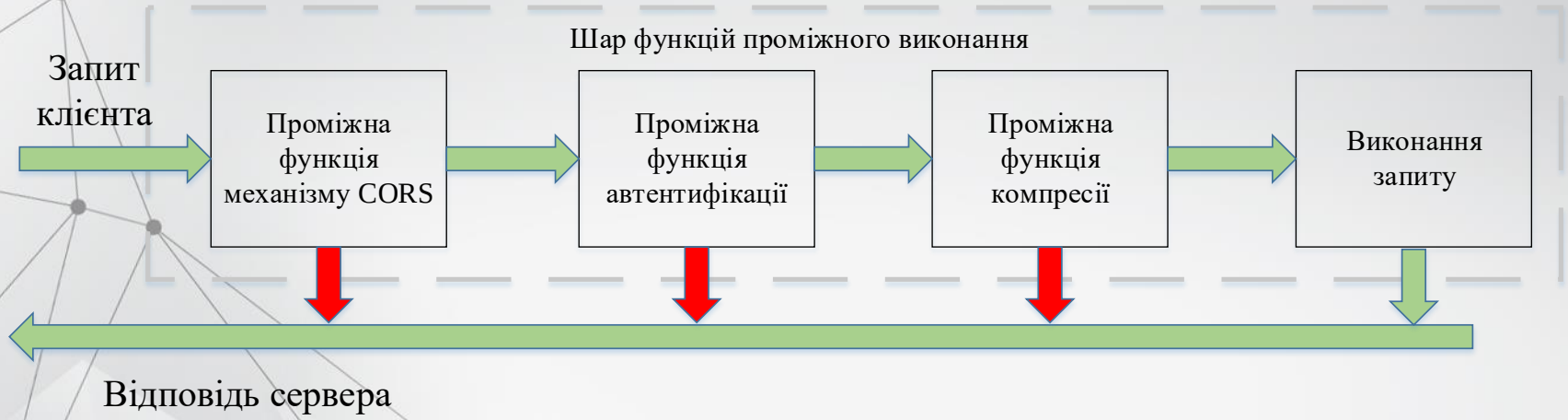


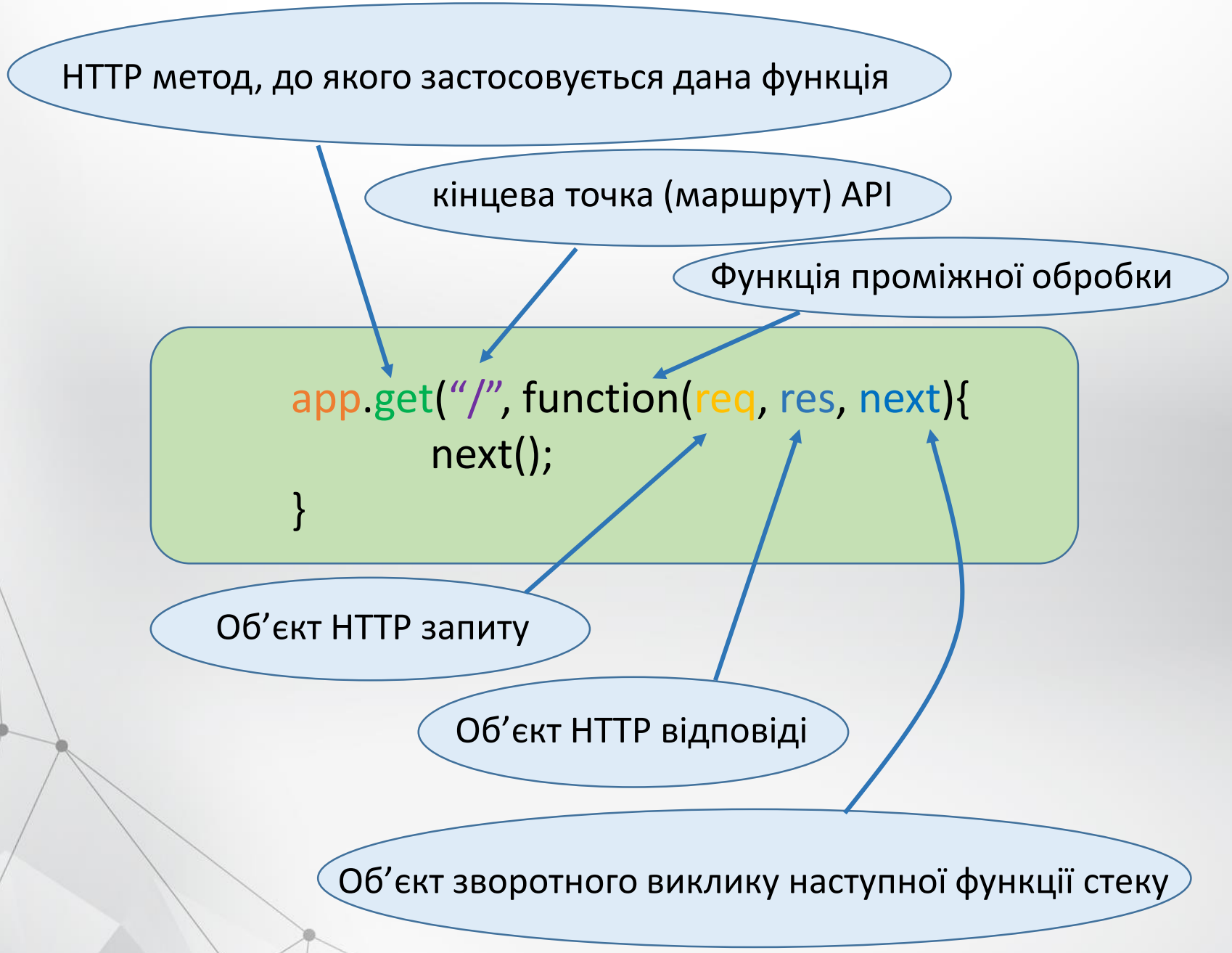
Нереляційна СУБД MongoDB

- ✓ гнучка організація даних;
- ✓ стабільність за рахунок підтримки реплікацій;
- ✓ потужний інструмент для обробки даних MapReduce;
- ✓ масштабованість та швидкий доступ до даних за рахунок технології sharding;

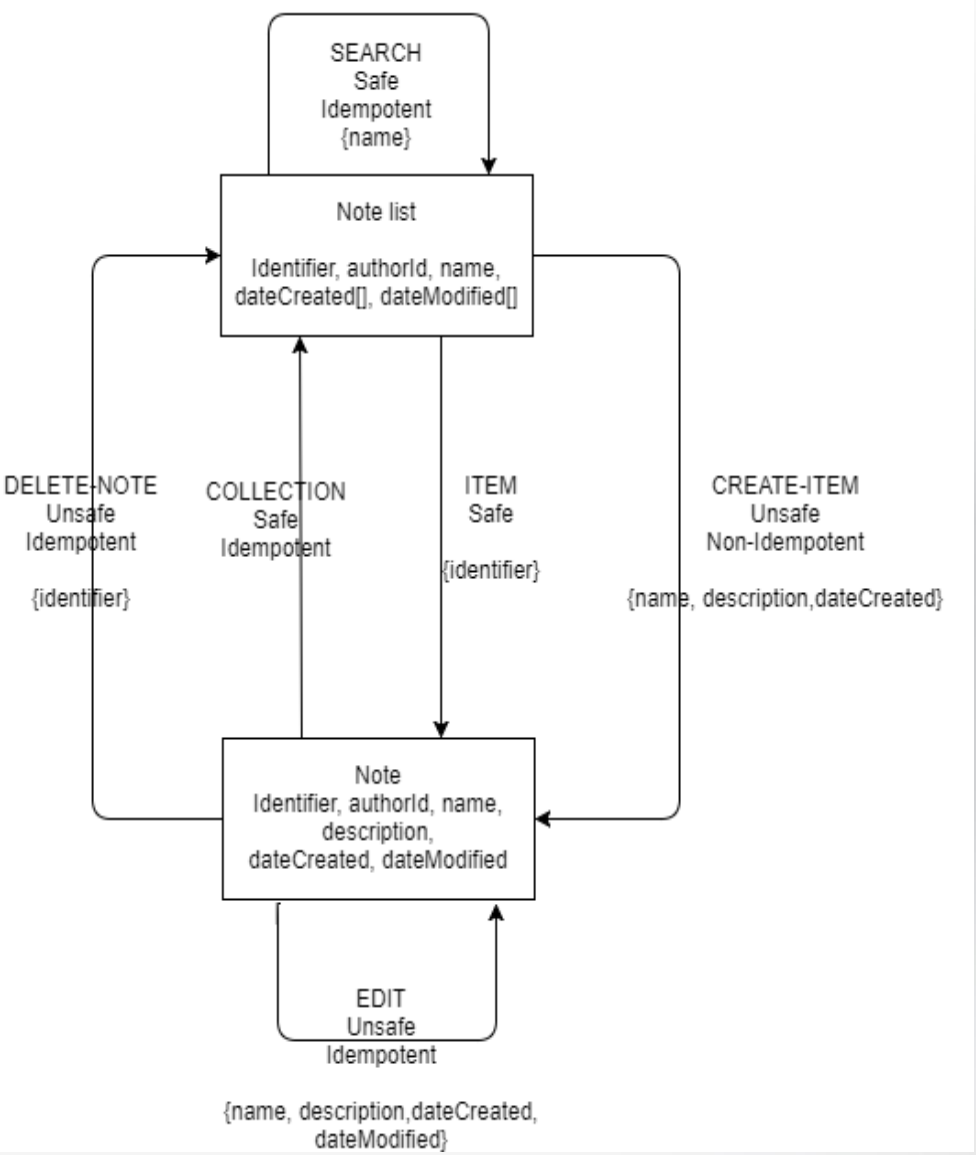
Проміжний шар MiddleWare

- Додатковий шар функцій проміжного виконання має вигляд стеку. Кожна функція виконує певну задачу і може вносити зміни в об’єкти запитів та відповідей, завершувати цикл «запит-відповідь», а також викликати наступну функцію проміжного стеку.




HTTP метод, до якого застосовується дана функція
кінцева точка (маршрут) API
Функція проміжної обробки
app.get("/", function(req, res, next){
next();
}
Об'єкт HTTP запиту
Об'єкт HTTP відповіді
Об'єкт зворотного виклику наступної функції стеку

- Для демонстрації роботи методу було розроблено API для веб-додатку для керування замітками. Діаграма стану додатку відображає семантичні стани додатку та переходи між ними, які ініціюються HTTP- запитами.



Система кінцевих точок для викликів API

- app.post('/login', notes.login);
- app.post('/reg', notes.reg);
- app.get('/logout', notes.logout);
- app.get('/profile', notes.getUserProfile);
- app.post('/create', notes.createNote);
- app.get('/notes', notes.getAllNotes);
- app.get('/note', notes.findNote);
- app.put('/edit', notes.editNote);
- app.delete('/delete', notes.deleteNote);

Запропонований метод дозволяє підвищити швидкодію роботи API за рахунок створення проміжного шару middleware, в якому виконується компресія відповідей на запити, а також ряд інших функцій, таких як автентифікація, авторизація користувачів, налаштування механізму спільного використання ресурсів між різними джерелами CORS та ін. Застосування бази даних MongoDB дозволило масштабованість системи. Асинхронна модель введення/виведення дає можливість підвищити швидкість роботи системи та масштабувати систему без втрат у продуктивності.

Дякую за увагу!