Міністерство освіти та науки України

Івано-Франківський Національний технічний університет нафти і газу

Кафедра Інженерії

програмного забезпечення

**Лабораторна робота №1**

“Ознайомлення з середовищем та базовими інструментами”

Виконав:

ст. гр. ІП-22-1

Токарєв С. О.

Перевірив:

Григорчук Л. І.

Україна

2025р.

**Завдання:**

1. Встановити необхідні бібліотеки та налаштувати середовище (Django + Vue Js)

2. Створити першу веб-сторінку.

3. Опублікувати проєкт на Github та Github Pages

**Виконання роботи:**

**1. Ініціалізація git**

За допомогою git init ініціалізовано порожній репозиторій та з

git add origin [git@github](mailto:git@github):retives/stud\_assistant

встановлено з’єднання з github-репозиторієм.

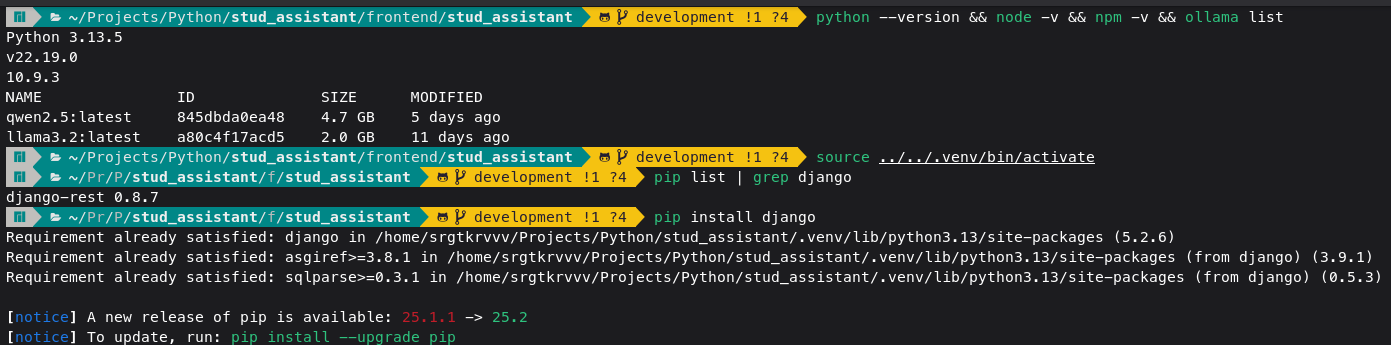
**2. Створення віртуального середовища python**

У корні проєкту використано **python -m venv .venv** що створює віртальне середовтще python, де можна встановити потрібні бібіліотеки не торкаючись глобального python.

3. **Встановлення бібліотек**

Для виконання цієї роботи потрібні наступні бібліотеки та програми:

* VSCode
* Python (virtual environment)
* Django
* Nodejs
* Npm
* Vue
* Ollama

На наступном скріншоті показані версії бібіліотек:

**4. Створення HTML-сторінки**

В базовому VueJs проєкті у файл App.vue введено код з розділу **Код Програми**. На даній сторінці створено основу чату яка буде використовуватись для підключення api мовної моделі.

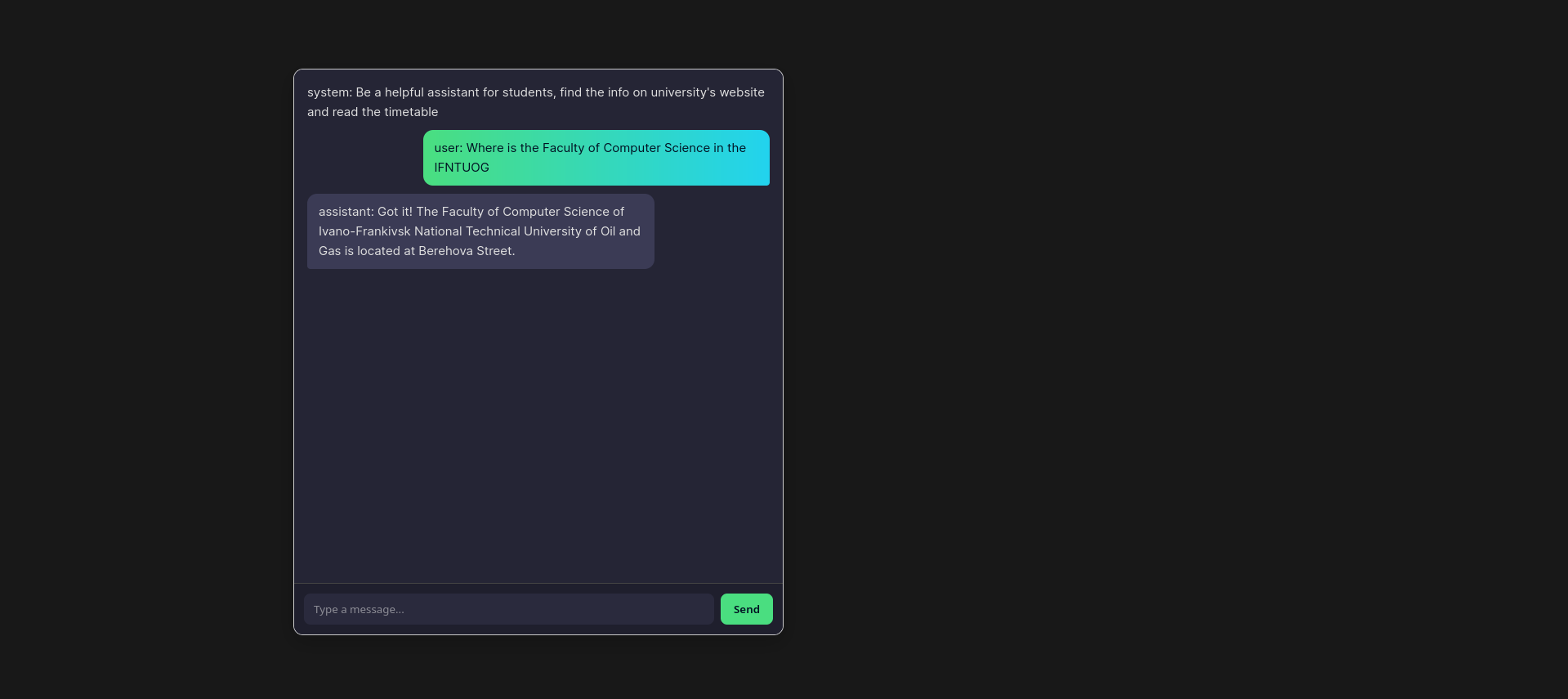
5. Запуск сервера

З допомогою **npm run dev** було запущено девелопмент сервер за адресою

[**http://loclahost:5173**](http://loclahost:5173/)

**6. Реультат виконання.**

Наразі на сторінці, що знаходиться на локальному сервері можна побачити наступне:



**7. Деплой на github pages**

Усі NodeJs проєкти об’єднанні штсрументом npm, що дозволяє запускати та деплоїти будь-який проєкт. Після використання команд:

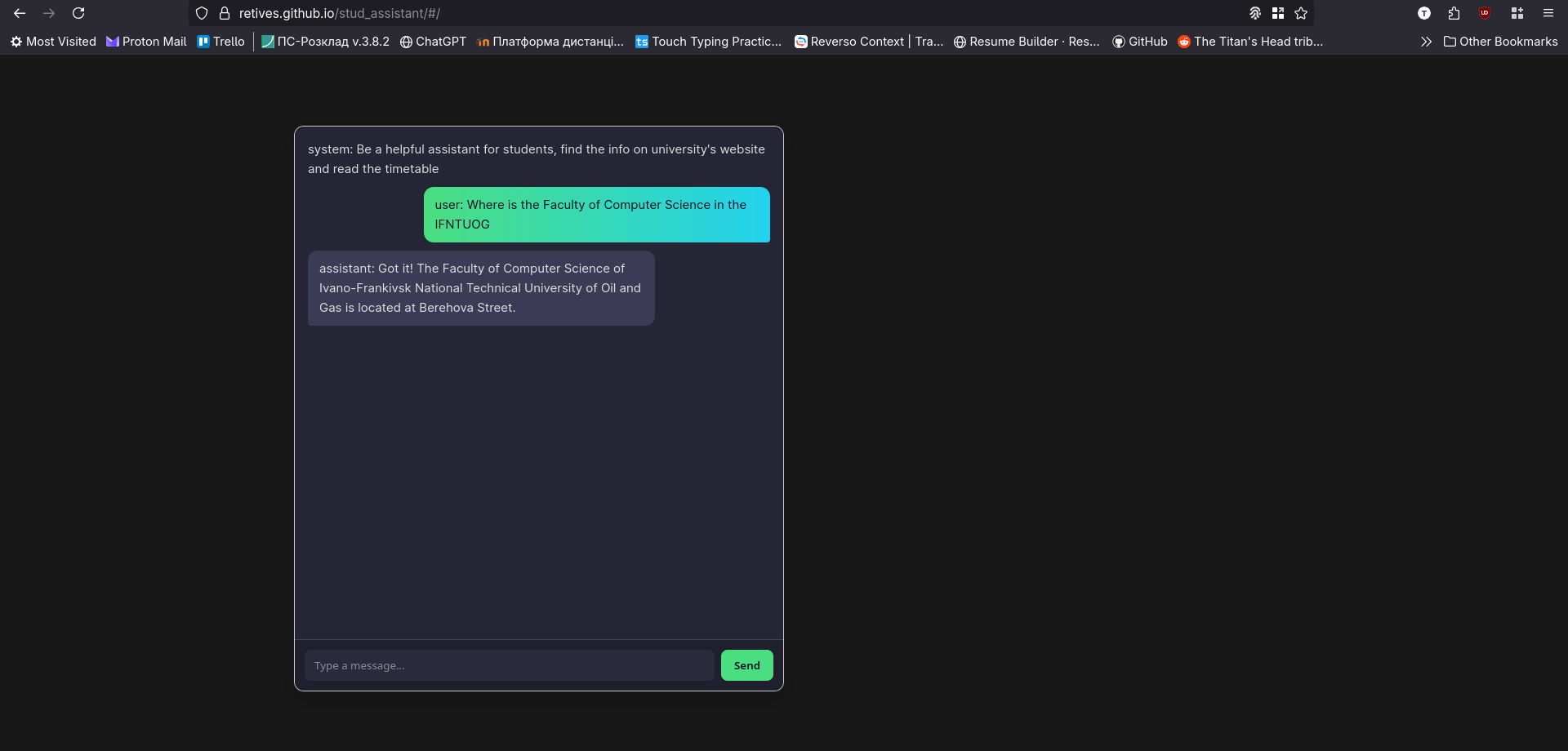
npm run build

npm run deploy

Було створено директорію dist, в якій містяться файли з сконфігурованими посиланнями на проєкт. Deploy в свою чергу відправляє цю папку на вказану в config.js гілку gh-pages.

**8. Результат**

Після успішного виконання команди за адресою https://<username>.github.io/<repo\_name> буде відображено index.html, що згенеровано в папці dist.

****

**Висновок:**

У цій роботі було завантажено усі бібліотеки для розробки інтелектуальних сервісів для мов Python(Django) та JS(NodeJS) та створено структуру проєкту, що слугуватиме основої для подальшої роботи в наступних лабораторних роботах. Також проєкт було опубліковано на Github Pages як перший досвід деплою на хмарні сервіси.

**Код програми**

<template>

<div class="chat-app">

<div class="messages">

<div v-for="(msg, i) in messages" :key="i" :class="msg.role">

<b>{{ msg.role }}:</b> {{ msg.content }}

</div>

</div>

<div class="composer">

<input v-model="input" @keyup.enter="formRequestJson" placeholder="Type a message..." />

<button @click="formRequestJson">Send</button>

</div>

</div>

</template>

<script>

export default {

name: "SimpleChat",

data() {

return {

messages: [

{"role":"system", "content":"Be a helpful assistant for students, find the info on university's website and read the timetable"},

{"role":"user", "content":"Where is the Faculty of Computer Science in the IFNTUOG"},

{"role":"assistant", "content":"Got it! The Faculty of Computer Science of Ivano-Frankivsk National Technical University of Oil and Gas is located at Berehova Street."}

],

input: "",

apiEndpoint: "https://localhost:11434",

apiKey: "",

model: "qwen2.5:latest",

};

},

methods: {

// Request method to the LLM

async sendMessage() {

if (!this.input.trim()) return;

const userMsg = { role: "user", content: this.input };

this.messages.push(userMsg);

const inputCopy = this.input;

this.input = "";

try {

const res = await fetch(this.apiEndpoint, {

method: "POST",

headers: {

"Content-Type": "application/json",

Authorization: `Bearer ${this.apiKey}`,

},

body: JSON.stringify({ model: this.model, messages: this.messages }),

});

const data = await res.json();

const reply = data.choices?.[0]?.message?.content || "[No response]";

this.messages.push({ role: "assistant", content: reply });

} catch (e) {

console.error(e);

this.messages.push({ role: "assistant", content: "Error calling API" });

}

},

async formRequestJson() {

if (!this.input.trim()) return;

const userMsg = { role: "user", content: this.input };

this.messages.push(userMsg);

const inputCopy = this.input;

this.input = "";

}

},

};

</script>

<style scoped>

.chat-app {

max-width: 600px;

margin: 40px auto;

font-family: 'Inter', sans-serif;

display: flex;

flex-direction: column;

height: 80vh;

border: 1px solid #ccc;

border-radius: 12px;

overflow: hidden;

background: #1e1e2f;

color: #e0e0e0;

box-shadow: 0 8px 24px rgba(0,0,0,0.2);

}

.messages {

flex: 1;

padding: 16px;

display: flex;

flex-direction: column;

gap: 10px;

overflow-y: auto;

background: #252535;

}

.messages .user {

align-self: flex-end;

background: linear-gradient(90deg, #4ade80, #22d3ee);

color: #012;

padding: 10px 14px;

border-radius: 12px 12px 4px 12px;

max-width: 75%;

word-wrap: break-word;

}

.messages .assistant {

align-self: flex-start;

background: #3b3b55;

padding: 10px 14px;

border-radius: 12px 12px 12px 4px;

max-width: 75%;

word-wrap: break-word;

}

.composer {

display: flex;

padding: 12px;

border-top: 1px solid #444;

background: #1f1f2d;

gap: 8px;

}

.composer input {

flex: 1;

padding: 10px 12px;

border-radius: 8px;

border: none;

background: #2a2a3d;

color: #e0e0e0;

}

.composer input:focus {

outline: none;

box-shadow: 0 0 0 2px #4ade80;

}

.composer button {

background: #4ade80;

color: #012;

border: none;

padding: 10px 16px;

border-radius: 8px;

cursor: pointer;

font-weight: 600;

}

.composer button:hover {

background: #22d3ee;

}

::-webkit-scrollbar {

width: 8px;

}

::-webkit-scrollbar-thumb {

background: #555;

border-radius: 4px;

}

::-webkit-scrollbar-track {

background: transparent;

}

</style>