# Державний вищий навчальний заклад

# «Запорізький національний університет»

# Міністерства освіти і науки України

# Лабораторна робота №1

# 

# Виконала студентка

# гр. 8.1212:

# Зінченко А.О.

# м. Запоріжжя

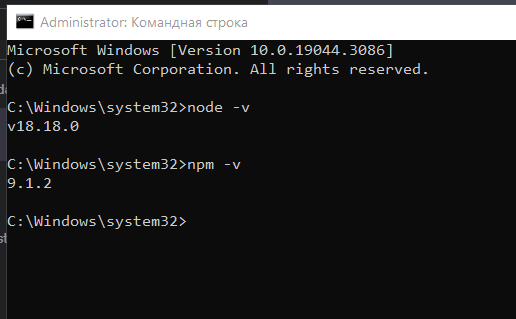
# 2022-2023 н.р.

**Крок 1: Ознайомлення з теоретичним матеріалом**

Перш ніж розпочати роботу, ретельно ознайомтеся з документацією Vue.js та зрозумійте основи роботи з цим фреймворком. Ви можете знайти документацію на офіційному сайті Vue.js.

**Крок 2: Встановлення Node.js та npm**

Переконайтеся, що на вашому комп'ютері встановлена остання стабільна версія Node.js та менеджера залежностей npm. Ви можете завантажити їх з офіційного сайту Node.js і встановити за інструкцією на сайті.



### Крок 3: Створення елементарного додатку за допомогою vue-cli + Крок 4: Тестування роботи додатку

1. Встановіть глобально утиліту vue-cli, якщо вона ще не встановлена, за допомогою команди npm:

npm install -g @vue/cli



Створіть новий Vue.js проект за допомогою vue-cli, виконавши команду:

npm init vite@latest <ваше-прізвище-англійською> -- --template vue

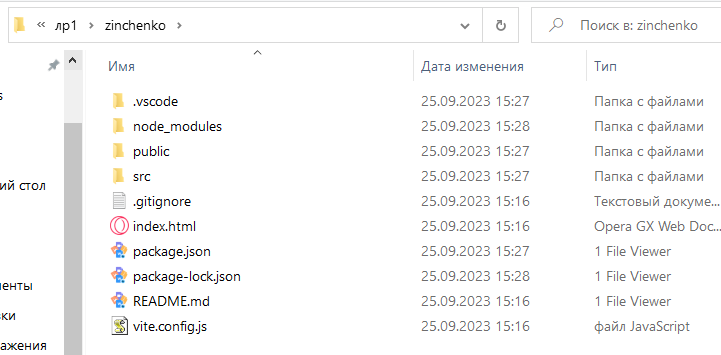
Слідуйте інструкціям на екрані для налаштування проекту.

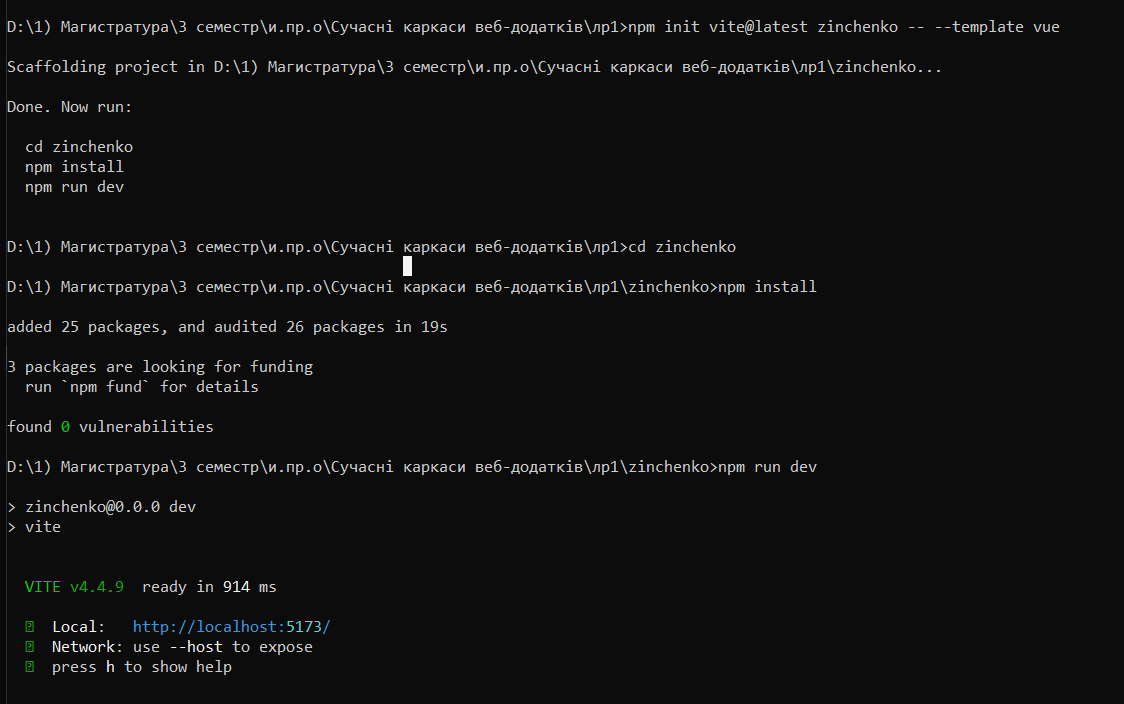
Запустіть ваш новий додаток за допомогою команди:

cd <ваше-прізвище-англійською>

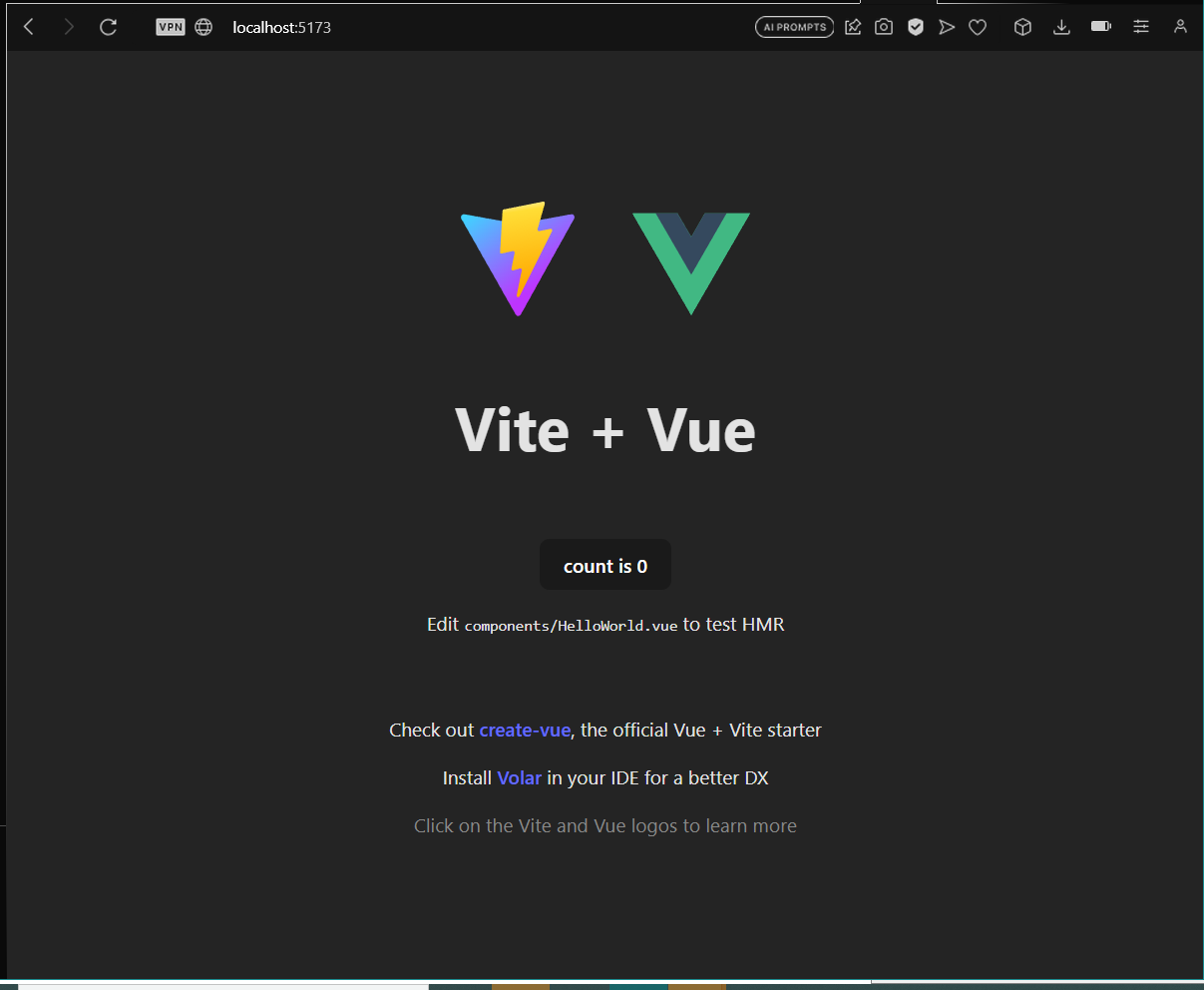
npm install

npm run dev



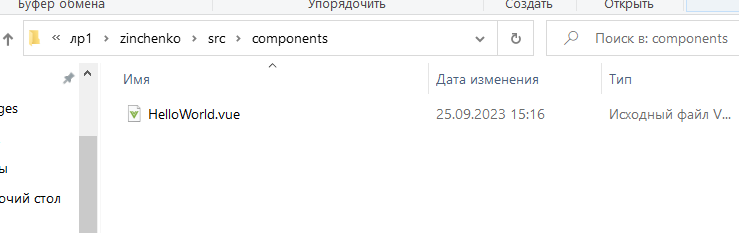


Відкрийте браузер і перейдіть за адресою http://localhost:8080 (або інша адреса, яка виведена в терміналі). Ви повинні бачити ваш елементарний Vue.js додаток у роботі.

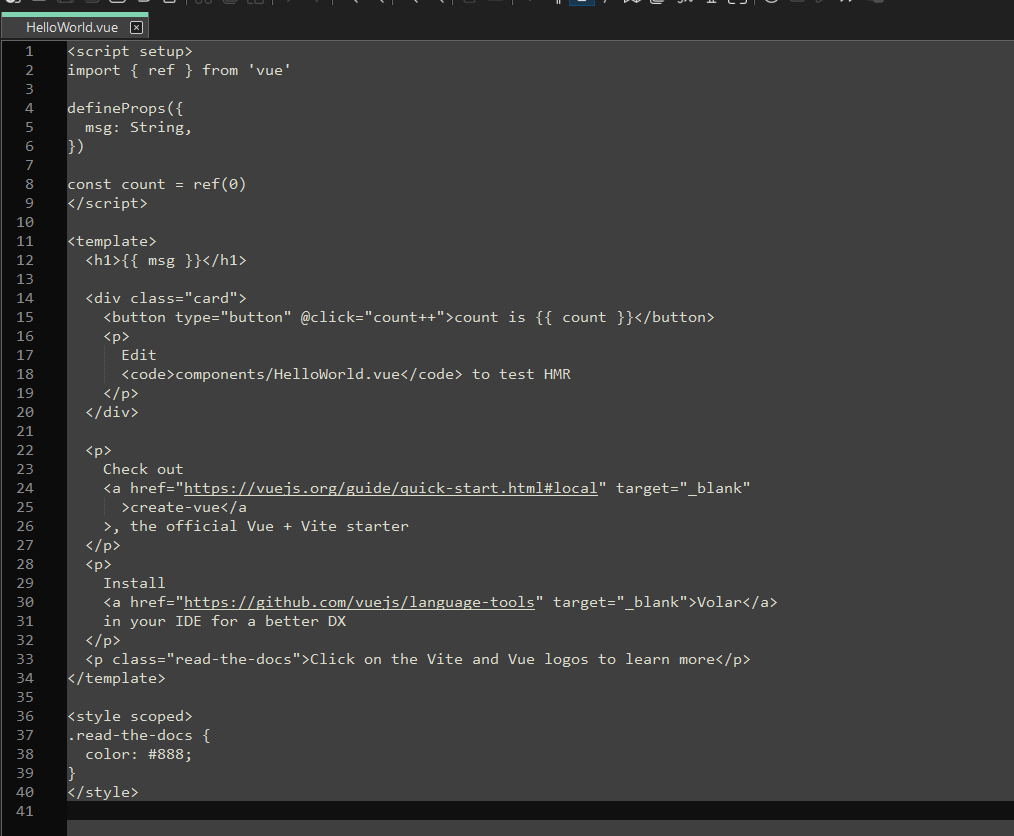


### Крок 6: Редагування компоненту

Знайдіть компонент HelloWorld.vue та додайте інформацїю про себе (Спеціальність, курс, група, ПІБ) у шаблоні компонента.



Початковий вміст HelloWorld.vue:



Після редагування:

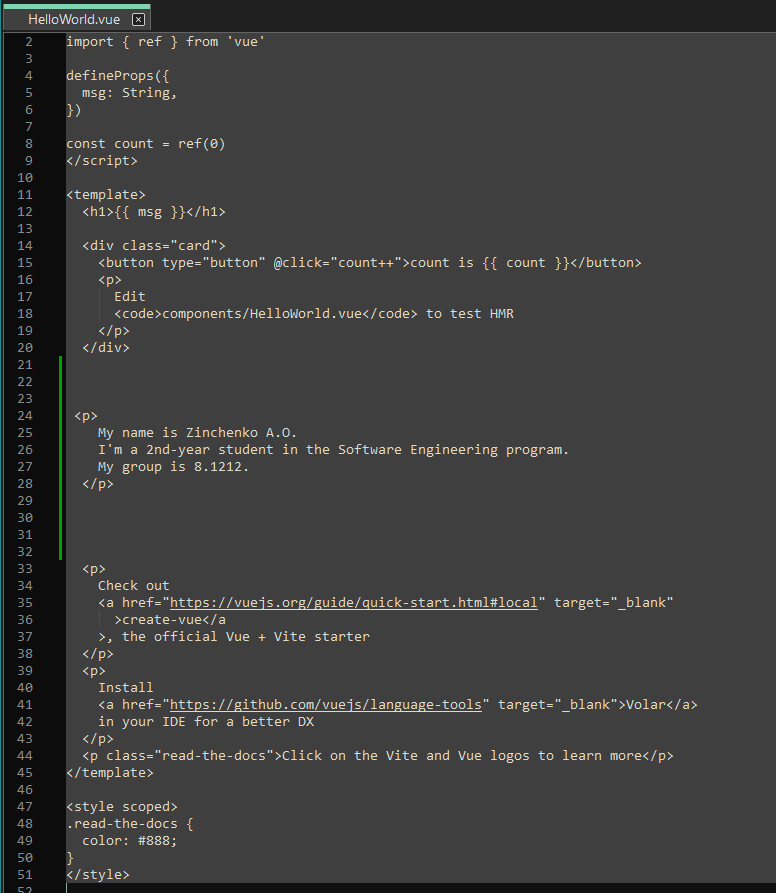
<p>

My name is Zinchenko A.O.

I'm a 2nd-year student in the Software Engineering program.

My group is 8.1212.

</p>





### Крок 7: Створення Dockerfile та .dockerignore

Створіть файл Dockerfile у кореневій директорії вашого проекту з наступним вмістом:

FROM node:16 as build-stage

WORKDIR /app

COPY package\*.json ./

RUN npm install

COPY . .

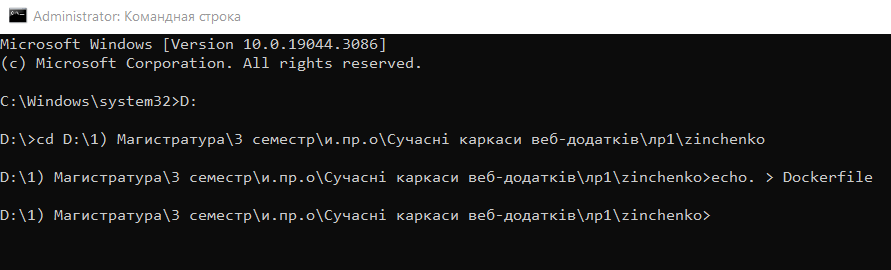
RUN npm run build

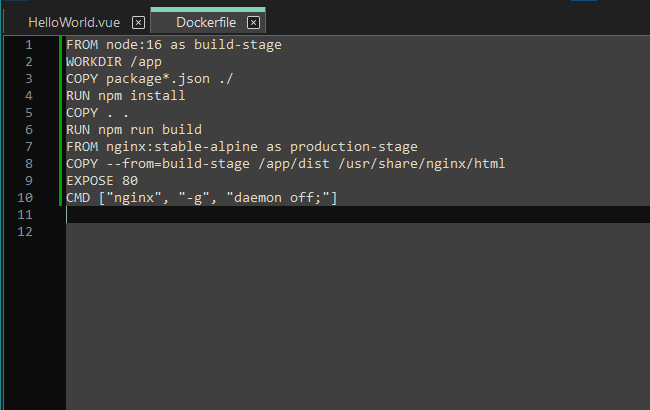
FROM nginx:stable-alpine as production-stage

COPY --from=build-stage /app/dist /usr/share/nginx/html

EXPOSE 80

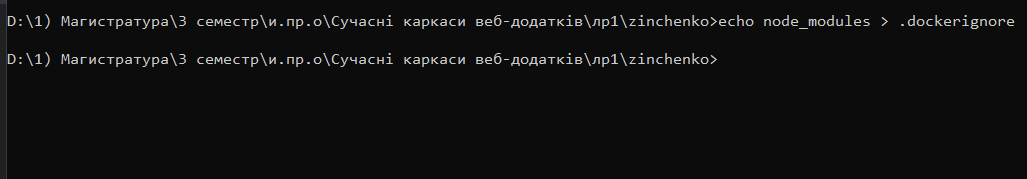
CMD ["nginx", "-g", "daemon off;"]





Створіть файл .dockerignore для виключення непотрібних файлів та директорій з образу Docker:

node\_modules





Додайте додаткові команти в секції scripts в файлі package.json

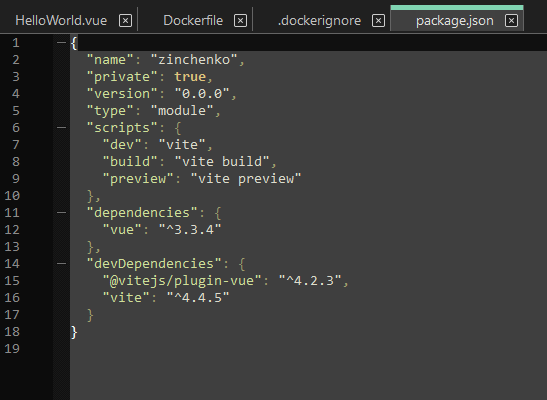
...

"docker-run": "docker build --tag vue-lab-1:latest . && docker run --rm -p 8080:80 --name vue-lab-1-instance vue-lab-1:latest",

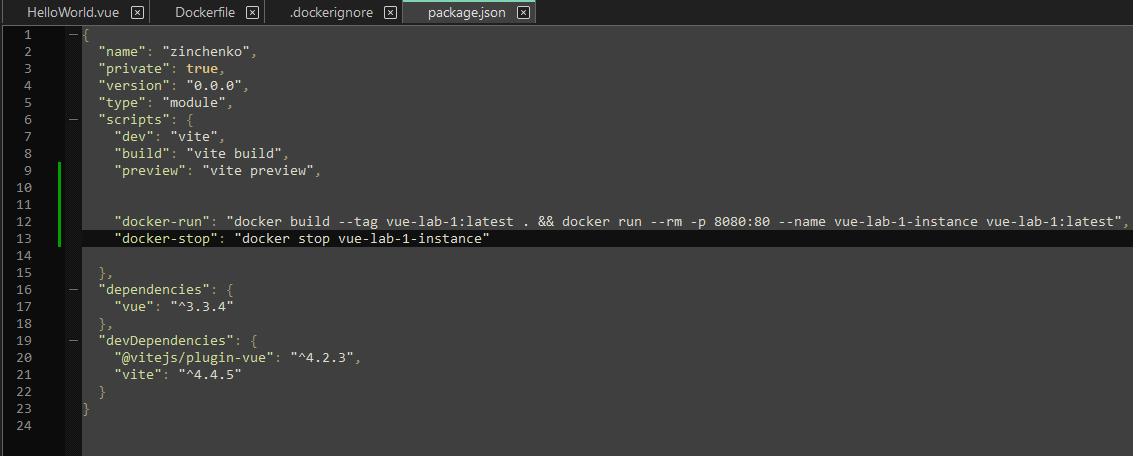
"docker-stop": "docker stop vue-lab-1-instance"

...

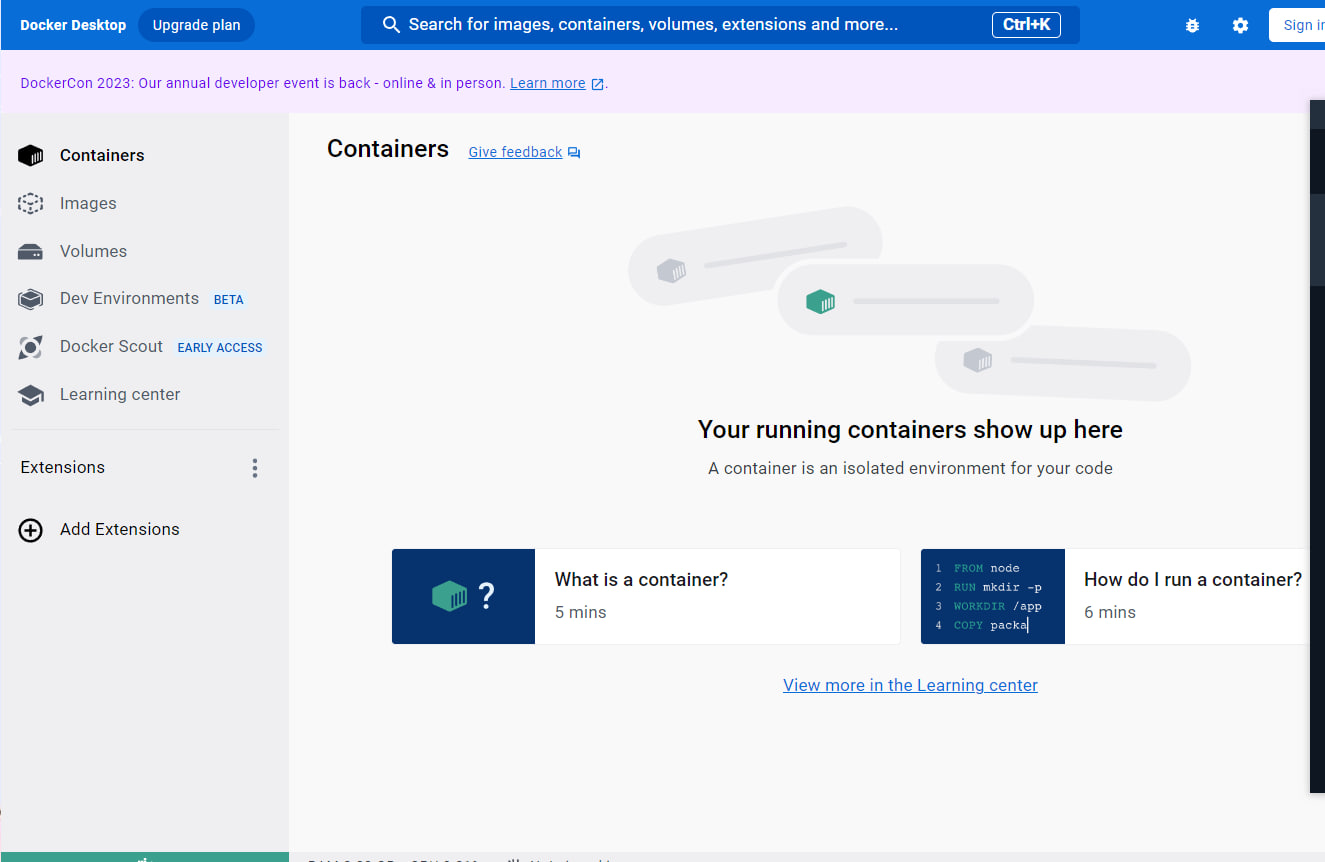
Початковий вміст package.json:

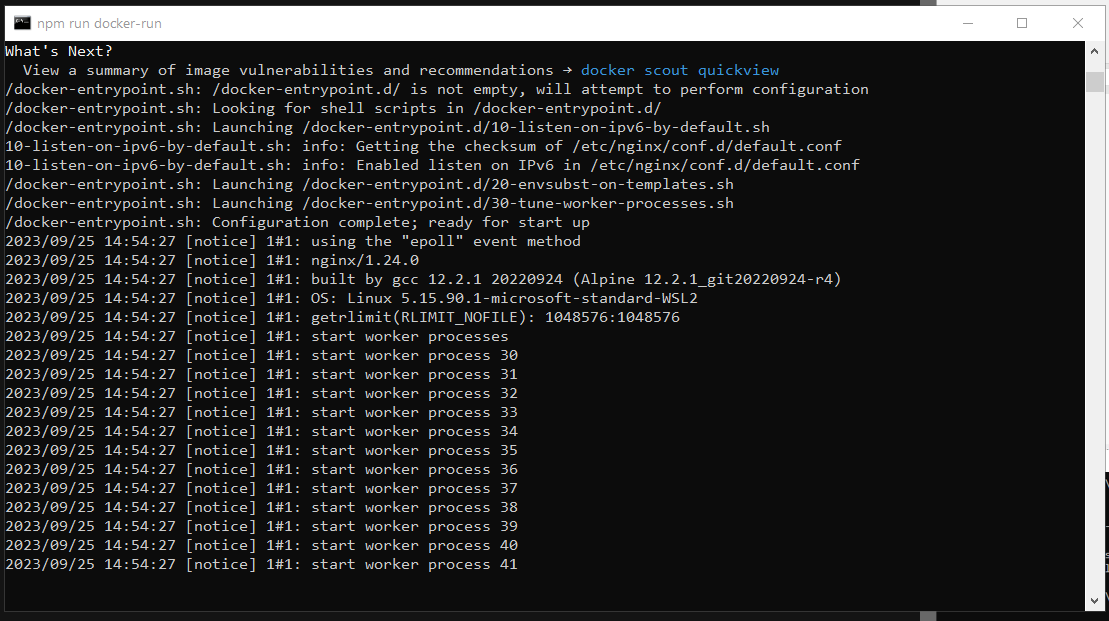


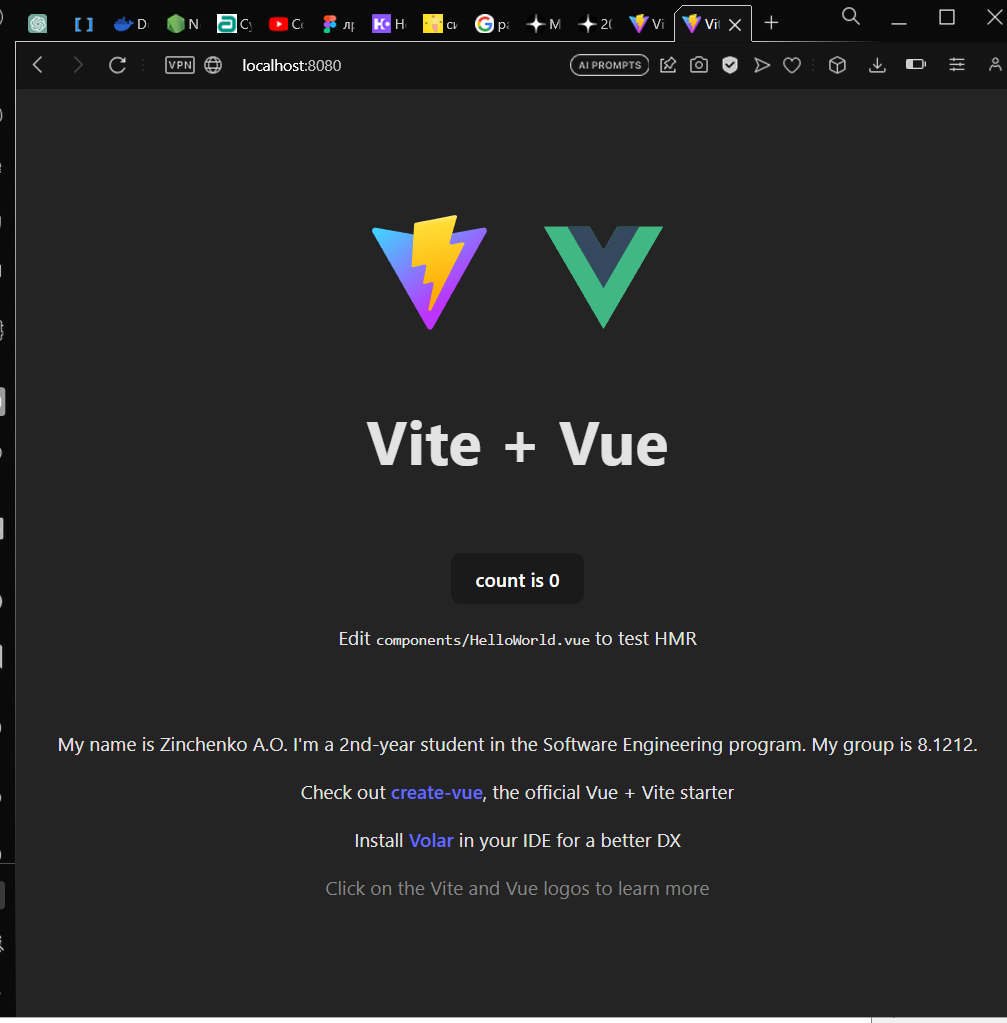
Після редагування:



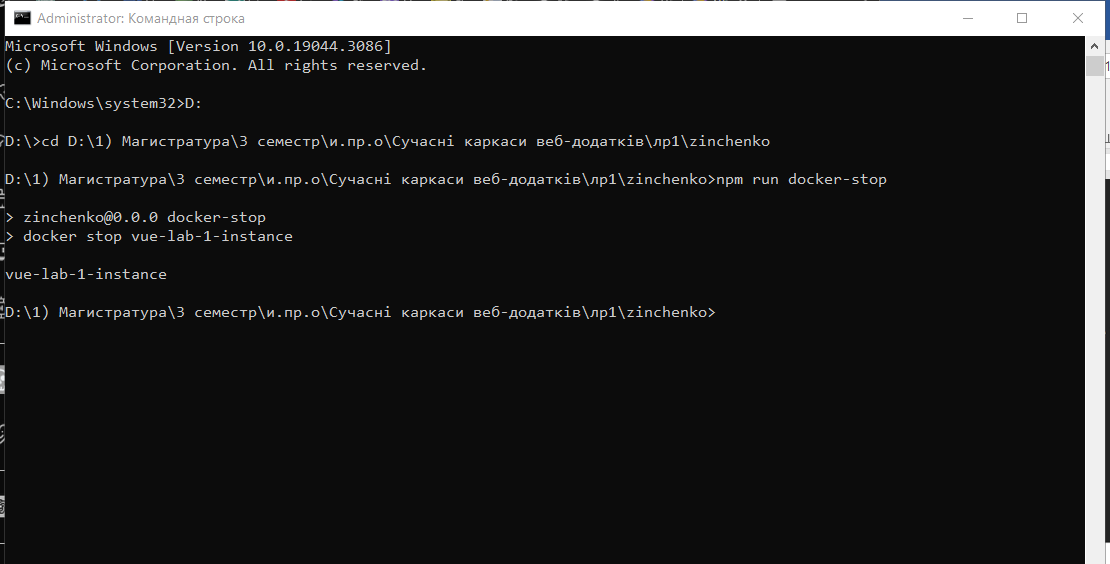
Виконайте команду *npm run docker-run* та відкрийте браузер на сторінці <http://localhost:8080/>







Зупиніть контейнер виконавши docker-stop.



## **Контрольні питання**

* Які кроки потрібно виконати перед початком роботи з Vue.js?

1. Встановлення Node.js
2. Встановлення Vue CLI (командний рядок)
3. Створення нового проекту Vue
4. Конфігурація проекту
5. Запуск

* Що таке Vue CLI і як його встановити глобально на комп'ютері?

Vue CLI (командний рядок Vue) - це інструмент командного рядка, який допомагає створювати та керувати проектами Vue.js. Він надає засоби для швидкого створення нових проектів, гарячої перезавантаження (Hot-Reload) під час розробки, побудови проекту для виробництва та багато інших корисних функцій.

Щоб встановити Vue CLI глобально, потрібно:  
1) Встановити Node.js

2) Встановити Vue CLI за допомогою npm та перевірити його наявність.

* Як створити новий Vue.js проект за допомогою Vue CLI?
  1. Встановити глобально утиліту vue-cli
  2. Створити проект, виконавши команду: npm init vite@latest <ваше-прізвище-англійською> -- --template vue
* Як запустити створений Vue.js додаток для перевірки його роботи?

cd <ваше-прізвище-англійською>

npm install

npm run dev

* Що таке Docker і для чого він використовується в розробці програмного забезпечення?

Docker - це платформа для контейнеризації програмного забезпечення, яка надає інструменти та середовище для створення, розгортання та управління контейнерами. Контейнери є віртуальними ізольованими середовищами, які містять програмне забезпечення та всі його залежності, що дозволяє програмам працювати однаково на будь-якому комп'ютері чи сервері, де встановлений Docker.

* Які основні етапи створення Dockerfile для розгортання Vue.js додатку?

Необхідно створити через термінал файл без розширання еcho. > Dockerfile і наповнити його:

FROM node:16 as build-stage

WORKDIR /app

COPY package\*.json ./

RUN npm install

COPY . .

RUN npm run build

FROM nginx:stable-alpine as production-stage

COPY --from=build-stage /app/dist /usr/share/nginx/html

EXPOSE 80

CMD ["nginx", "-g", "daemon off;"]

* Які файли можна включити до файлу .dockerignore, і для чого це робиться?

Файл .dockerignore використовується для визначення файлів та каталогів, які слід виключити з контексту збірки Docker-контейнера під час використання команди docker build. Всі файли та каталоги, перераховані в цьому файлі, будуть проігноровані під час збірки контейнера. Включення .dockerignore допомагає зменшити обсяг контексту збірки і прискорює процес збірки контейнера, особливо в великих проектах.

Файли node\_modules та yarn.lock можна виключити, оскільки Docker буде встановлювати залежності заново в контейнері.

Можна виключити файли, які створюються локальним середовищем розробника, такі як файли для IDE (наприклад, .vscode/, .idea/), кеш і тимчасові файли.

Якщо є файли з налаштуваннями, які варіються залежно від середовища (наприклад, файли .env), вони також можуть бути виключені.

Файли для тестів та налагодження, які не потрібні в контейнері, також можна виключити.

Як завантажити проект до репозиторію на GitHub або Bitbucket?

* 1. Зареєструватись на ресурсі.

2)Створити репозиторій та виконати команди:

git init

git add .

git commit -m "Перший коміт: додано Vue.js додаток"

git branch -M main

git remote add origin <посилання-на-ваш-репозиторій>

git push -u origin main

* Які основні переваги використання системи контролю версій для розробки програмного забезпечення?

СКВ дозволяють зберігати повну історію версій вашого програмного коду. Можна відстежувати, як змінювався код з часом, і переглядати історію змін для кожного файлу. Це допомагає зрозуміти, коли і чому внесені певні зміни.

Дозволяють вам вести розробку паралельно, об'єднувати зміни в одному коді та вирішувати конфлікти, які можуть виникнути при одночасних змінах.

Забезпечують збереження копії вашого коду в безпечному місці.

Дозволяють створювати гілки розробки, що дозволяє розробникам працювати над новими функціями або виправленнями помилок, не впливаючи на основний код. Гілки дозволяють розгалуження розробки та об'єднання змін пізніше.

Можна легко створювати і тестувати різні версії вашого коду на відокремлених гілках, що сприяє покращенню якості програмного забезпечення та зменшує ризик внесення помилок.

СКВ можуть інтегруватися з іншими інструментами та сервісами для автоматизації процесів розробки, таких як неперервна інтеграція (CI) та неперервна доставка (CD).