**МИНОБРНАУКИ РОССИИ**

**САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ**

**ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**

**«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)**

**Кафедра МОЭВМ**

**ИНДИВИДУАЛЬНОЕ ДОМАШНЕЕ ЗАДАНИЕ**

**по дисциплине «Введение в нереляционные базы данных»**

**Тема: Сервис поиска психологов**

| Студентка гр. 1381 |  | Демчук П.Д. |
| --- | --- | --- |
| Студент гр. 1381 |  | Мелькумянц Д.А. |
| Студент гр. 1381 |  | Нгуен Ш.Х. |
| Преподаватель |  | Заславский М.М. |

Санкт-Петербург

2024

**ЗАДАНИЕ**

| Студентка Демчук П.Д.  Студент Мелькумянц Д.А.  Студент Нгуен Ш.Х. | | |
| --- | --- | --- |
| Группа 1381 | | |
| Тема: Сервис поиска психологов | | |
| Исходные данные:  Необходимо реализовать веб-приложение для сервиса поиска психологов с использование СУБД MongoDB. | | |
| Содержание пояснительной записки:  «Содержание»  «Введение»  «Сценарии использования»  «Модель данных»  «Разработанное приложение»  «Выводы»  «Приложения»  «Список использованных источников» | | |
| Предполагаемый объем пояснительной записки:  Не менее 10 страниц. | | |
| Дата выдачи задания: 05.09.2024 | | |
| Дата сдачи реферата: 21.12.2024 | | |
| Дата защиты реферата: 21.12.2024 | | |
| Студентка |  | Демчук П.Д. |
| Студент |  | Мелькумянц Д.А. |
| Студент |  | Нгуен Ш.Х. |
| Преподаватель |  | Заславский М.М. |

**АННОТАЦИЯ**

В рамках ИДЗ разработано веб-приложение, представляющее собой сервис поиска психологов. Приложение включает функционал для просмотра страниц психологов. Реализована система фильтрации и поиска психологов по различным критериям.

Для разработки использованы технологии Vue.js, Python, СУБД MongoDB.

Найти исходный код можно по ссылке: [phs](https://github.com/moevm/nosql2h24-phs).

**SUMMARY**

Within the framework of the IHW, a web application has been developed, which is a service for searching for psychologists. The application includes functionality for viewing pages of psychologists. A system for filtering and searching for psychologists by various criteria has been implemented.

Vue.js, Python, and MongoDB technologies were used for the development.

The source code can be found at the link: [phs](https://github.com/moevm/nosql2h24-phs).

**СОДЕРЖАНИЕ**

| 1. | Введение | 6 |
| --- | --- | --- |
| 1.1. | Актуальность проблемы | 6 |
| 1.2. | Постановка задачи | 6 |
| 1.3. | Предлагаемое решение | 6 |
| 1.4. | Качественные требования к решению | 6 |
| 2. | Сценарии использования | 7 |
| 2.1. | Макет UI | 7 |
| 2.2. | Сценарий использования для импорта данных | 12 |
| 2.3. | Сценарий использования для представления данных | 14 |
| 2.4. | Сценарий использования для анализа данных | 16 |
| 2.5. | Сценарий использования для экспорта данных | 17 |
| 2.6. | Вывод | 17 |
| 3. | Модель данных | 19 |
| 3.1. | Нереляционная модель данных | 19 |
| 3.2. | Аналог модели данных для SQL СУБД | 24 |
| 3.3. | Сравнение моделей | 27 |
| 4. | Разработанное приложение | 29 |
| 5. | Выводы | 31 |
| 6. | Приложения | 33 |
| 7. | Литература | 35 |

**1. ВВЕДЕНИЕ**

**1.1. Актуальность проблемы**

В современном мире все больше людей сталкиваются с эмоциональными и психологическими трудностями. Психическое здоровье стало такой же важной частью жизни, как и физическое. Тем не менее, найти квалифицированного специалиста может быть настоящим вызовом. Веб-приложение для поиска психологов поможет пользователям легко и быстро находить профессионалов в области психологии.

**1.2. Постановка задачи**

Задача проекта заключается в разработке веб-приложения, которое позволяет пользователям:

* Просматривать карточки психологов, модераторов и клиентов;
* Фильтровать психологов по различным параметрам;
* Оставлять отзывы;
* Взаимодействовать с удобным интерфейсом модерирования.

Также требуется обеспечить надежное хранение данных и высокую

производительность приложения.

**1.3. Предлагаемое решение**

Для реализации создается веб-приложение с использованием Vue.js, Python для серверной части и MongoDB для хранения данных. Приложение поддерживает фильтрацию, поиск.

**1.4. Качественные требования к решению**

Решение должно быть удобным, производительным, надёжным и легко расширяемым.**2. СЦЕНАРИИ ИСПОЛЬЗОВАНИЯ**

**2.1. Макет UI**

Ниже представлены макеты страниц приложения.

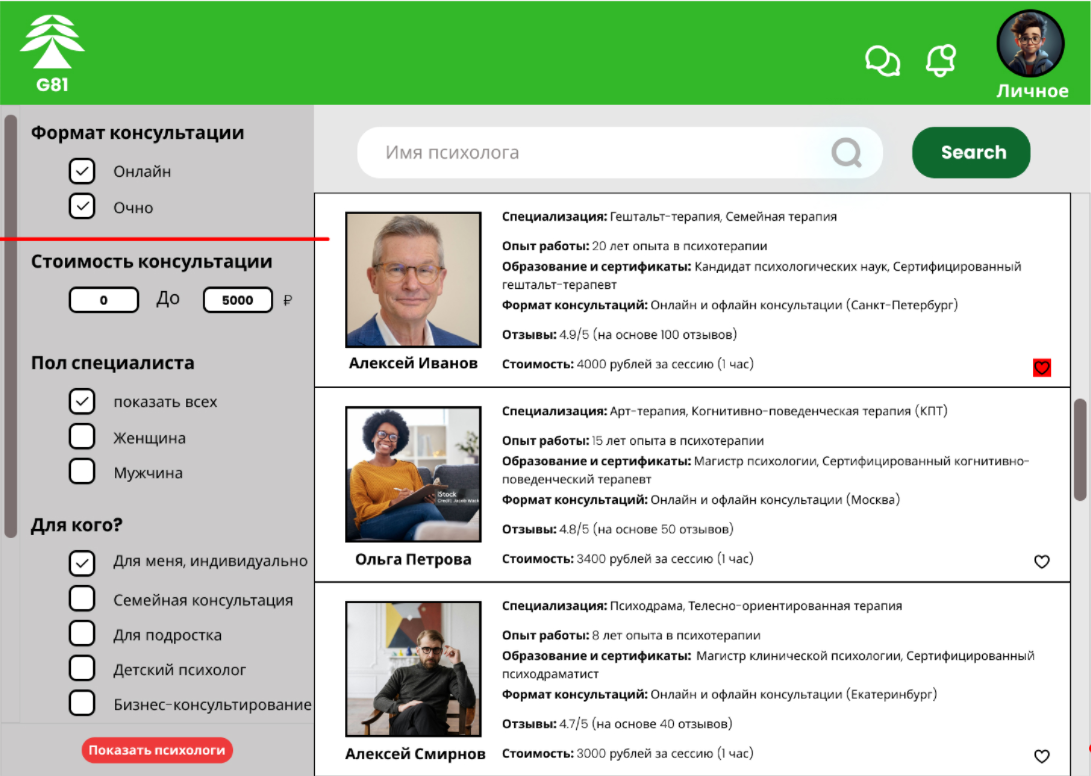
****

Рис. 1. Страница поиска психологов

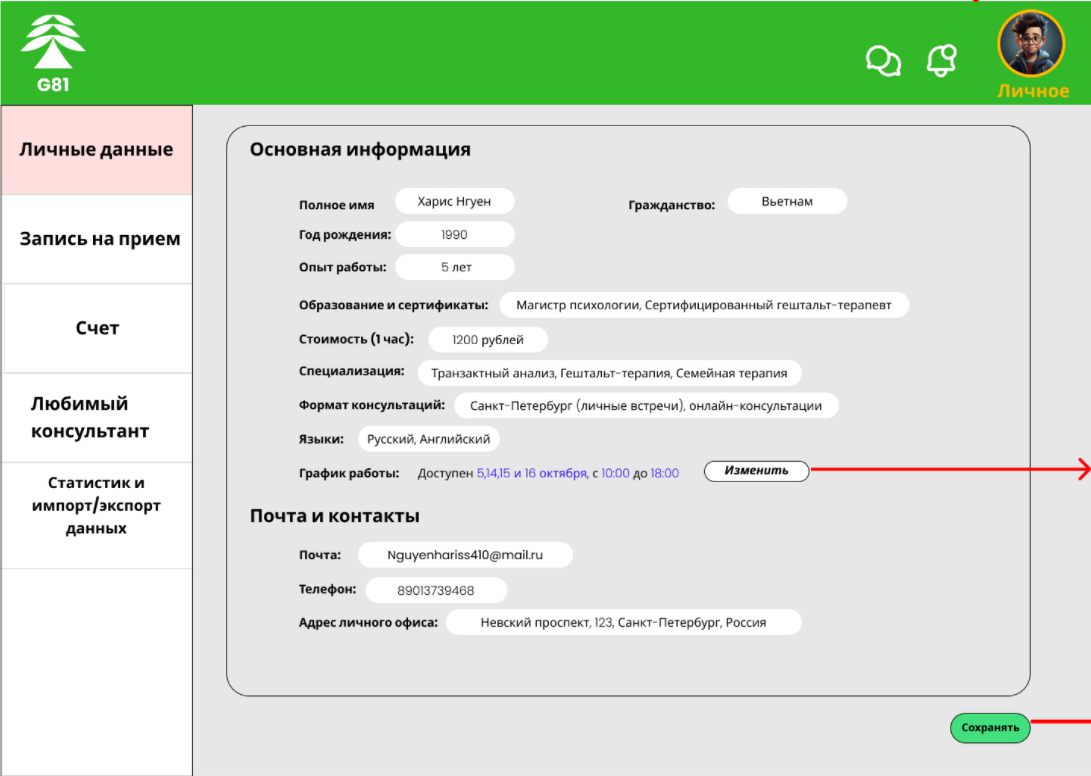
****

Рис. 2. Страница редактирования личной информации

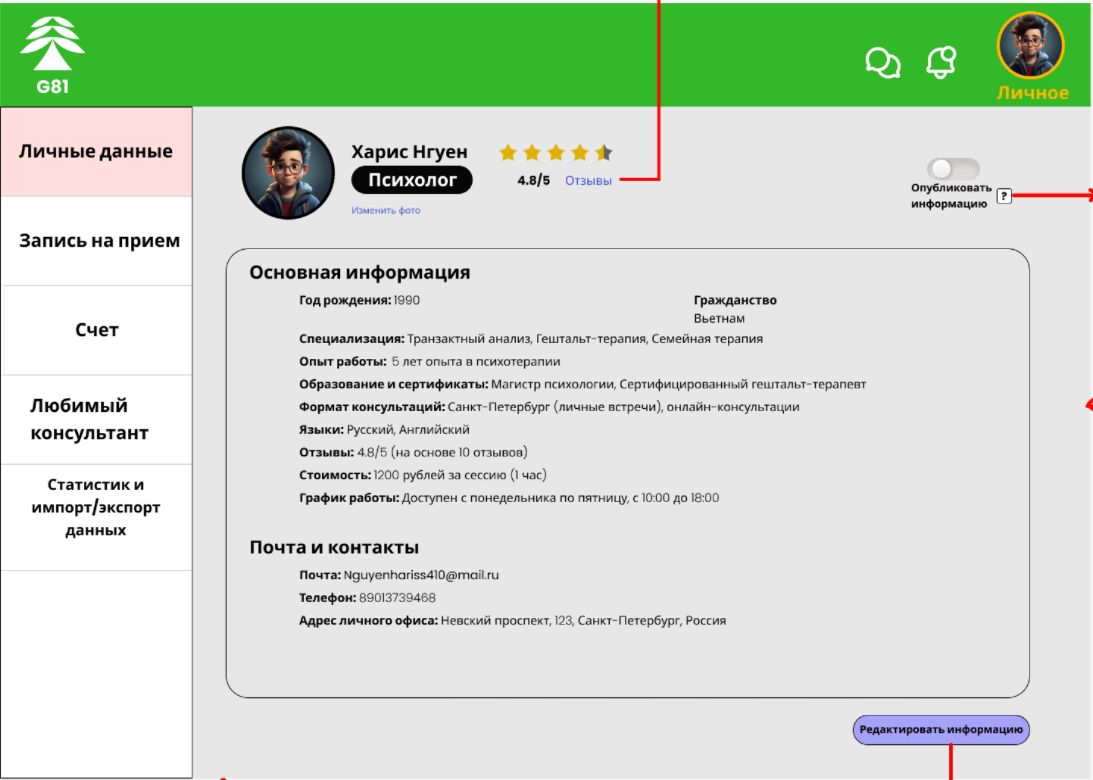
****

Рис. 3. Личные данные

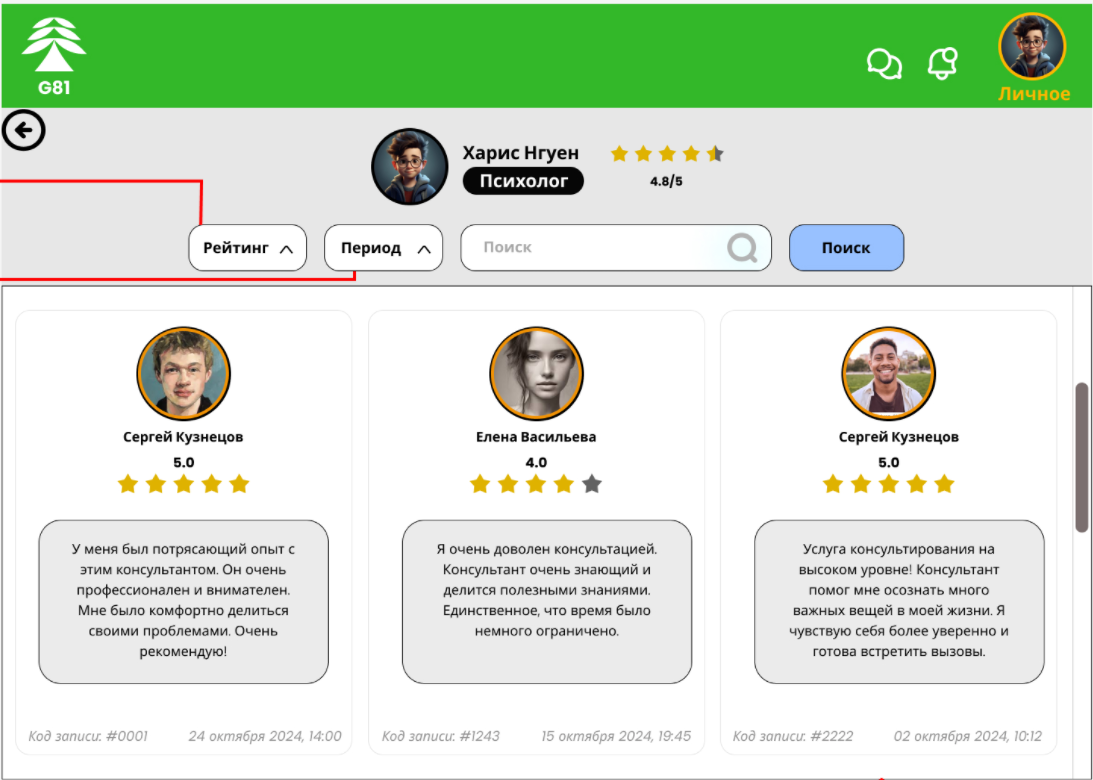
****

Рис. 4. Отзывы

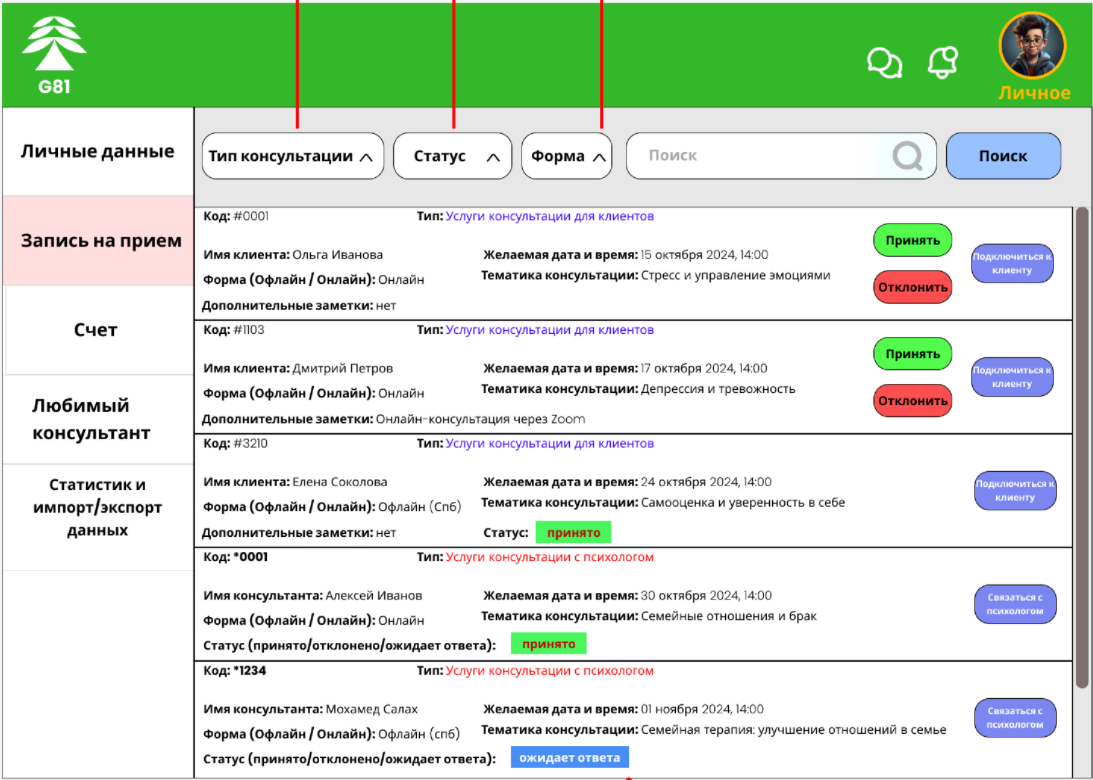
****

Рис.5. Запись на прием

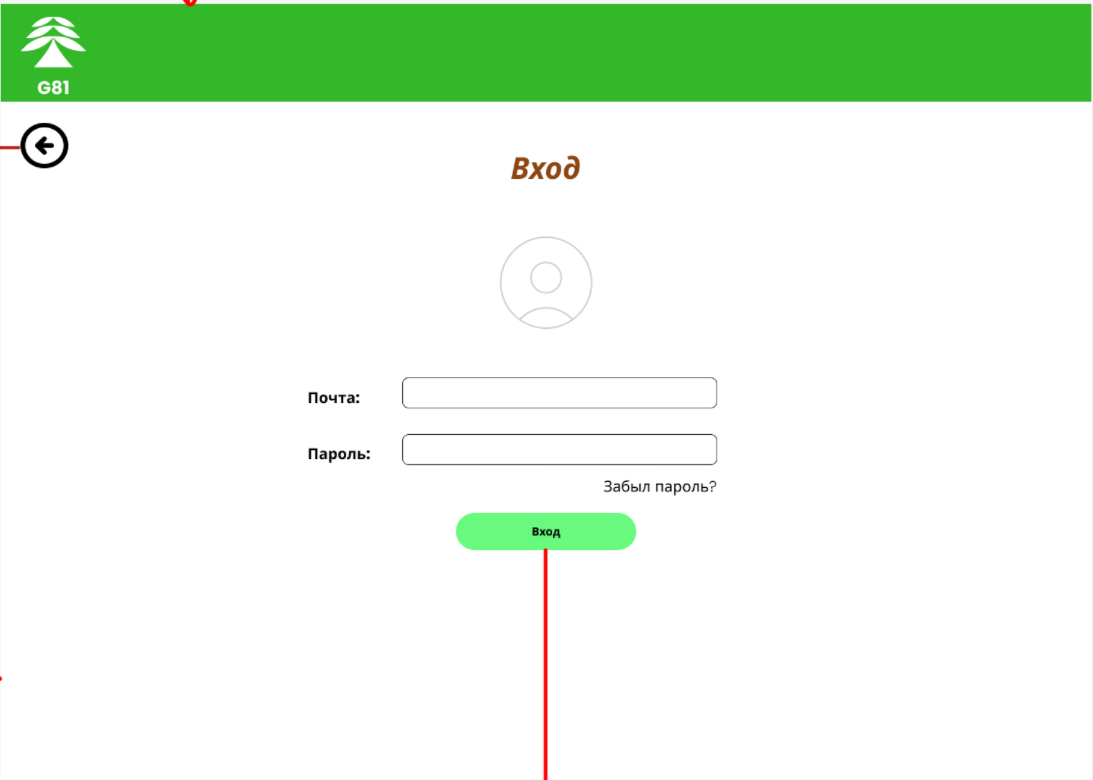
****

Рис. 6. Вход

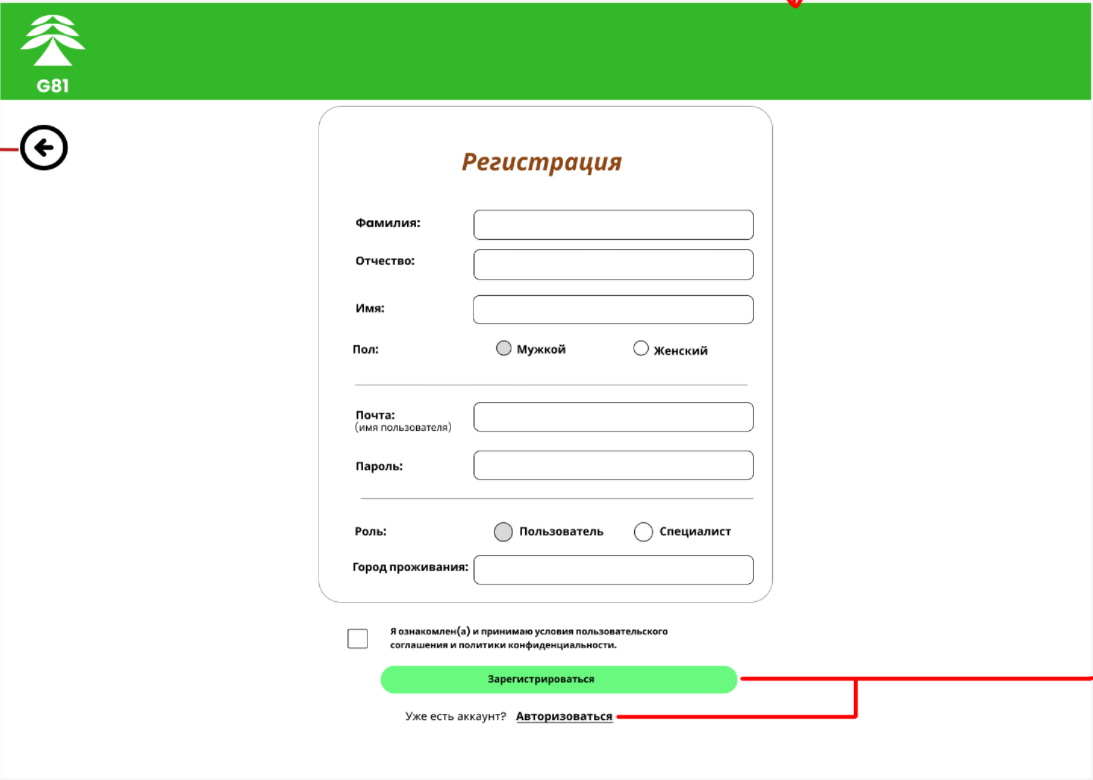
****

Рис. 7. Регистрация

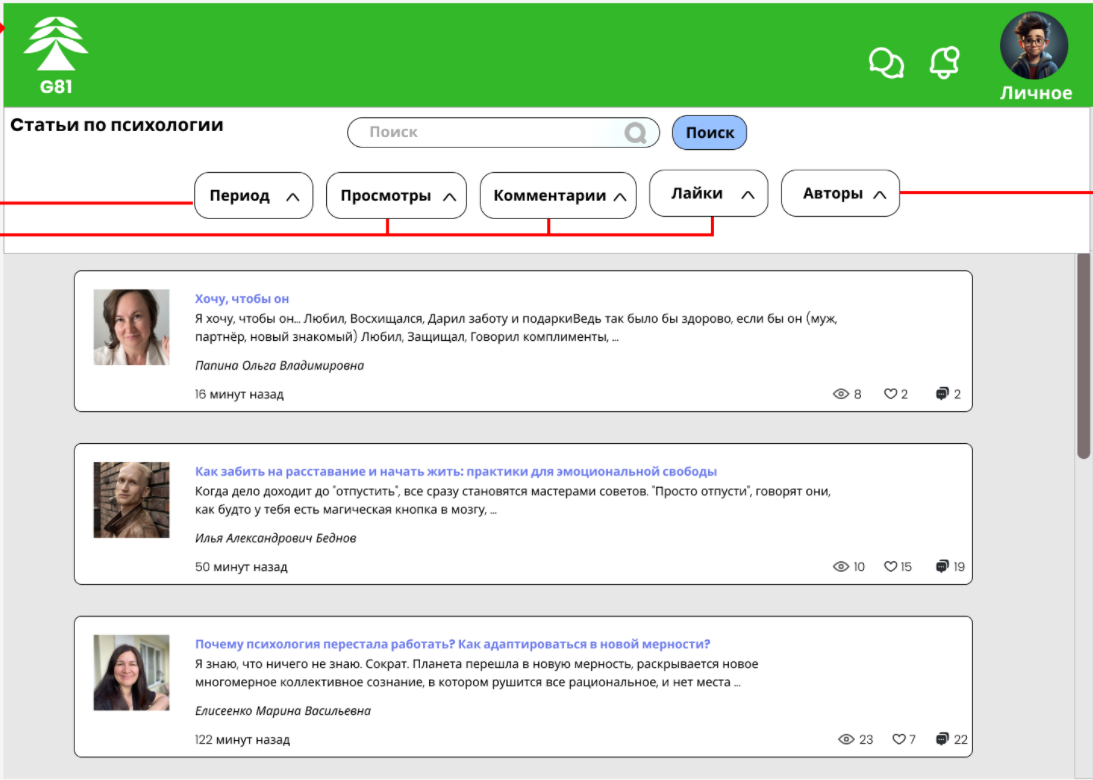


Рис. 8. Статьи по психологии

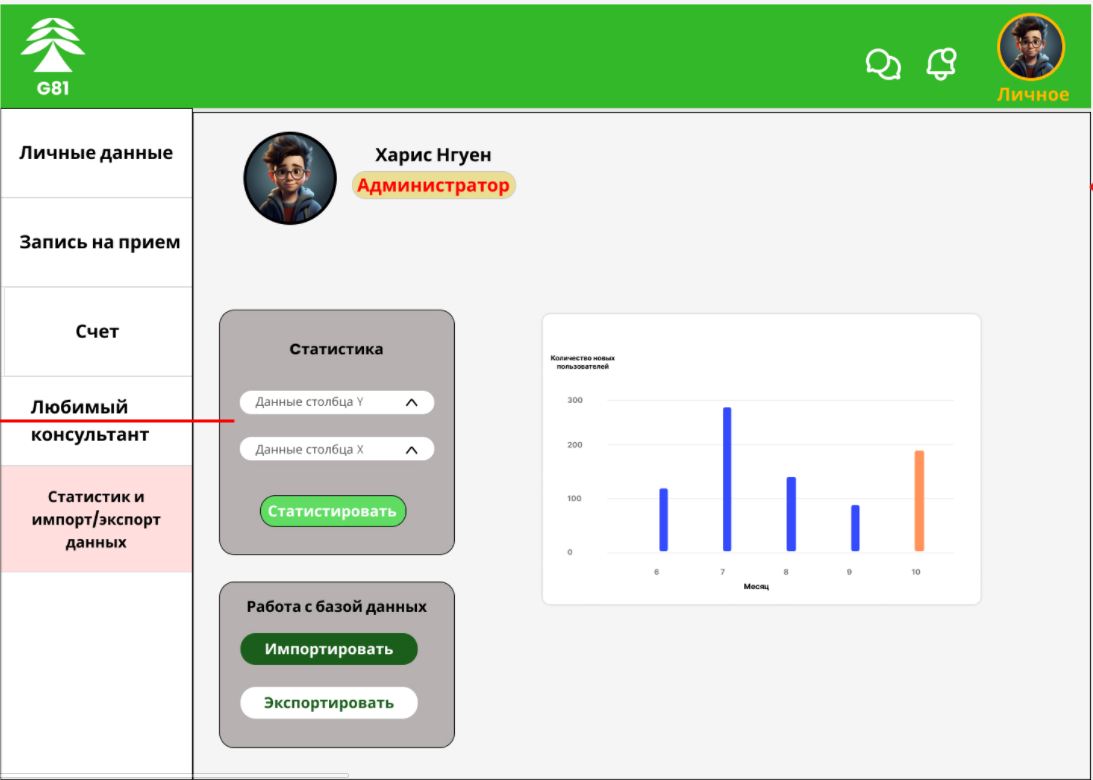


Рис. 9. Статистика, экспорт, импорт

**2.2. Сценарий использования для импорта данных**

**а) Сценарий использования: Импорт данных**

**Цель:**

Обновить базу данных.

**Действующее лицо:**

Администратор

**Основной сценарий:**

Обновленные данные в базе данных преобразуются в новые строки в таблице,

нажав кнопку "импортировать".

**Результат:**

База данных обновлена.

**б) Сценарий использования: Личные данные**

**Цель:**

Пользователь добавляет информацию о себе.

**Действующее лицо:**

Пользователь

**Основной сценарий:**

Пользователь может изменить информацию о себе.

Психолог нажимает на переключатель справа от интерфейса, чтобы опубликовать свою информацию на рынке

**Результат:**

Редактирование информации и опубликовать информацию о психологе на рынке.

**2.3. Сценарий использования для представления данных**

**1. Сценарий использования: Просмотр списка психологов**

**Цель:**

Пользователь просматривает каталог упражнений и применяет фильтры для поиска.

**Действующее лицо:**

Пользователь

**Основной сценарий:**

1. Введите имя психолога в строку поиска, затем нажмите кнопку "Поиск", чтобы найти.
2. Отметьте подходящие критерии, затем нажмите кнопку "Показать психологов", чтобы найти.

**Результат:**

Пользователь находит необходимых психологов.

**2. Сценарий использования: Баланс счета**

**Цель:**

Пользователь просматривает детальную информацию о балансе.

**Действующее лицо:**

Пользователь

**Основной сценарий:**

1. Баланс счета вместе с историей отображается
2. Выберите фильтр или введите ключевое слово, затем нажмите кнопку «Поиск» для поиска подходящих операций

**Результат:**

Управление балансом счета и поиск подходящих операций.

**3. Сценарий использования: Статьи по психологии**

**Цель:**

Пользователь просматривает статьи по психологии.

**Действующее лицо:**

Пользователь

**Основной сценарий:**

1. Нажмите на статью, которая вас интересует, чтобы отобразить детали
2. Выберите фильтр или введите ключевое слово, затем нажмите кнопку «Поиск» для поиска подходящих статьей

**Результат:**

Пользователь может изучать информацию через функцию поиска подходящих статьей.

**2.4. Сценарий использования для анализа данных**

**Сценарий использования: Просмотр статистики**

**Цель:**

Посмотреть статистику данных в системе.

**Действующее лицо:**

Администратор

**Основной сценарий:**

Выберите необходимые данные оси X и Y, затем нажмите кнопку "Статистировать", чтобы отобразить статистику

**Результат:**

Пользователь смотрит статистику статистику по таблице.

**2.5. Сценарий использования для экспорта данных**

**Сценарий использования: Экспорт данных**

**Цель:**

Экспортировать данные в BSON файл.

**Действующее лицо:**

Администратор

**Основной сценарий:**

Экспортировать данные пользователей, нажав кнопку "Экспортировать".

**Результат:**

Экспорт данных.

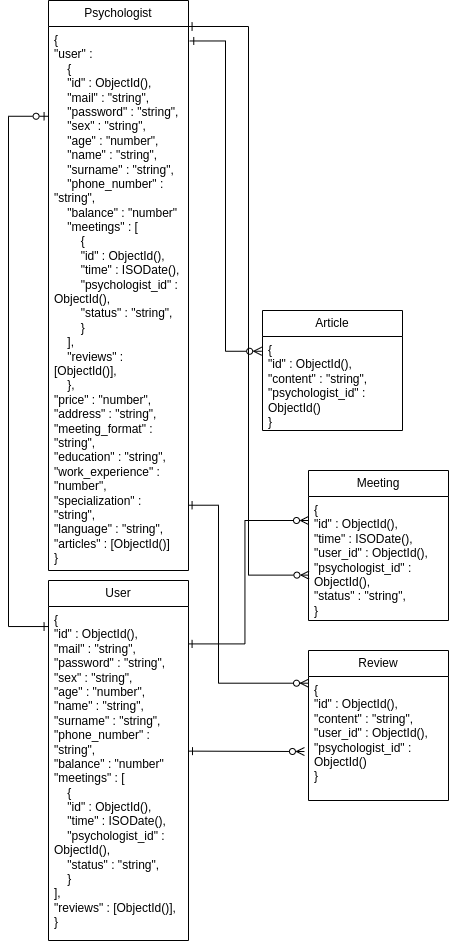
**2.6. Вывод**

Наше решение подразумевает примерно одинаковый объем операций записи и чтения. Пользователи не только активно просматривают данные, но и часто добавляют и редактируют информацию в системе.

**3. МОДЕЛЬ ДАННЫХ**

**3.1. Нереляционная модель данных**

**Графическое представление модели**



**Коллекции и сущности**

Пусть n = 200 - максимальная длина строки.

**User**

Назначение: хранение информации о пользователе.

{

"id" : ObjectId(),

"mail" : "string",

"password" : "string",

"sex" : "string",

"birth\_date": ISODate(),

"name" : "string",

"surname" : "string",

"phone\_number" : "string",

"meetings" : [

{

"id" : ObjectId(),

"time" : ISODate(),

"psychologist\_id" : ObjectId(),

"status" : "string",

}

],

"reviews" : [ObjectId()],

"transactions" : [ObjectId()],

}

**Psychologist**

Назначение: хранение информации о психологе.

{

"user" :

{

"id" : ObjectId(),

"mail" : "string",

"password" : "string",

"sex" : "string",

"birth\_date": ISODate(),

"name" : "string",

"surname" : "string",

"phone\_number" : "string",

"balance" : "number"

"meetings" : [

{

"id" : ObjectId(),

"time" : ISODate(),

"psychologist\_id" : ObjectId(),

"status" : "string",

}

],

"reviews" : [ObjectId()],

},

"price" : "number",

"address" : "string",

"meeting\_format" : "string",

"education": [{ "institution": "string", "degree": "string", "year\_graduated": ISODate() }],

"work\_experience": [{ "position": "string", "place": "string", "years": [ISODate()] }],

"language" : "string",

"articles" : [ObjectId()]

}

**Meeting**

Назначение: хранение информации о сессии клиента с психологом.

{

"id" : ObjectId(),

"time" : ISODate(),

"user\_id" : ObjectId(),

"psychologist\_id" : ObjectId(),

"status" : "scheduled" | "completed" | "canceled",

}

**Review**

Назначение: хранение информации об отзыве о психологе.

{

"id" : ObjectId(),

"content" : "string",

"user\_id" : ObjectId(),

"psychologist\_id" : ObjectId(),

"created\_at": ISODate(),

"rating": "number"

}

**Article**

Назначение: хранение информации о статьях.

{

"id" : ObjectId(),

"content" : "string",

"psychologist\_id" : ObjectId(),

"created\_at": ISODate(),

"updated\_at": ISODate()

}

**Transactions**

Назначение: хранение информации о транзакциях пользователя.

{

"id": ObjectId(),

"user\_id": ObjectId(),

"amount": "number",

"created\_at": ISODate()

}

**Оценка объема информации, хранимой в модели**

Пусть пользователь будет создавать 1 отзыв, 1 встречу, а психолог будет писать 1 статью. И предположим, что на 1 психолога приходится 10 обычных пользователей.

User: 12 + n + 20 + 1 + 8 + 25 + 25 + 11 + 8 + 54 = 356

Psychologist: 12 + n + n + 7 + n + 8 + n + n + 356 + 12 = 1399

Meeting: 12 + 12 + 12 + 8 + 10 = 54

Review: 12 + 4 \* n + 12 + 12 = 836

Article: 12 + 100 \* n + 12 = 20024

Тогда при количестве пользователей равным N удельный объем:

V(N)=(356+836+54+(1399+20024)/10)∗N=3389∗N

**Примеры запросов для совершения сценариев использования**

**1. Поиск психолога по имени:**

db.users.find({

role: "psychologist",

name: { $regex: "имя\_психолога", $options: "i" }

})

**2. Получение статьи по ID:**

db.articles.findOne({ \_id: ObjectId("id\_статьи") })

**3. Получение информации о пользователе:**

db.users.findOne({ \_id: ObjectId("id\_пользователя") })

**4. Отображение зарегистрированных встреч для конкретного пользователя:**

db.meetings.find({ user\_id: ObjectId("id\_пользователя") })

**5. Изменение пользователем информации о себе:**

db.users.updateOne(

{ \_id: ObjectId("id\_пользователя") },

{ $set: { mail: "новый\_email", phone\_number: "новый\_номер\_телефона", name: "новое\_имя" } }

)

**6. Поиск клиентов, у которых больше всего отмененных сессий:**

db.meetings.aggregate([

{ $match: { status: "canceled" } },

{ $group: { \_id: "$user\_id", canceled\_sessions: { $sum: 1 } } },

{ $sort: { canceled\_sessions: -1 } },

{ $limit: 10 }

]);

**7. Поиск самых загруженных:**

db.meetings.aggregate([

{ $match: { time: { $gte: new ISODate(new Date().setMonth(new Date().getMonth() - 1)) } } },

{ $group: { \_id: "$psychologist\_id", sessions\_count: { $sum: 1 } } },

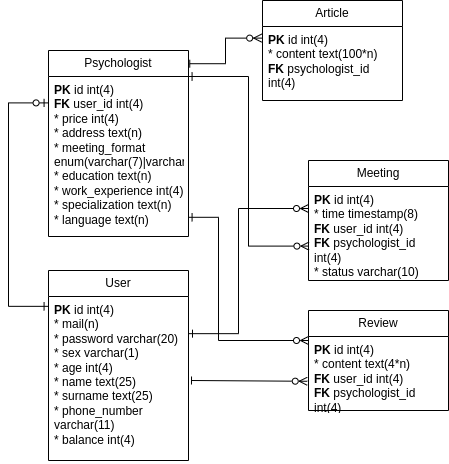
{ $sort: { sessions\_count: -1 } },

{ $limit: 10 }

]);

**3.2. Аналог модели данных для SQL СУБД**

**Графическое представление модели**



**Коллекции и сущности**

Пусть n = 200 - максимальная длина строки.

**User**

Назначение: хранение информации о пользователе.

* id int(4)
* mail(n)
* password varchar(20)
* sex varchar(1)
* birth\_date DATE
* name text(25)
* surname text(25)
* phone\_number varchar(11)
* balance int(4)
* messengers JSONB

**Psychologist**

Назначение: хранение информации о психологе.

user\_id int(4)

* price int(4)
* address text(n)
* meeting\_format enum(varchar(7)|varchar(6))
* education JSONB
* work\_experience JSONB
* specialization text(n)
* language text(n)

**Meeting**

Назначение: хранение информации о сессии клиента с психологом.

* time timestamp(8)
* user\_id int(4)
* psychologist\_id int(4)
* status ENUM ('scheduled', 'completed', 'canceled')

**Review**

Назначение: хранение информации об отзыве о психологе.

* id int(4)
* content text(4\*n)
* user\_id int(4)
* psychologist\_id int(4)
* created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP
* rating INT

**Article**

Назначение: хранение информации о статьях.

* id int(4)
* content text(100\*n)
* psychologist\_id int(4)
* created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP
* updated\_at TIMESTAMP

**Transactions**

Назначение: хранение информации о транзакциях пользователя.

* id int(4)
* user\_id int(4)
* amount int
* created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP

**Оценка объема информации, хранимой в модели**

User: 4 + n + 20 + 1 + 4 + 25 + 25 + 11 + 4 = 294

Psychologist: 4 + n + n + 7 + n + 4 + n + n = 1019

Meeting: 8 + 4 + 4 + 10 = 26

Review: 4 + 4 \* n + 4 + 4 = 812

Article: 4 + 100 \* n + 4 = 20008

Пусть пользователь будет создавать 1 отзыв, 1 встречу, а психолог будет писать 1 статью. И предположим, что на 1 психолога приходится 10 обычных пользователей. Тогда при количестве пользователей равным N удельный объем:

V(N)=(294+26+812+(1019+20008+294)/10)∗N=3276∗N

**Примеры запросов**

**1. Поиск психолога по имени:**

SELECT \*

FROM users

WHERE name ILIKE 'имя\_психолога'

AND id IN (SELECT user\_id FROM psychologists);

**2. Получение статьи по ID:**

SELECT \*

FROM articles

WHERE id = id\_статьи;

**3. Получение информации о пользователе:**

SELECT \*

FROM users

WHERE id = id\_пользователя;

**4. Отображение зарегистрированных встреч для конкретного пользователя:**

SELECT \*

FROM meetings

WHERE user\_id = id\_пользователя;

**5. Изменение пользователем информации о себе:**

UPDATE users

SET mail = 'новый\_email', phone\_number = 'новый\_номер\_телефона', name = 'новое\_имя'

WHERE id = id\_пользователя;

**6. Поиск клиентов, у которых больше всего отмененных сессий:**

SELECT user\_id, COUNT(\*) AS canceled\_sessions

FROM meetings

WHERE status = 'canceled'

GROUP BY user\_id

ORDER BY canceled\_sessions DESC

LIMIT 10;

**7. Поиск самых загруженных:**

SELECT psychologist\_id, COUNT(\*) AS sessions\_count

FROM meetings

WHERE time >= (CURRENT\_DATE - INTERVAL '1 month')

GROUP BY psychologist\_id

ORDER BY sessions\_count DESC

LIMIT 10;

**3.3. Сравнение моделей**

**Удельный объем информации**

Для сохранения объектов в реляционной модели требуется меньше памяти из-за отсутствия дублирования данных.

В MongoDB наблюдается некоторая избыточность данных, так как сущности, например, User и Psychologist, дублируют информацию друг друга. Это увеличивает объем хранимых данных, особенно когда психологи дублируют данные пользователей.

**Запросы по отдельным юзкейсам**

Количество запросов для каждой модели одинаково и равно 1.

**Количество задействованных коллекций**

Обе модели используют одну коллекцию при любом сценарии использования, кроме поиска психолога по именени в реляционной модели, так используются две коллекции для поиска – users и psychologists.

**Вывод**

Для данного проекта наиболее подходящей будет MongoDB (NoSQL). Она обеспечивает гибкость в структуре данных, высокую производительность при работе с вложенными документами и легкость в масштабировании. Это особенно важно в условиях частых изменений в структуре данных и необходимости обрабатывать большие объемы информации, связанных с психологами, пользователями, встречами и отзывами.

**4. РАЗРАБОТАННОЕ ПРИЛОЖЕНИЕ**

**4.1 Краткое описание**

Весь код приложения разделен на две части: back-end и front-end.

Back-end реализован на языке Python и служит элементом, обеспечивающим передачу данных из БД на фронтенд для дальнейшего отображения.

Front-end обращается к API back-end части приложения и отображает данные в удобном для пользователя виде.

**4.2 Использованные технологии**

БД: MongoDB.

Back-end: Python.

Front-end: TypeScript, Vue3.

**4.3 Снимки экранов приложения**

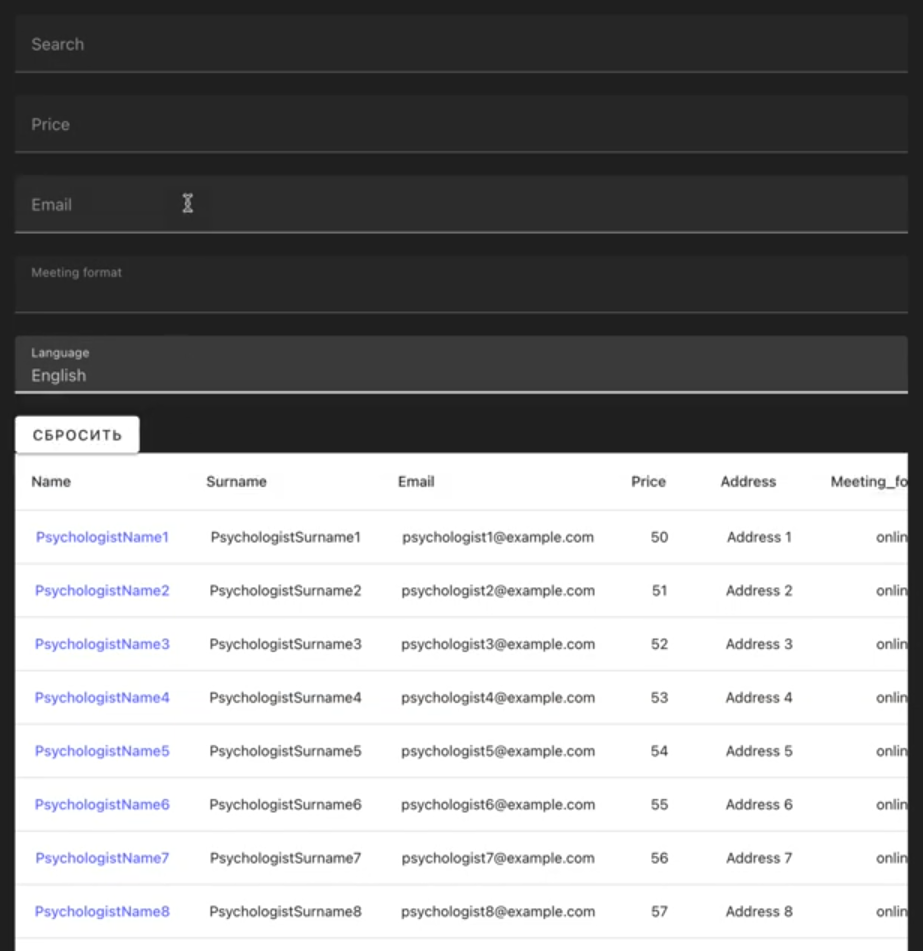
****

Рис. 10. Страница поиска психологов

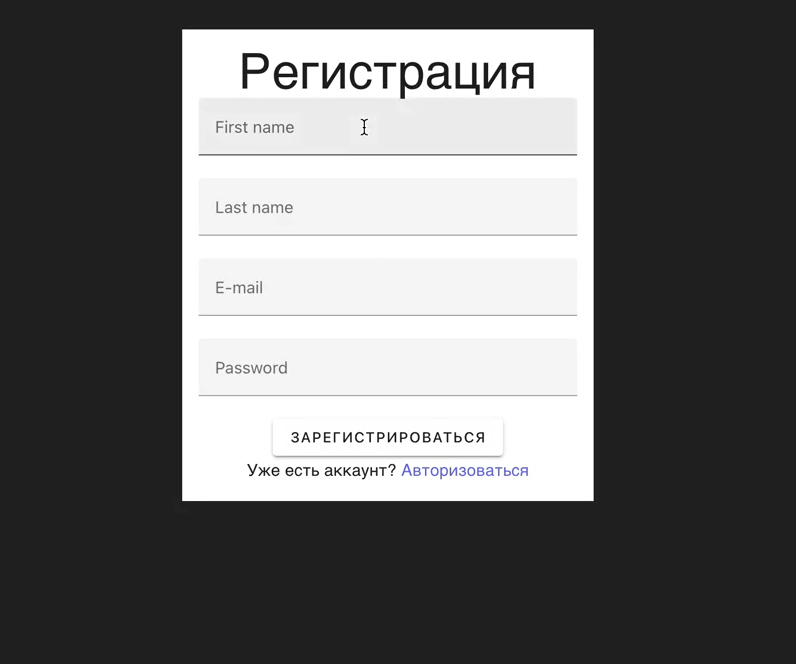
****

Рис. 11. Страница регистрации

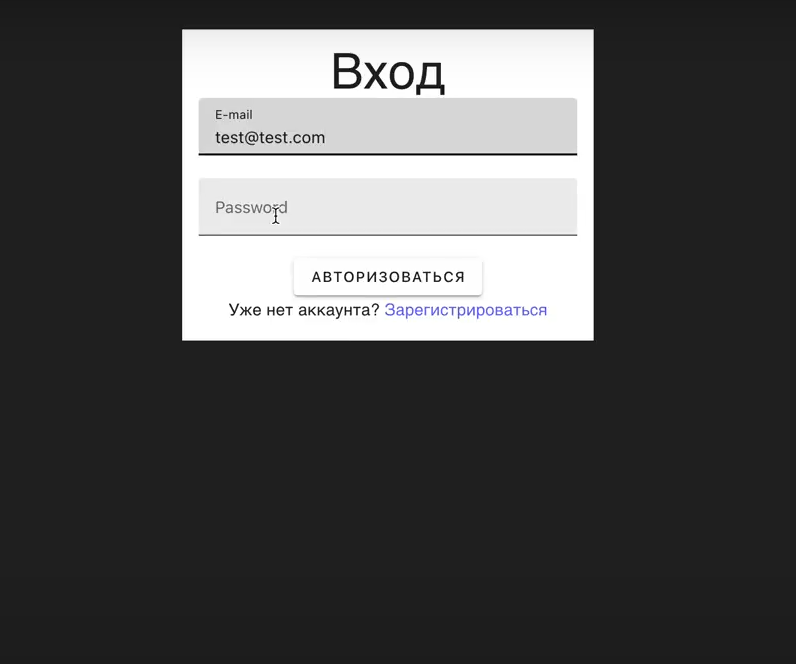
****

Рис. 12. Вход

**5. ВЫВОДЫ**

**5.1 Достигнутые результаты**

В ходе работы было разработано приложение, представляющее собой сервис поиска психологов. Приложение включает функционал для просмотра страниц психологов. Реализована система фильтрации и поиска психологов по различным критериям.

Для удобства работы с данными доступны функции массового импорта и экспорта данных.

**5.2 Недостатки и пути для улучшения**

На данный момент приложение не предоставляет возможность оставлять отзывы, публиковать и просматривать статьи.

Не реализован функционал обмена сообщениями.

Приложение доступно только на русском языке, что ограничивает аудиторию пользователей. Внедрение поддержки нескольких языков позволит расширить аудиторию.

На данный момент приложение доступно только в веб-формате, что ограничивает его использование на мобильных устройствах.

Также необходимо улучшить адаптивность текущего веб-приложения, чтобы оно корректно отображалось на устройствах с разными размерами экранов.

**5.3 Будущее развитие решения**

В будущем планируется реализация недостающих приложению функций, а также улучшение адаптивности веб-приложения сделают его доступным и удобным для пользователей на разных устройствах.

**6. ПРИЛОЖЕНИЯ**

**6.1 Документация по сборке и развертыванию приложения.**

1. Склонировать репозиторий с проектом и перейти в директорию проекта.
2. Собрать контейнеры приложения командой:

*docker-compose build --no-cache*

1. Запустить контейнеры командой:

*docker-compose up*

1. Открыть приложение в браузере по адресу 127.0.0.1:3000 или нажав на порт контейнера frontend в приложении Docker Desktop.

**6.2 Инструкция для пользователя.**

1. **Регистрация.**

Необходимо перейти на страницу регистрации пользователя, после чего заполнить все требуемые поля и нажать на кнопку “Зарегистрироваться”. Если пользователь уже имеет акканут, нужно перейти на страницу для входа в аккаунт.

1. **Поиск психолога.** Для того, чтобы найти психолога, необходимо перейти на страницу отображения психологов, заполнить поля для фильтрации. При изменении фильтров список психологов обновляется. Чтобы сбросить фильтры, нужно нажать на кнопку “сбросить”.
2. **Экспорт/импорт.**

Необходимо перейти на страницу для эспорта/импорта. Для экспорта данные пользователей нажать кнопку "экспортировать". Скачается файл в формате json. Для импорта нажать кнопку "импортировать" и загрузить файл в формате json, соответствующий схеме данных.

**7. ЛИТЕРАТУРА**

1. Ссылка на GitHub. - [Электронный ресурс]. - URL: https://github.com/moevm/nosql2h24-phs.
2. Vue.js. – [Электронный ресурс]. – URL: https://vuejs.org/ (дата обращения: 21.12.2024).
3. MongoDB. – [Электронный ресурс]. – URL: https://www.mongodb.com/ (дата обращения: 21.12.2024).
4. Python. – [Электронный ресурс]. – URL: https://docs.python.org/3/ (дата обращения: 21.12.2024).