**Проект: Приложение для**

Задание 1: спроектировать схему базы данных для выбранного проекта. Схема должна включать в себя модели с полями, а также связи моделей. Должен быть указан тип поля и данные которые он будет в себе хранить.

**1. Схема базы данных проекта**

**Модель «Users»**

Поля:

* id SERIAL PRIMARY KEY (целочисленный, первичный ключ, автоинкрементируемый)
* name VARCHAR (35) NOT NULL UNIQUE (строковый, не пустой, уникальный (35))
* email VARCHAR (35) NOT NULL UNIQUE (строковый, не пустой, уникальный (35))
* password VARCHAR 20) NOT NULL CHECK (password BETWEEN 8 AND 20) (строковый, не пустой (8 - 20))
* fk\_user\_profile INT **UNIQUE** (целочисленный)

Связи:

* **Один ко многим** с моделью «Users\_Tasks»
* **Многие ко многим** с моделью «Tasks» через модель «Users\_Tasks»
* FOREIGN KEY (fk\_user\_profile) REFERENCES Profiles (id) (**один к одному** с моделью «Profiles»)

**Модель «Profiles»**

Поля:

* id SERIAL PRIMARY KEY (целочисленный, первичный ключ, автоинкрементируемый)
* phone VARCHAR (15)
* status VARCHAR (7) NOT NULL CHECK(status IN(‘ОБЫЧНЫЙ, ‘ПРЕМИУМ’ (строковый, не пустой (7))
* avatar VARCHAR (255) (строковый (255)) – ссылка или путь к файлу с изображением
* task\_complited INT
* profile VARCHAR (255) (строковый (255))

Связи:

* **один к одному** с моделью «Users»)

**Модель «Tasks»**

Поля:

* id SERIAL PRIMARY KEY (целочисленный, первичный ключ, автоинкрементируемый)
* name VARCHAR (50) NOT NULL (строковый, не пустой (50))
* description VARCHAR (255) NOT NULL (строковый, не пустой (255))
* file BYTEA (информация из файла)
* status VARCHAR (15) NOT NULL CHECK(status IN(‘ЗАПЛАНИРОВАНА’, ‘В РАБОТЕ’, ‘ВЫПОЛНЕНА’, ‘ОТМЕНЕНА’ (строковый, не пустой (15))
* date\_of\_create TIMESTAMP NOT NULL DEFAULT CURRENT\_TIMESTAMP
* deadline TIMESTAMP NOT NULL

Связи:

* **Один ко многим** с моделью «Users\_Tasks»
* **Многие ко многим** с моделью «Users» через модель «Users\_Tasks»

**Модель «Users\_Tasks»**

Поля:

* id SERIAL PRIMARY KEY (целочисленный, первичный ключ, автоинкрементируемый)
* user\_id INT (ссылка на id пользователя в таблице пользователей)
* task\_id INT (ссылка на id задачи в таблице задач)

Связи:

- FOREIGN KEY (user\_id) REFERENCES Users (id) (**многие к одному** с моделью «Users»)

- FOREIGN KEY (task\_id) REFERENCES Tasks (id) (**многие к одному** с моделью «Tasks»)

Задание 2: сформировать список функциональности системы.

**2. Описание функционала проекта**

1. Регистрация нового пользователя с использованием адреса электронной почты, имени и пароля. Одновременно создается незаполненный профиль пользователя.

2. Авторизация пользователя с использованием адреса электронной почты и пароля.

3. Смена пароля пользователем.

4. Удаление учетной записи пользователем. Одновременно удаляется профиль пользователя.

5. Изменение профиля пользователем.

5.1 Просмотр информации о профиле пользователя

6. Создание новой задачи.

7. Просмотр выбранной задачи

8. Изменение выбранной задачи.

8.1 Изменение статусы выбранной задачи.

9. Удаление выбранной задачи.

10. Просмотр всех задач.

11. Добавление файла к задаче

12. Просмотр файла в задаче.

13. Удаление файла из задачи.

Задание 3: подготовить описание эндпоинтов. Описание должно включать URL, метод, краткое описание его назначения, список принимаемых аргументов с указанием обязательное ли это поле или нет, валидацию данных, информацию о том, что будет происходить с этими данными.

**3. Описание эндпоинтов**

**3.1 api/auth/register**

**-** Метод: POST / **register**

- Описание: Регистрация пользователя. Создание профиля незаполненного профиля.

- Входные данные:

* name VARCHAR (35) NOT NULL UNIQUE (строковый, не пустой, уникальный (35))
* email VARCHAR (35) NOT NULL UNIQUE (строковый, не пустой, уникальный (35))
* password VARCHAR 20) NOT NULL CHECK (password BETWEEN 8 AND 20) (строковый, не пустой (20))

- Преобразование данных: Создание новой записи в таблице «Users» с введенными данными

- Запись в таблицу «Users»

- Валидация – проверка формата электронной почты, проверка уникальности email, проверка длины пароля.

**3.2** **api/auth/login**

- Метод: POST / **login**

- Описание: Авторизация пользователя с использованием адреса электронной почты и пароля.

- Входные данные:

* email VARCHAR (35) NOT NULL UNIQUE (строковый, не пустой, уникальный (35))
* password VARCHAR 20) NOT NULL CHECK (password BETWEEN 8 AND 20) (строковый, не пустой (20))

- Преобразование данных: Проверка соответствия введенных данных с данными в БД.

- Запись в таблицы: Нет.

- Валидация – проверка формата электронной почты и наличия пароля.

**3.3 api/users/{userID}/change\_pass**

- Метод: PUT **/users**

- Описание: Смена пароля пользователем.

- Входные данные:

- password VARCHAR 20) NOT NULL CHECK (password BETWEEN 8 AND 20) (строковый, не пустой (20))

- Преобразование данных: Обновление записи в таблице «Users» с указанным ID

- Запись в таблицу «Users»

- Валидация – Проверка длины пароля

**3.4 api/users/{userID}/delete\_user**

- Метод: DELETE **/users**

- Описание: Удаление учетной записи пользователем. Одновременно удаляется профиль пользователя.

- Входные данные: Нет

- Преобразование данных: Удаление данных из таблицы «Users» с указанным ID. Удаление данных из таблицы «Profile» в строке с User\_ID = ID

- Запись в таблицы: «Users», «Profiles»

- Проверка существования пользователя с указанным ID

**3.5 api/users/{userID}/profile**

- Метод: PUT **/users\_put\_profile**

- Описание: Изменение профиля пользователем.

- Входные данные:

- profile VARCHAR (255) (строковый, не пустой (255))

* phone VARCHAR (15)
* status VARCHAR (7) NOT NULL CHECK(status IN(‘ОБЫЧНЫЙ, ‘ПРЕМИУМ’ (строковый, не пустой (7))
* avatar VARCHAR (255) (строковый (255)) – ссылка или путь к файлу с изображением
* task\_complited INT

- Преобразование данных: Обновление данных в таблице «Profiles» ссылающейся на в своем поле User\_ID на ID пользователя

- Запись таблицы «Profiles»

- Валидация - проверка длины текста.

**3.5.1** **api/users/{userID}/profile**

- Метод GET **/users**

- Описание: просмотр профиля пользователя

- Входные данные: ID пользователя

- Преобразование данных: получение данных из таблицы «Profiles»

- Запись в таблицу: Нет.

- Валидация: проверка существования пользователя с указанным ID

**3.6 api/tasks/create**

- Метод: POST **/tasks**

- Описание: создание новой задачи

- Входные данные:

* name VARCHAR (50) NOT NULL (строковый, не пустой (50))
* description VARCHAR (255) NOT NULL (строковый, не пустой (255))
* deadline TIMESTAMP NOT NULL

- Преобразование данных: создание новой записи в таблице «Tasks» с указанными данными

- Запись таблицы «Tasks»

- Валидация данных: проверка наличия названия и дедлайна.

**3.7 api/tasks/{taskID}**

- Метод: GET **/tasks**

- Описание: просмотр задачи указанной в ID

- Преобразование данных: Получение информации о задаче с указанным ID из табл. «Tasks»

- Запись в таблицы: Нет.

- Валидация данных: Проверка существования задачи с указанным ID

**3.8** **api/tasks/{taskID}**

- Метод: PUT **/tasks**

- Описание: Изменение выбранной задачи.

- Входные данные:

* name VARCHAR (50) NOT NULL (строковый, не пустой (50))
* description VARCHAR (255) NOT NULL (строковый, не пустой (255))
* deadline TIMESTAMP NOT NULL

- Преобразование данных: Обновление записи c указанным ID в таблице «Tasks»

- Валидация данных: Проверка наличия названия и дедлайна

**3.8.1 api/tasks/{taskID}**

- Метод: PATCH **/tasks**

- Описание: Изменение статуса выбранной задачи.

- Входные данные:

- status VARCHAR (15) NOT NULL CHECK(status IN(‘ЗАПЛАНИРОВАНА’, ‘В РАБОТЕ’, ‘ВЫПОЛНЕНА’, ‘ОТМЕНЕНА’ (строковый, не пустой (15))

- Преобразование данных: Обновление записи c указанным ID в таблице «Tasks»

- Валидация данных: проверка допустимости значений в поле status в таблице «Tasks»

**3.9 api/tasks/{taskID}**

- Метод: DELETE **/tasks**

- Описание: Удаление задачи

- Входные данные: Нет

- Преобразование данных: Удаление записи с заданным ID из таблицы «Tasks»

- Запись в таблицу: «Tasks»

- Валидация данных: Проверка существования задачи с заданным ID

**3.10 api/tasks**

- Метод: GET **/tasks\_get**

- Описание: Просмотр списка всех задач с возможностью пагинации, фильтрации и поиска

- Входные данные:

* status VARCHAR (15) NOT NULL (опционально, строковый, не пустой (15))
* deadline TIMESTAMP NOT NULL (опционально, дата и время)
* page (опционально, число)
* page\_size (опционально, число)

- Преобразование данных: Получение данных из таблицы «Tasks»

- Запись в таблицу: Нет

- Валидация данных: Проверка наличия названия и дедлайна, проверка допустимости значений в поле status в таблице «Tasks». Проверка допустимости значений page и page\_size.

**3.11 api/tasks/{taskID}/file**

- Метод: PUT **-file**

- Описание: Добавление файла к задаче

- Входные данные: file BYTEA (информация из файла)

- Преобразование данных: Обновление записи c ID в таблице «Tasks»

- Запись в таблицу: «Tasks»

- Валидация данных: проверка отсутствия записей в поле file выбранной задачи. NULL в поле file

**3.12 api/tasks/{taskID}/file**

- Метод: GET **-file**

- Описание: Просмотр файла из задачи по ее ID

- Входные данные: ID задачи

- Преобразование данных: Получение информации о файле в задаче с ID из «Tasks»

- Запись в таблицы: Нет

- Валидация данных: Проверка существования задачи с заданным ID

**3.13 api/tasks/{taskID}/file**

- Метод: DELETE **-file**

- Описание: удаление файла из задачи по ID

- Входные данные: ID задачи

- Преобразование данных: удаление файла из задачи по ее ID

- Запись в таблицу: «Tasks»

- Валидация данных: проверка наличия записи в поле file выбранной задачи.