Изображение выглядит как текст, снимок экрана, программное обеспечение

Контент, сгенерированный ИИ, может содержать ошибки.

**Корневые папки и файлы**

**flask\_prog**—основная логика:

app.py — главный файл

Модель User хранит данные пользователей:

* + id, name, password
  + games (количество сыгранных игр),
  + point (очки за правильные ответы),
  + all\_ans (всего ответов).
* **type=start**: запускает игру, возвращая данные через get\_gamemode.
* **type=ending**: обновляет статистику пользователя после игры (увеличивает games, point, all\_ans).

**/api/game-map**

* Возвращает **изображение карты** (через Яндекс.Карты API) по координатам lat, lon.

**/authorization (POST/GET)**

* **Логин (type=login)**:
  + Проверяет пароль (сравнивает хэши через check\_password\_hash).
  + Устанавливает куки login и password (хэш).

**/user/info (GET)**

* Возвращает статистику пользователя:
  + games — количество игр,
  + point — очки,
  + KD — соотношение правильных ответов к общему числу.

**Фронтенд-роуты**

* Рендерят HTML-страницы:
* Доступ к user только при наличии кук.

config.py — ключи API

game.py — логика игры.

Функции:

**get\_gamemode**

Возвращает данные в зависимости от выбранного игрового режима (mode).

Использует словарь responses, где:

* + 'default' вызывает get\_city(5)
  + 'infinity' вызывает get\_city(1)

**get\_response\_image**

Кодирует изображение из файла (image\_path) в **Base64** (текстовый формат для передачи изображений).

**get\_img**

Генерирует **статичную карту** через API Яндекс.Карт, используя координаты (lat, lon).

Добавляет небольшой случайный сдвиг к координатам (±0.016), чтобы избежать названия города на изображении.

Формирует URL запроса

Возвращает содержимое **изображения**.

lol.py — модуль, где содержатся словарь со всеми городами и их координатами и функция, выбирающая вперемешку города.

**instance**

default\_base.db —база данных (с id , name, password, games, point, all\_ans , PRIMARY KEY (id)) данные о пользователях

**static**

css — стили для разных страниц:

* + - * authorization.css — оформление формы входа/регистрации.
      * game.css — стили для игрового интерфейса.
      * homeStyle.css — главная страница
      * sty.css – общий файл
      * user.css — профиль пользователя.

img —изображения для сайта (aboba.png-изображение для кнопки возврата на главную страницу, home.jpg- фон).

Js — скрипты:

* + - * game\_script.js — логика игры.

function makeRequest

Универсальная функция для HTTP-запросов (GET/POST).

Отправляет данные в JSON, обрабатывает ошибки.

**callbackBtn(url**

Загружает изображение карты с сервера и отображает его в элементе #map-image.

**function arraysEqual**

Сравнивает два массива, возвращает количество совпадений.

Используется для проверки правильных ответов.

**processDataLater**

Сохраняет правильные ответы (rAnswers) из полученных данных.

Загружает первую карту через callbackBtn.

* + При клике на кнопку :
    - Добавляет ответ пользователя в массив answers.
    - Загружает следующую карту (если вопросы не закончились).
    - После 5 ответов:
      * Отправляет результаты на сервер (/api/game?type=ending).
      * Выводит статистику (количество правильных ответов, KD-ratio).
      * login.js / registration.js — обработка форм входа/регистрации.
      * user\_profile.js — взаимодействие с профилем.

**templates**

HTML-шаблоны, создают внешний вид страницы.