Министерство науки и высшего образования Российской Федерации

федеральное государственное автономное   
образовательное учреждение высшего образования   
«Самарский национальный исследовательский университет   
имени академика С.П. Королева»

Институт информатики, математики и электроники

Факультет информатики

Кафедра технической кибернетики

Отчет по лабораторной работе №0

Дисциплина: «Технологии сетевого программирования»

Тема «Проектирование приложения»

Выполнила: Залевская В.Ю.

Группа: 6302-010302D

Самара, 2025

Задание на лабораторную работу.

1. Проектирование архитектуры:
   * Разработать схему взаимодействия компонентов (сервер, БД, клиент);
   * Разработать логическую схему базы данных (использование, как минимум, 3-х сущностей и 1-ой связи many-to-many);
   * Определить структуру API (REST, методы, URL, параметры, форматы запросов и ответов).
2. Создание Git-репозитория:
   * Инициализировать репозиторий (рекомендуется использовать GitHub);
   * Настроить базовую структуру проекта.
   * Добавить .gitignore для исключения ненужных файлов (виртуальное окружение, логи, артефакты сборки и т. д.).
3. Подготовка окружения:
   * Установить Java/Python;
   * Настроить виртуальное окружение и зависимости.
4. Оформление документации:
   * Подготовить отчет с результатами работы.
5. **Архитектура**

Клиент Сервер База данных

1. **Структура API**

\* POST /auth/register Зарегистрироваться

\* POST /auth/confirm Подвердить регистрацию с кодом

\* GET /habits Получить список всех привычек

\* POST /habit Создать новую привычку

\* PUT /habit/{id} Изменить привычку по id

1. **Схема взаимодействия**
2. **Клиент (Frontend)**

Это веб-интерфейс, с которым взаимодействует пользователь.

Функции:

* Отображение интерфейса (списки привычек, прогресс, статистика).
* Отправка запросов на сервер (например, добавление новой привычки, обновление статуса).
* Получение данных от сервера (например, список привычек, история выполнения).

1. **Сервер (Backend)**

Обрабатывает запросы от клиента, взаимодействует с базой данных и возвращает результаты.

Функции:

* Прием HTTP-запросов от клиента.
* Валидация данных.
* Логика приложения (например, подсчет прогресса привычек).
* Взаимодействие с базой данных (запросы на чтение/запись).
* Отправка ответов клиенту (JSON, XML и т.д.).

1. **База данных (Database)**

Хранилище данных о пользователях, привычках, прогрессе и т.д.

Функции:

* Хранение данных о пользователях (логины, пароли, профили).
* Хранение данных о привычках (название, описание, частота, цель).
* Хранение истории выполнения привычек (даты, статус выполнения).

1. **Схема БД**

**Сущности**

1. **Пользователь (User)**:

Хранит информацию о пользователях.

Поля:

* id (Primary Key)
* username (уникальное имя пользователя)
* email (уникальный email)
* password\_hash (хэш пароля)
* created\_at (дата регистрации)

1. **Привычка (Habit)**:

Хранит информацию о привычках.

Поля:

* id (Primary Key)
* user\_id (Foreign Key, связь с пользователем)
* name (название привычки)
* description (описание привычки)
* frequency\_id (частота выполнения, например, "ежедневно")
* goal (цель, например, "30 дней подряд")
* created\_at (дата создания привычки)

1. **Отметка о выполнении (Completion)**:

Хранит информацию о выполнении привычек.

Поля:

* id (Primary Key)
* habit\_id (Foreign Key, связь с привычкой)
* completed\_at (дата и время выполнения привычки)
* status (статус выполнения, например, "выполнено" или "пропущено")

1. **Категория (Category)**:

Хранит категории привычек (например, "Здоровье", "Спорт", "Образование").

Поля:

* id (Primary Key)
* name (название категории)

1. **Частота (Frequency)**

Хранит информацию о том, как часто нужно выполнять привычку.

Поля:

* id (Primary Key)
* name (название частоты)
* value (Числовое значение частоты (например, 1 для ежедневно, 7 для еженедельно)

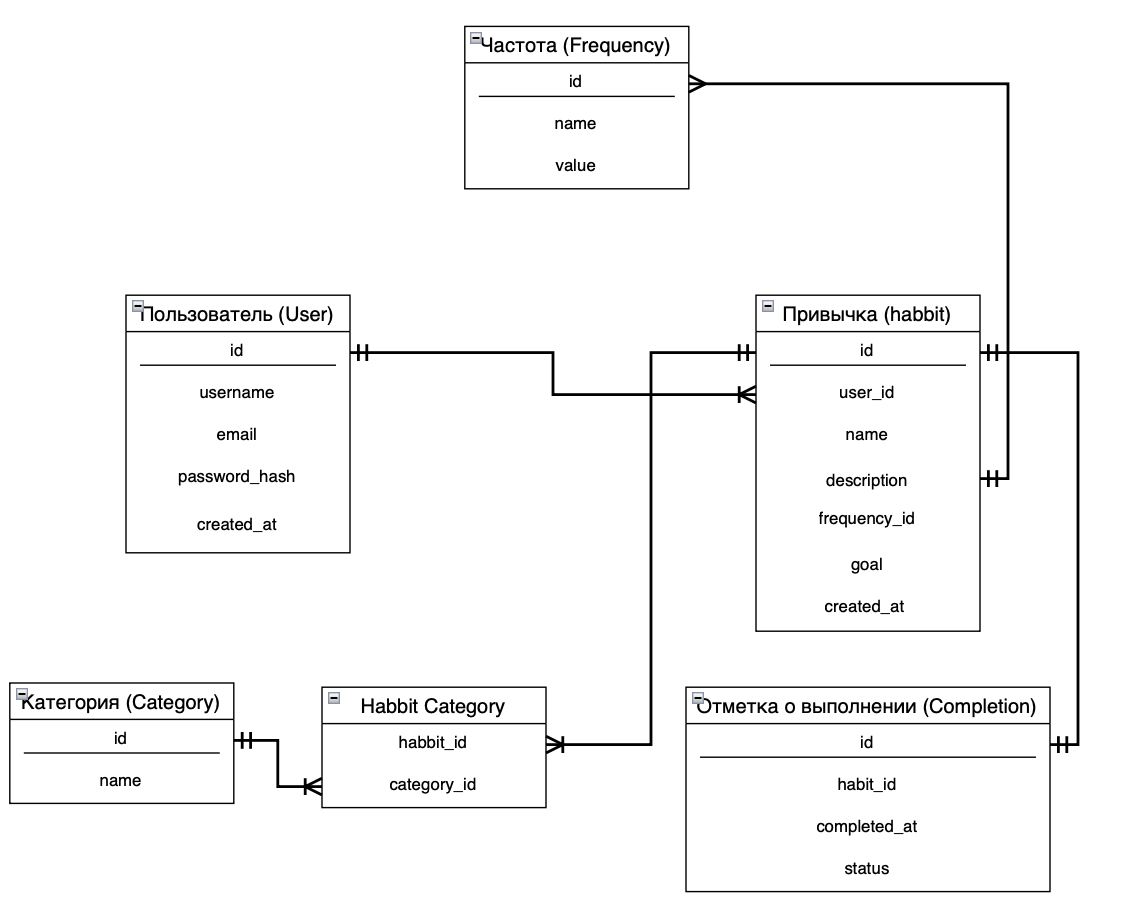
1. **Промежуточная таблица (HabitCategory)**

Хранит категории привычек

Реализует связь many-to-many между таблицами Habits и Categories.

Поля:

* habit\_id (Foreign Key, ссылается на id в таблице Habits)
* category\_id (Foreign Key, ссылается на id в таблице Categories)



1. **Стек технологий**

**1. Frontend (Клиентская часть)**

* Язык программирования: JavaScript.
* Фреймворк: React.js (или Vue.js, Angular)

**2. Backend (Серверная часть)**

* Язык программирования: Python
* Фреймворк: Django (или Flask/FastAPI)

**3. База данных**

* Реляционная СУБД: PostgreSQL

Стек технологий будет меняться и обновляться по мере выполнения проекта