**МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ**

**БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ**

**ФАКУЛЬТЕТ ПРИКЛАДНОЙ МАТЕМАТИКИ И ИНФОРМАТИКИ**

**Кафедра многопроцессорных систем и сетей**

Курс «Распределенные и параллельные системы»

Руководство разработчика

«**QuanThink Wolfram**»

Команда студентов:

Чернышева К.Ю., Костецкий П.С., Демин В.О., Гулин Е.Н.

Минск, 2024

**Руководство разработчика**

**Стек технологий**

Представленная система использует архитектуру клиент-сервер, где фронтенд и бэкенд части разделены для достижения высокой производительности и масштабируемости. Ключевой акцент сделан на эффективности вычислений и взаимодействии в режиме реального времени.

Реализация осуществляется с использованием следующих языков программирования: Java, C++, JavaScript, TypeScript.

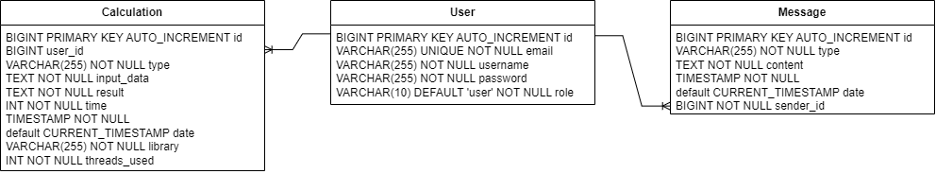
Применяются следующие фреймворки: Spring Boot, Spring WebSocket, Springfox Swagger, React Native, Expo Go.

Основные используемые библиотеки включают: React, Lombok, Spring Data JPA, Spring Security, Apache Commons Math, JMathArray, RxJS, Angular Material и другие.

Создано два клиента: веб-приложение и мобильное приложение. Взаимодействие между клиентской и серверной частями реализовано через RESTful API, созданный на основе Spring Boot. API обеспечивает передачу данных и выполнение операций. Для реализации чата и интерактивного взаимодействия используется технология WebSocket. Развертывание и управление серверами осуществляется посредством контейнеризации с помощью Docker.

Проект сочетает мощную серверную логику на Java и высокопроизводительные вычисления на C++ с удобным и отзывчивым клиентским интерфейсом на Angular. Использование Swagger для документирования делает систему гибкой, безопасной и удобной для разработчиков и пользователей.

**Схема базы данных**

****

**Инструкция по развертыванию и настройке проекта для локального запуска (включая docker)**

Развертывание и настройка проекта

**1. Клонирование репозитория**

Клонируйте репозиторий проекта на ваш локальный компьютер: https://github.com/xenia155/QuanthinkWolfram/

**2. Установка зависимостей**

Перейдите в ветку с Angular приложением:

checkout frontend-dev

Установите зависимости Angular приложения:

npm install

npm run build

npm start

**При работе с мобильным клиентом:**

checkout frontend-mobile

Замените <http://localhost:8080> в файлах, находящихся в QuanthinkWolfram\services\, на http://<ip-адрес>:8080, где <ip-адрес> — это ваш локальный IP-адрес.

npm install

Установите Expo go на вашем мобильном устройстве.

**3. Настройка базы данных**

Убедитесь, что у вас установлен и запущен MySQL на вашем компьютере. Создайте базу данных quanthink\_wolfram:

CREATE DATABASE quanthink\_wolfram;

Таблицы будут создаваться сами в автоматическом режиме.

Откройте файл application.properties в Spring Boot приложении и обновите настройки подключения к базе данных:

spring.datasource.url=jdbc:mysql://localhost:3306/quanthink\_wolfram spring.datasource.username=root

spring.datasource.password=11111111

**4. Запуск проекта**

Запустите Spring Boot приложение (ветка backend):

mvn spring-boot:run

Запустите Angular приложение в отдельном терминале (ветка frontend-dev):

ng serve -o

Теперь ваш проект должен быть доступен по адресу http://localhost:4200.

**При работе с мобильным клиентом:**

Запустить приложение в терминале:

npx expo start

После этого открыть приложение Expo Go на мобильном устройстве, нажать Scan QR code и отсканировать с помощью телефона в терминале QR код.

После этого на мобильном устройстве запустится мобильный клиент.

**5. Использование Docker**

Для запуска проекта в контейнерах Docker выполните следующие шаги:

Установите Docker на вашем компьютере, если еще не сделали.

Создайте Dockerfile для Spring Boot приложения в директории backend со следующим содержимым:

FROM openjdk:22-jdk

COPY target/QuanthinkWolfram-0.0.1-SNAPSHOT.jar .

EXPOSE 8080 ENTRYPOINT ["java", "-jar", "QuanthinkWolfram-0.0.1-SNAPSHOT.jar"]

Создайте Dockerfile для Angular приложения в директории frontend со следующим содержимым:

FROM node:alpine

WORKDIR /usr/src/app

COPY . /usr/src/app

RUN npm install -g @angular/cli

RUN npm install CMD ["ng", "serve", "--host", "0.0.0.0"]

Постройте Docker-образы для обоих приложений:

docker build -t angular-docker .

docker images

Создаем docker-compose.yml такого содержания:

version: '3.6'

services:

mysql:

image: mysql:latest

environment:

MYSQL\_DATABASE: quanthink\_wolfram

MYSQL\_ROOT\_USERNAME: root

MYSQL\_ROOT\_PASSWORD: 11111111

ports:

- "3307:3306"

volumes:

- /home/Docker/Volumes/Mysql/lib/mysql

networks:

- app-network

healthcheck:

test: ["CMD", "mysql", "-uroot", "-p11111111", "-e", "select 1"]

interval: 1s

retries: 120

app:

image: new-spring-boot-app:latest

build:

context: ./QuanthinkWolfram-backend

dockerfile: Dockerfile

environment:

SPRING\_DATASOURCE\_URL: jdbc:mysql://mysql/quanthink\_wolfram

SPRING\_DATASOURCE\_USERNAME: root

SPRING\_DATASOURCE\_PASSWORD: 11111111

ports:

- 8080:8080

networks:

- app-network

depends\_on:

mysql:

condition: service\_healthy

app-ui:

image: new-angular-front-end:latest

build: ./QuanthinkWolfram-frontend-dev

ports:

- 4200:4200

depends\_on:

- app

networks:

- app-network

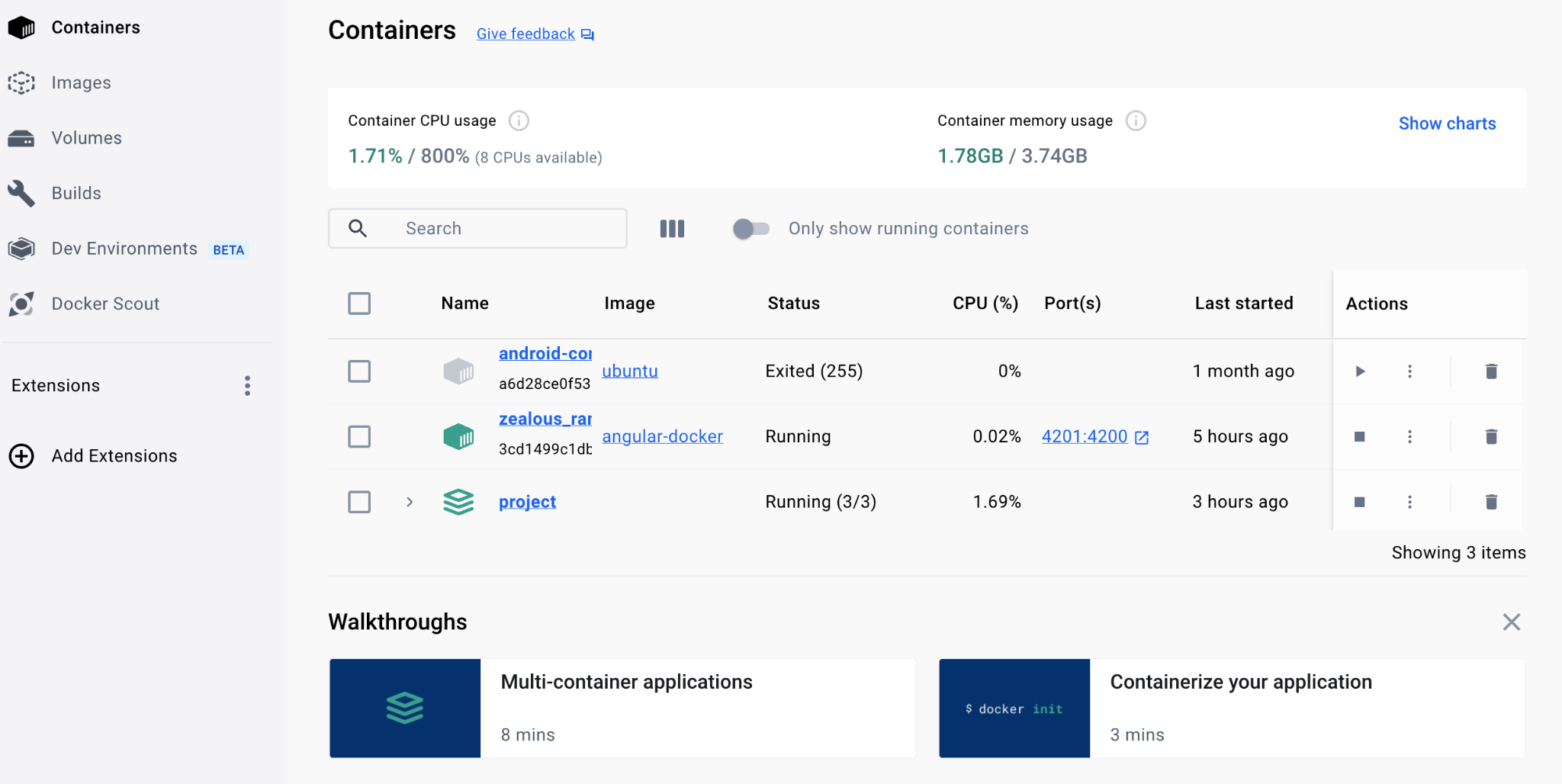
networks:

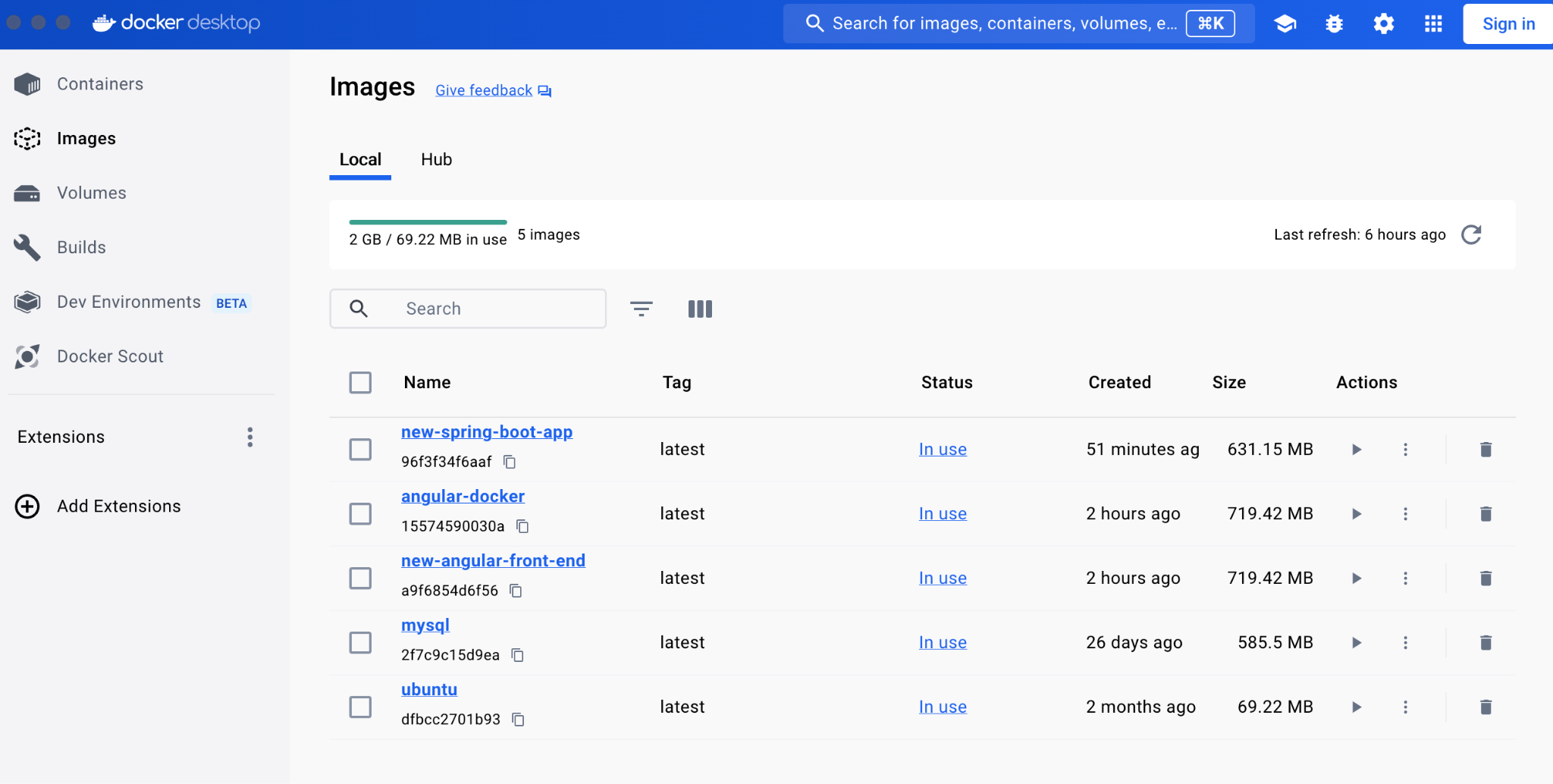
app-network:

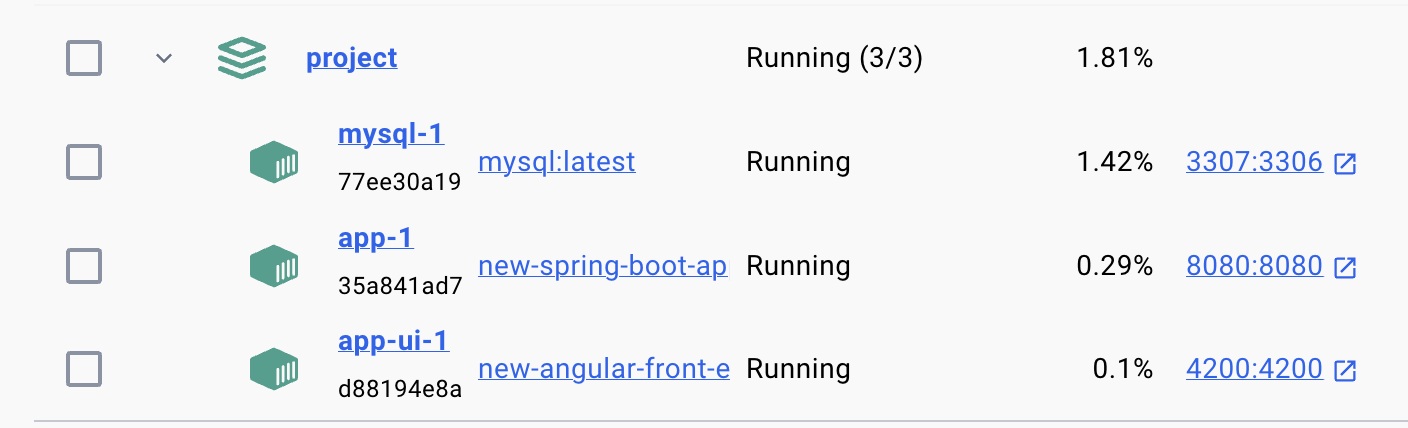
driver: bridge

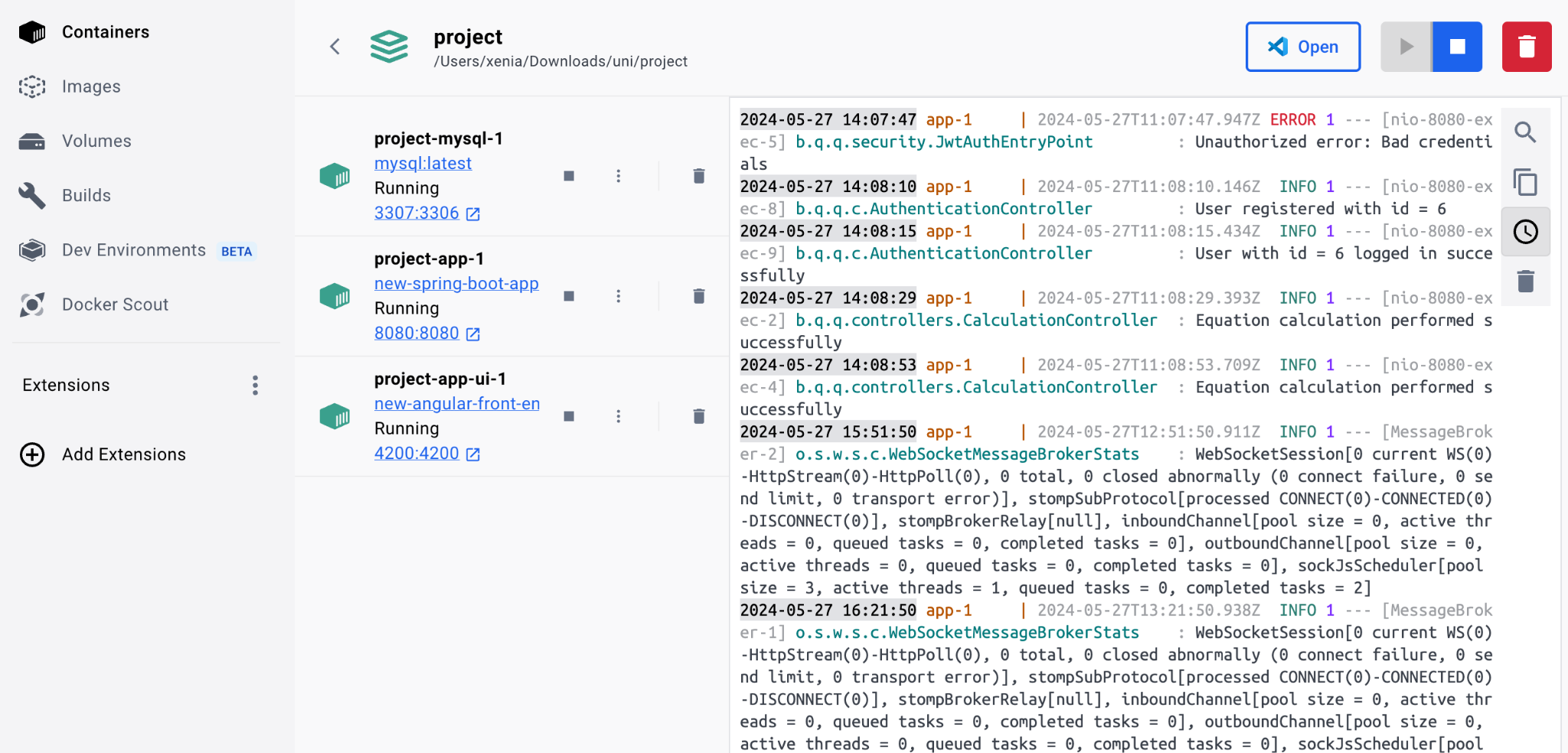
Запустите все вместе:

docker-compose up

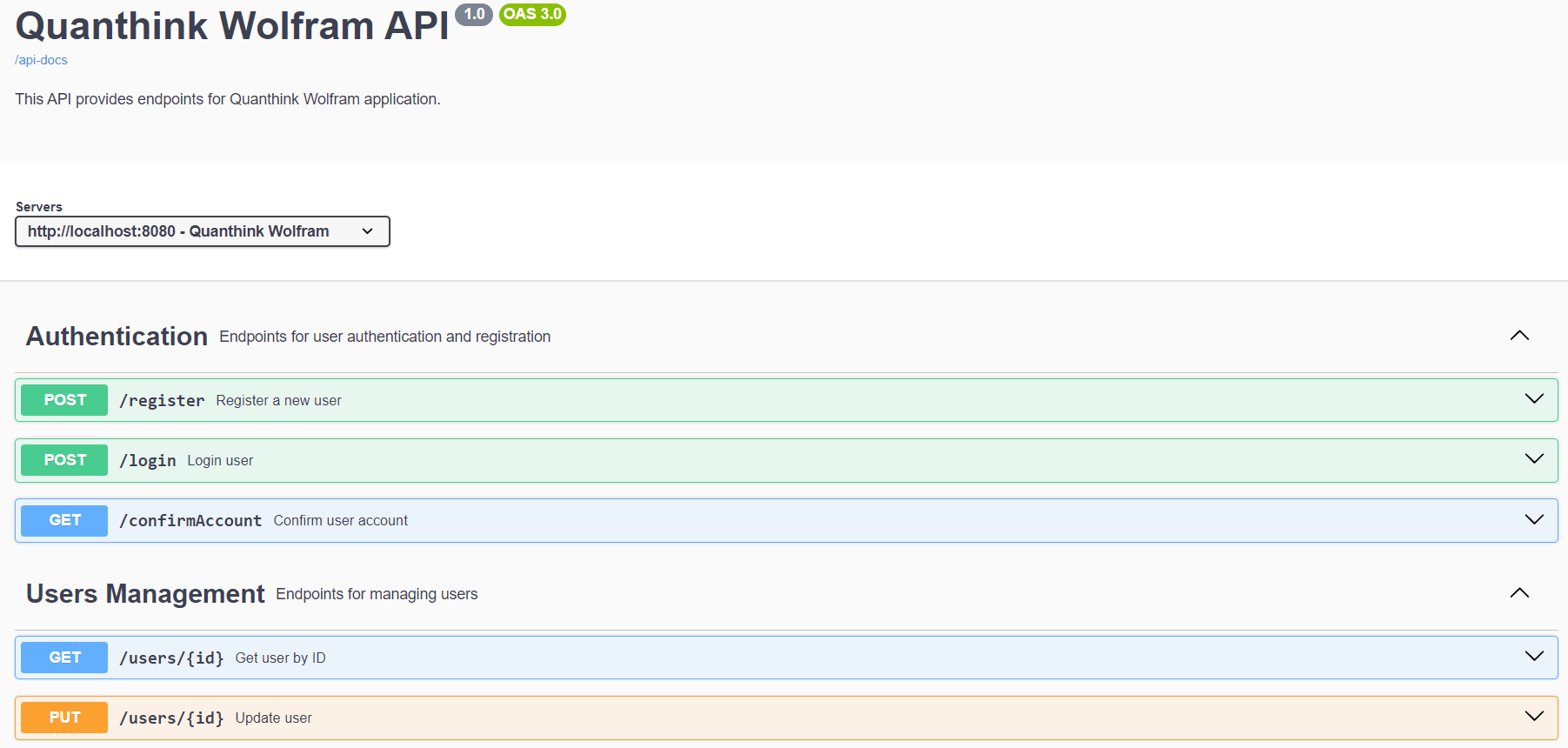






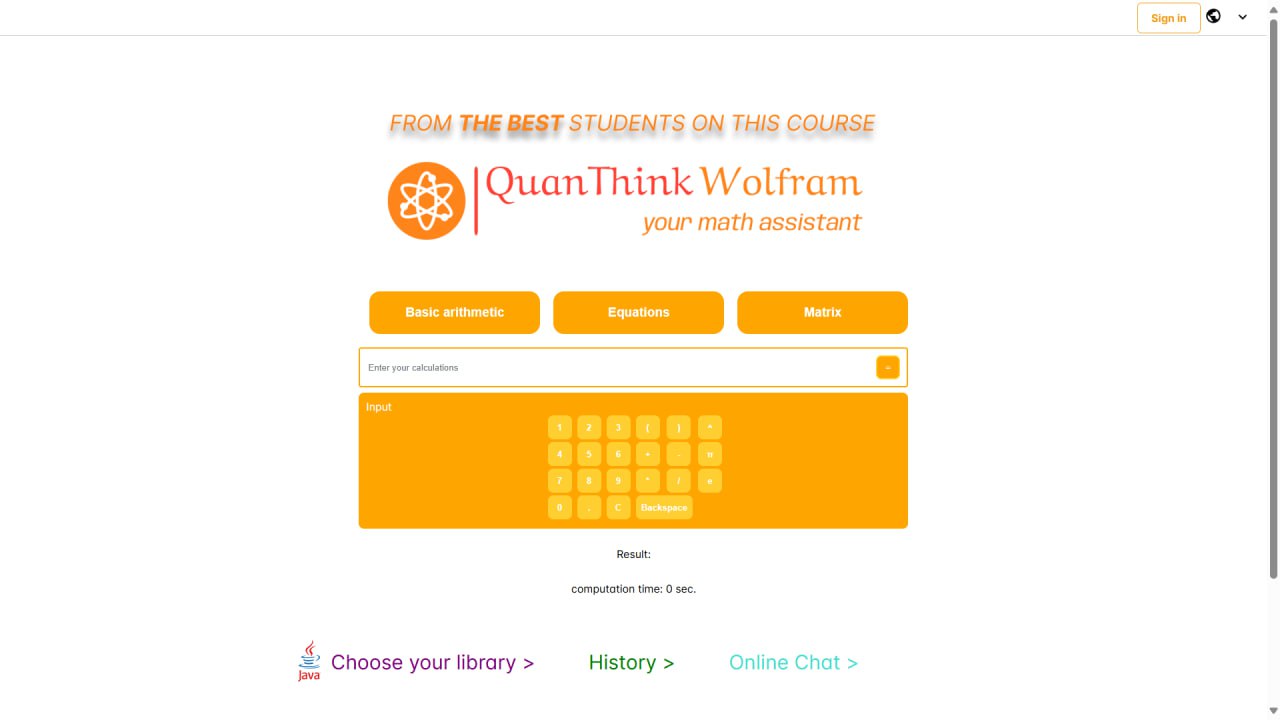


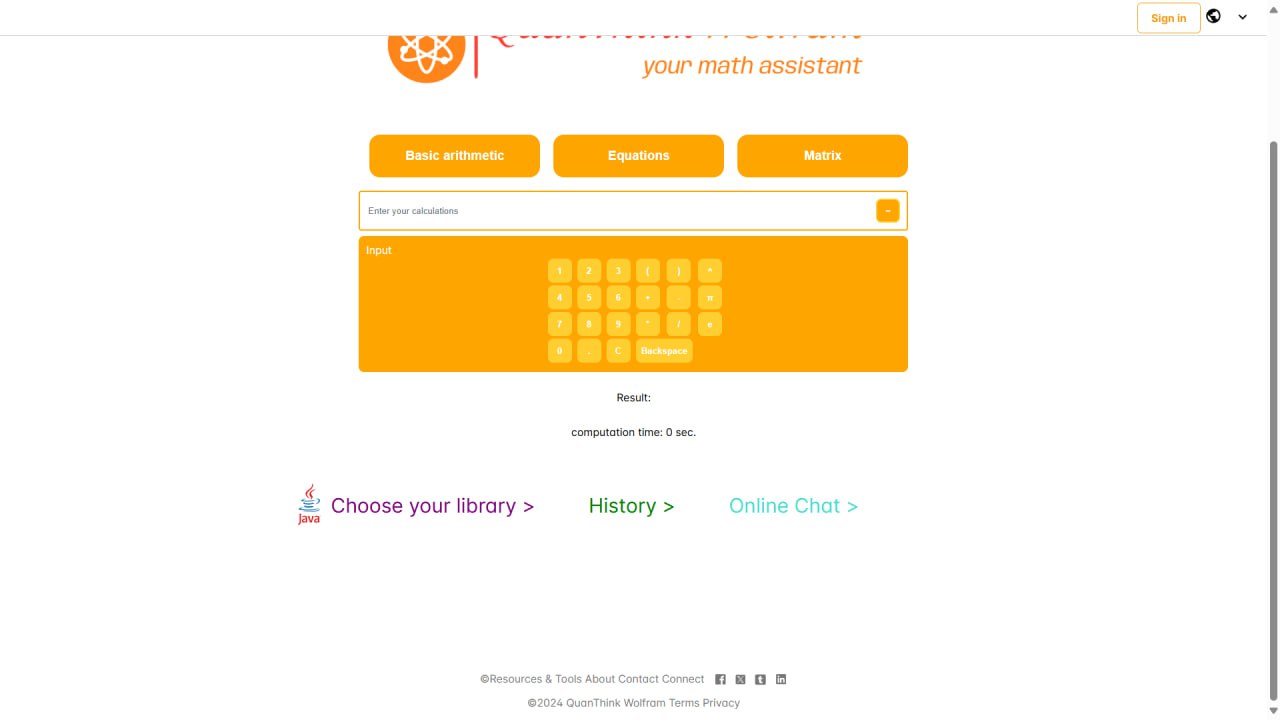
**Документирование API - swagger.**

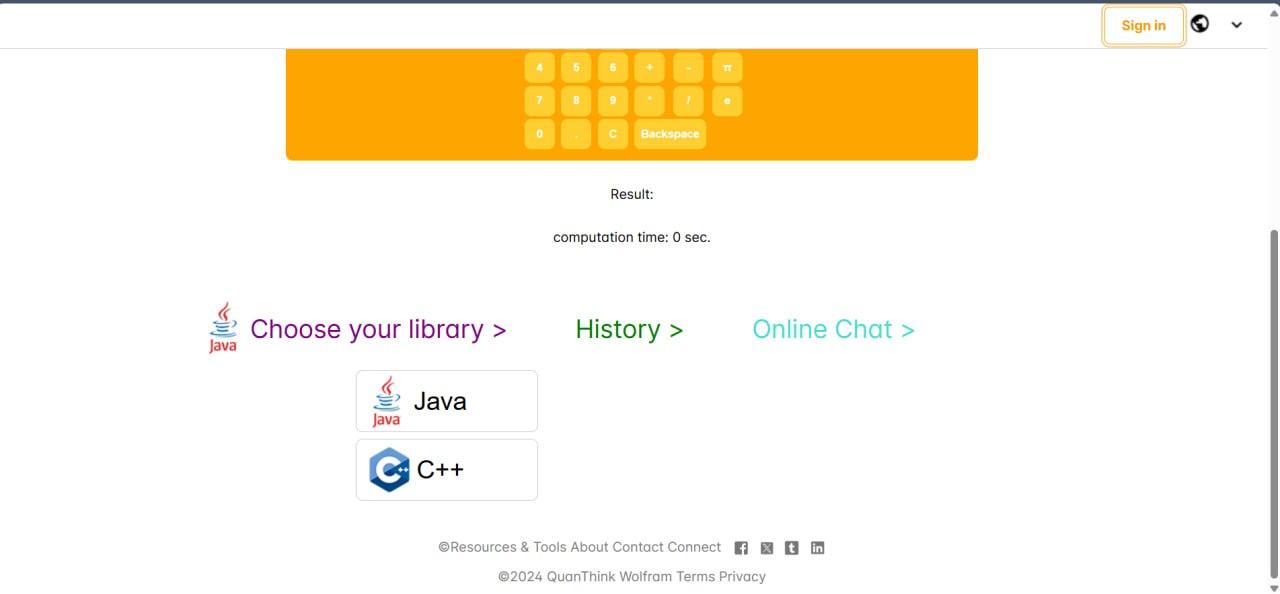
****

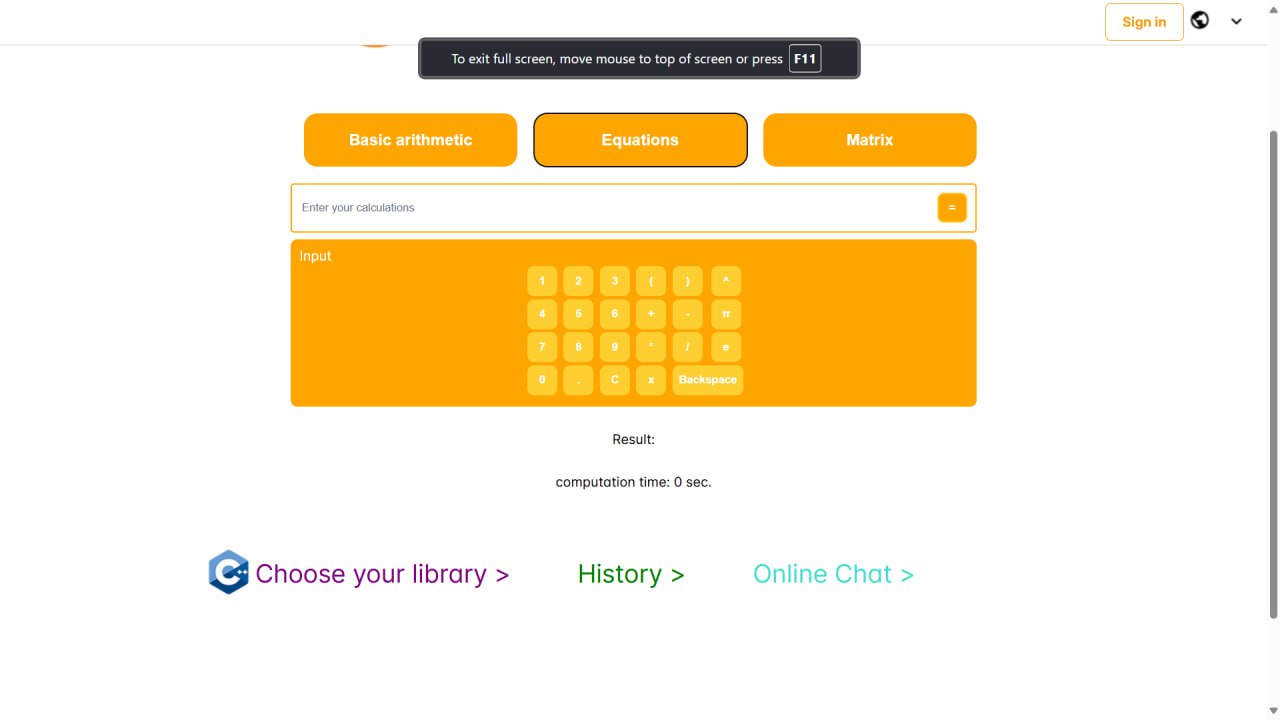
****

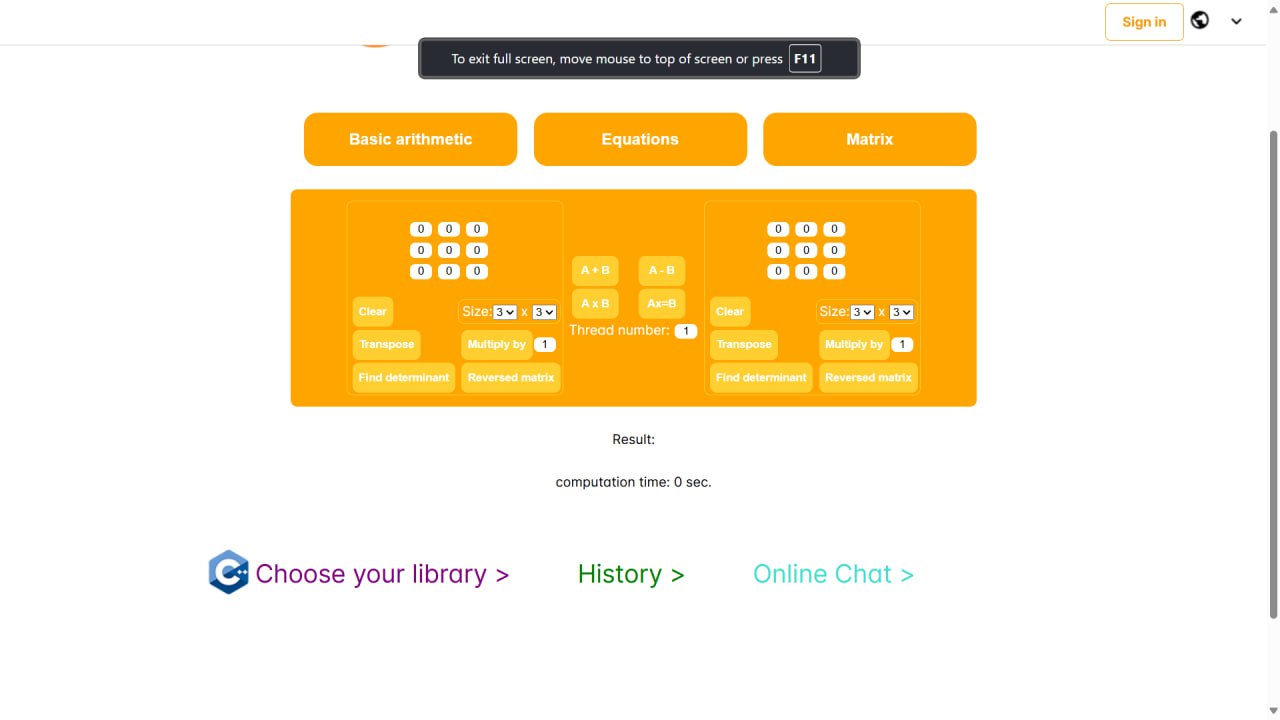
**Скриншоты работы приложения (экран login, главной страницы, вычислений, истории, чата)**

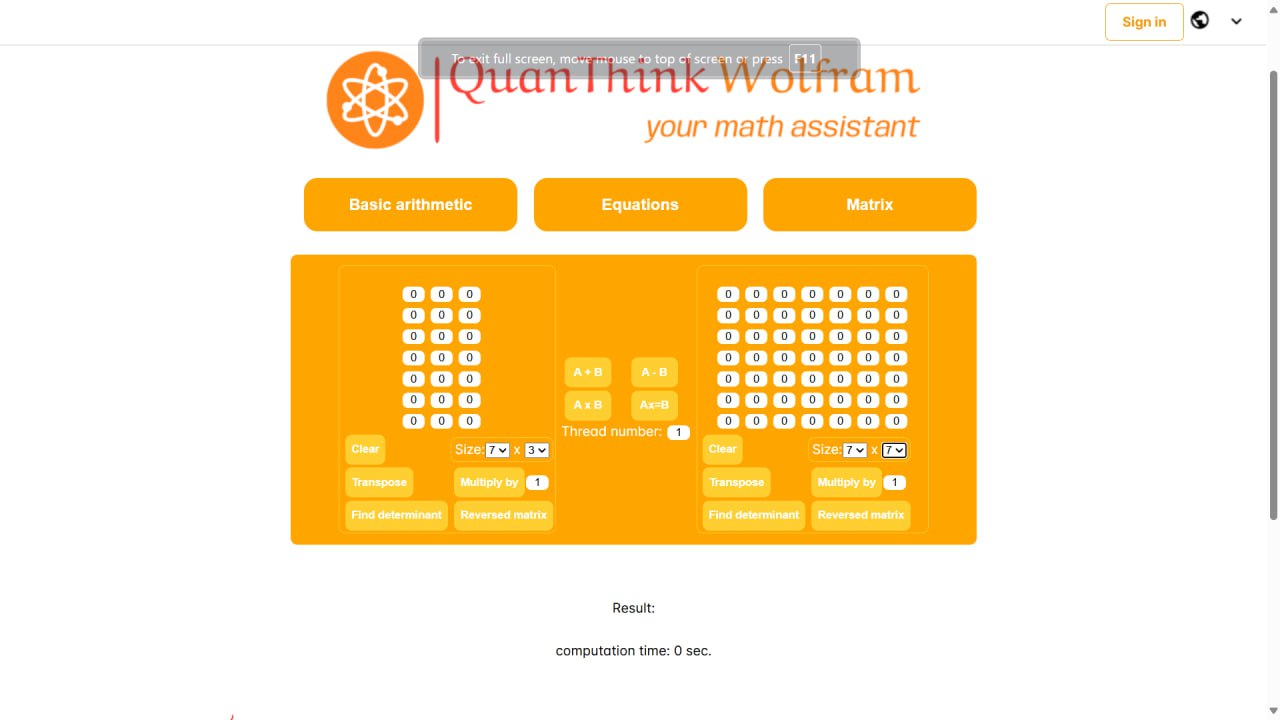
****

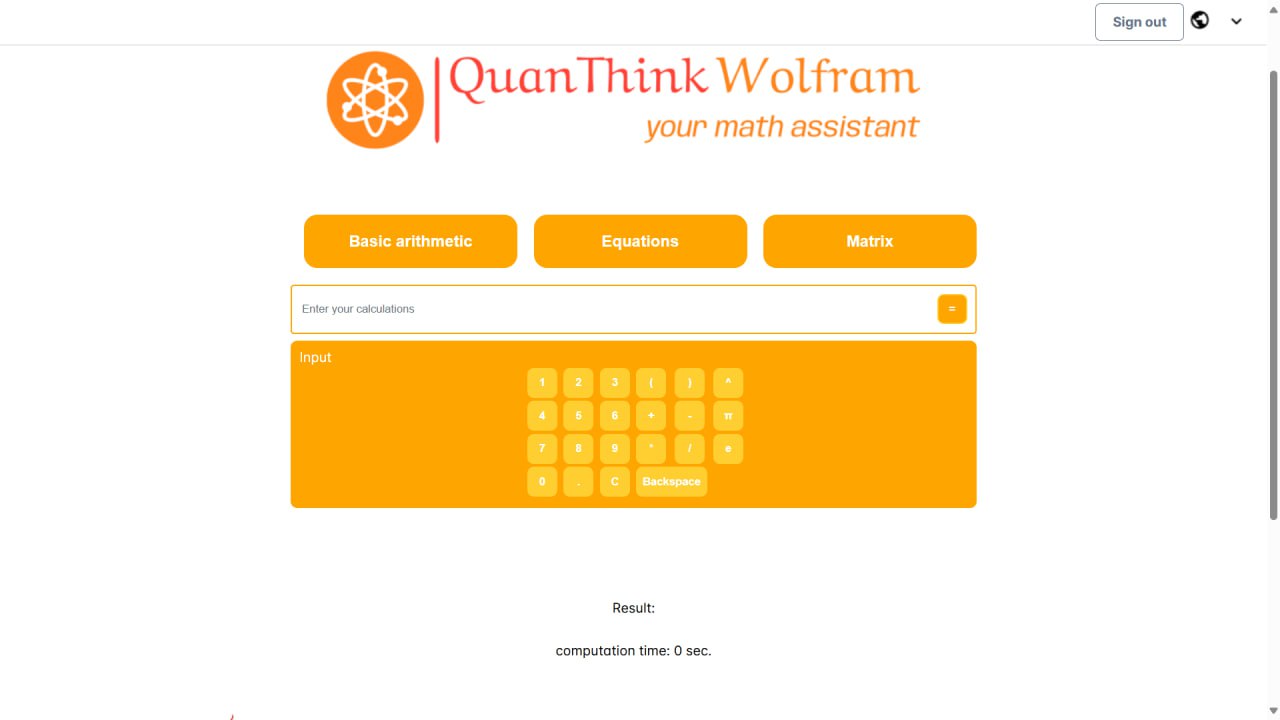
****

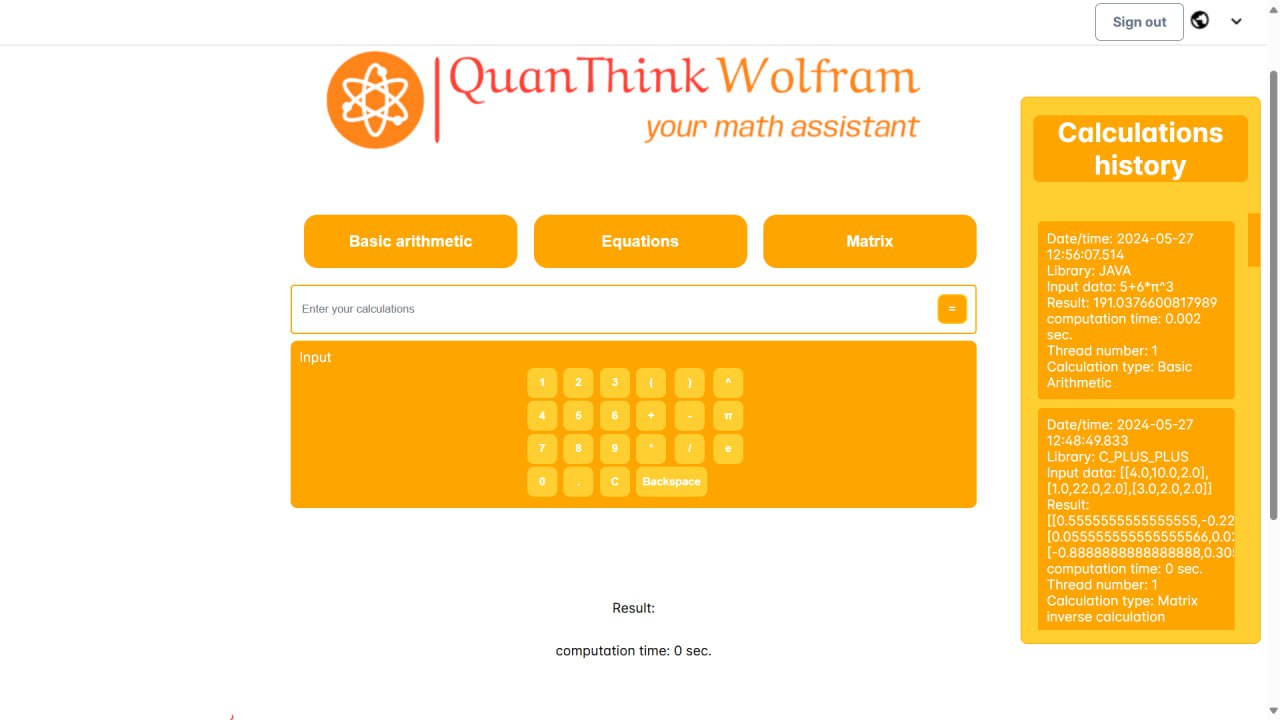
****

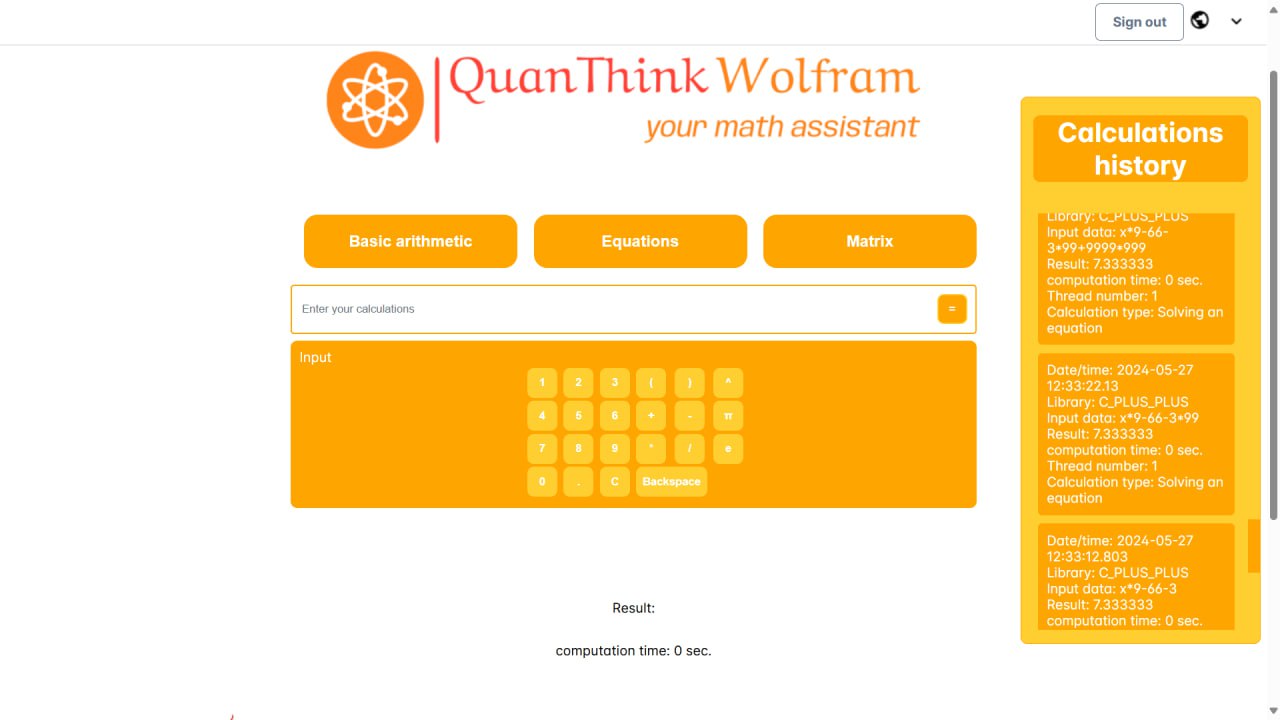
****

****

****

****

****

****

**Пример логов**

****

Полный файл логов можно посмотреть на гугл диске в папке Logs.