**МИНОБРНАУКИ РОССИИ**
**САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ**
**ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**
**«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)**
**Кафедра МО ЭВМ**

# ИНДИВИДУАЛЬНОЕ ДОМАШНЕЕ ЗАДАНИЕ

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

**Тема: Приложение для мониторинга пациентов**

Студент гр. 9382 _____ Дерюгин Д.А.

Студент гр. 9383 _____ Вербин К.М.

Студент гр. 9383 _____ Гордон Д.А.

Преподаватель _____ Заславский М.М.

Санкт-Петербург
2023

**ЗАДАНИЕ**

Студенты

Дерюгин Д.А.
Вербин К.М.
Гордон Д.А.
Группа 9382, 9382

Тема работы: Приложение для мониторинга пациентов.

Исходные данные:

Необходимо реализовать приложение для СУБД(MongoDB).

Содержание пояснительной записки:

«Содержание»

«Введение»

«Качественные требования к решению»

«Сценарий использования»

«Модель данных»

«Разработка приложения»

«Вывод»

Предполагаемый объем пояснительной записки:

Не менее 20 страниц.

Дата выдачи задания:

Дата сдачи реферата:

Дата защиты реферата:

| | | |
|---|---|---|
| Студент гр. 9382 | | Дерюгин Д.А. |
| Студент гр. 9383 | | Вербин К.М. |
| Студент гр. 9383 | | Гордон Д.А. |
| Преподаватель | | Заславский М.М. |

## АННОТАЦИЯ

В данной работе было разработано приложение для мониторинга пациентов. Для разработки использовались MongoDB в качестве базы данных, Nodejs для серверной части React для клиентской части приложения.

## SUMMARY

In this work, an application for patient monitoring was developed. MongoDB was used as the database, Nodejs for the server-side, and React for the client-side of the application.

**СОДЕРЖАНИЕ**

## 1. Введение

Цель работы - создать веб-приложение для предоставления докторам возможности наблюдать за состоянием пациентов.

## 2. Качественные требования к решению

Требуется разработать веб-приложение с использованием СУБД MongoDB.

## 3. Сценарии использования

**Макет UI:**

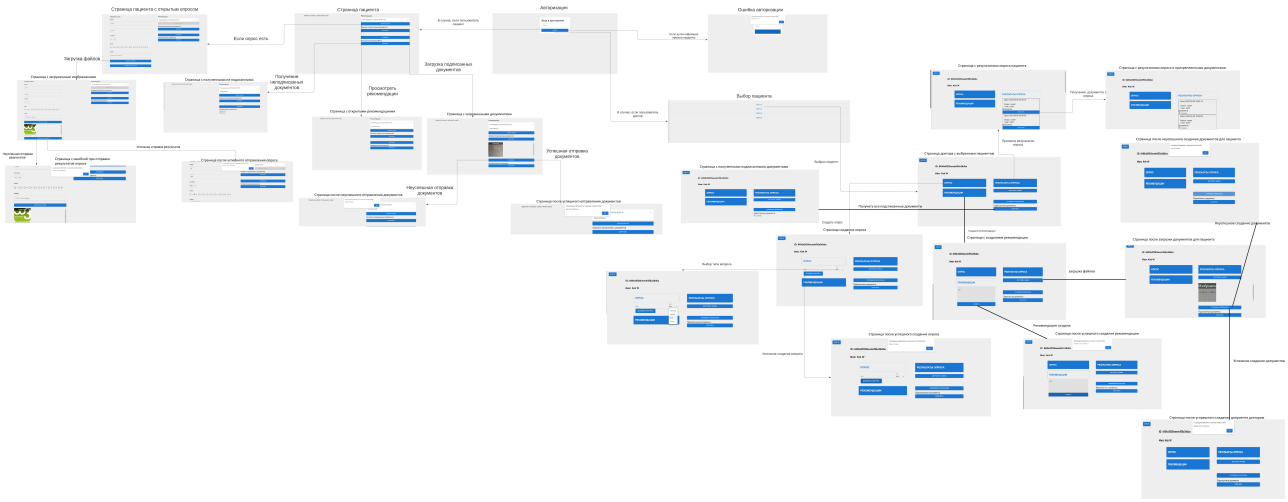


Рисунок 1 –Use Case

**Описание сценариев использования:**

**Сценарий использования – "Регистрация"**

Действующее лицо: Пользователь.

Начальные условия: Пользователь зашёл на сайт приложения (экран «Авторизация»).

Основной сценарий:

Пользователь нажимает на кнопку "Регистрация" (экран 1)

Пользователь вводит данные для входа, нажимает кнопку "Войти" (экран «Авторизация»)

Если данные введены корректно и пользователь - Пациент, то пользователь входит на сайт (экран «Страница пациента»)

Если данные введены корректно и пользователь - Доктор, то пользователь входит на сайт (экран «Выбор пациента»)

Альтернативный сценарий:

В случае некорректно введённых данных выдается ошибка (экран «Ошибка авторизации»)

**Сценарий использования – "Пройти опрос"**

Действующее лицо: Пациент.

Начальные условия: Пользователь авторизован на сайт приложения как пациент (экран «Страница пациента»).

Основной сценарий:

Пациент нажимает на кнопку «пройти опрос» (экран «Страница пациента»)

Открывается доступный опрос (экран «Страница пациента с открытым опросом»)

Пациент отвечает на все доступные вопросы(экран «Страница пациента с открытым опросом»)

Пациент нажимает на кнопку «загрузить файлы»

Пациент загружает файлы (экран «Страница с загруженным изображением»)

Пациент нажимает на кнопку «отправить результаты»(экран «Страница с загруженным изображением»)

Происходит успешная отправка результатов(экран «Страница после успешного отправления опроса»)

Альтернативный сценарий:

Пациент нажимает на кнопку «пройти опрос» (экран «Страница пациента»)

Открывается доступный опрос (экран «Страница пациента с открытым опросом»)

Пациент отвечает не на все доступные вопросы(экран «Страница пациента с открытым опросом»)

Пациент нажимает на кнопку «отправить результаты»(экран «Страница с загруженным изображением»)

Происходит ошибка отправка результатов(экран «Страница с ошибкой при отправке результата»)

**Сценарий использования – "Просмотр рекомендаций врача"**

Действующее лицо: Пациент.

Начальные условия: Пользователь авторизован на сайт приложения как пациент (экран «Страница пациента»). Рекомендации есть.

Основной сценарий:

Пациент нажимает на рекомендацию(экран «Страница пациента»)

Появляется текст рекомендации(экран «Страница с открытыми рекомендациями»)

**Сценарий использования – "Загрузка подписанных документов"**

Действующее лицо: Пациент.

Начальные условия: Пользователь авторизован на сайт приложения как пациент (экран «Страница пациента»). Рекомендации есть.

Основной сценарий:

Пациент нажимает кнопку "загрузить" (экран «страница пациента»)

Пациент загружает подписанные документы (экран «страница с загруженными документами»)

Пациент нажимает на кнопку «отправить»(экран «страница с загруженными документами»)

Пациент переходит на страницу с успешным отправлением документов(экран «страница после успешного отправления документов»

Альтернативный сценарий:

Пациент нажимает кнопку "загрузить" (экран «страница пациента»)

Пациент не загружает подписанные документы (экран «страница с загруженными документами»)

Пациент нажимает на кнопку «отправить»(экран «страница с загруженными документами»)

Пациент переходит на страницу с ошибкой отправления документов (экран «страница после неуспешного отправления документов»)

**Сценарий использования – "Получение неподписанных документов"**

Действующее лицо: Пациент.

Начальные условия:  Пользователь авторизован на сайт приложения как пациент (экран «Страница пациента»).

Основной сценарий:

Пациент нажимает кнопку "получить " (экран экран «Страница пациента»)

Пациент нажимает на кнопку «скачать» (экран «страница с полученными неподписанными документами»)

**Сценарий использования – "Выбор пациента"**

Действующее лицо: Доктор.

Начальные условия: Пользователь авторизован на сайт приложения как пациент (экран «Выбор пациента»).

Основной сценарий:

Доктор выбирает из списка пациента (экран «Выбор пациента»)

Доктор переходит к странице пациента(экран «Страница доктора с выбранным пациентом»)

**Сценарий использования – "Создания опроса"**

Действующее лицо: Доктор.

Начальные условия: Пользователь авторизован на сайт приложения как пациент (экран «Выбор пациента»).

Основной сценарий:

Доктор нажимает кнопку "опрос" (экран «страница доктора с выбранным пациентом»)

Открывается страница с созданием опроса (экран «страница создания опроса»)

Доктор нажимает на поле выбора типа вопроса(экран «страница создания опроса»)

Появляется выпадающее меню с типами вопросов(экран «выбор типа вопроса»)

Доктор пишет опрос  (экран «страница создания опроса»)

Доктор нажимает на кнопку «добавить вопрос» (экран «страница создания опроса»)

Доктор переходит на страницу с успешным добавлением вопроса (экран «страница после успешного создания опроса»)

**Сценарий использования – "Создание рекомендации"**

Действующее лицо: Доктор.

Начальные условия: Пользователь авторизован на сайт приложения как пациент (экран «Выбор пациента»).

Основной сценарий:

Доктор нажимает кнопку "рекомендации" (экран «страница доктора с выбранным пациентом»)

Открывается страница с созданием рекомендации(экран «страница с созданием рекомендации»)

Доктор пишет рекомендацию(экран «страница с созданием рекомендации»)

Доктор нажимает на кнопку «создать» (экран «страница с созданием рекомендации»)

Доктор переходит на страницу с успешным созданием рекомендации(экран «страница после успешного создания рекомендации»)

**Сценарий использования – "Просмотр ответов на опрос"**

Действующее лицо: Доктор.

Начальные условия: Пользователь авторизован на сайт приложения как пациент (экран «Выбор пациента»).

Основной сценарий:

Доктор нажимает кнопку "результаты опроса" (экран «страница доктора с выбранным пациентом»)

Открывается страница с результатами опросов(экран «страница с результатами опроса пациента»)

Доктор нажимает на кнопку «получить»(экран «страница с результатами опроса пациента»)

Доктор получает возможность сказать прикрепленные к опросу файлы(экран «Страница с результатами опроса и прикрепленными документами»)

**Сценарий использования – "Загрузка документов для подписания"**

Действующее лицо: Доктор.

Начальные условия: Пользователь авторизован на сайт приложения как пациент (экран «Выбор пациента»).

Основной сценарий:

Доктор нажимает кнопку "загрузить файлы" (экран «страница доктора с выбранным пациентом»)

Доктор загружает документы на подписание(экран «страница после загрузки документов для пациента»)

Доктор нажимает на кнопку «отправить результаты»(экран «страница после загрузки документов для пациента»)

Доктор переходит на страницу с успешным созданием документов(экран «страница после успешного создания документа доктором»

Альтернативный сценарий:

Доктор нажимает кнопку "загрузить файлы" (экран «страница доктора с выбранным пациентом»)

Доктор не загружает документы на подписание (экран «страница доктора с выбранным пациентом»)

Доктор нажимает на кнопку «отправить результаты» (экран «страница доктора с выбранным пациентом»)

Доктор переходит на страницу с ошибкой отправления документов(экран «страница после неуспешного создания документов для пациента»

**Сценарий использования – "Загрузка документов для подписания"**

Действующее лицо: Доктор.

Начальные условия: Пользователь авторизован на сайт приложения как пациент (экран «Выбор пациента»).

Основной сценарий:

Доктор нажимает кнопку "загрузить файлы" (экран «страница доктора с выбранным пациентом»)

Доктор загружает документы на подписание(экран «страница после загрузки документов для пациента»)

Доктор нажимает на кнопку «отправить результаты»(экран «страница после загрузки документов для пациента»)

Доктор переходит на страницу с успешным созданием документов(экран «страница после успешного создания документа доктором»

Альтернативный сценарий:

Доктор нажимает кнопку "загрузить файлы" (экран «страница доктора с выбранным пациентом»)

Доктор не загружает документы на подписание (экран «страница доктора с выбранным пациентом»)

Доктор нажимает на кнопку «отправить результаты» (экран «страница доктора с выбранным пациентом»)

Доктор переходит на страницу с ошибкой отправления документов(экран «страница после неуспешного создания документов для пациента»

**Сценарий использования – "Получение подписанных документов"**

Действующее лицо: Доктор.

Начальные условия: Пользователь авторизован на сайт приложения как пациент (экран «Выбор пациента»).

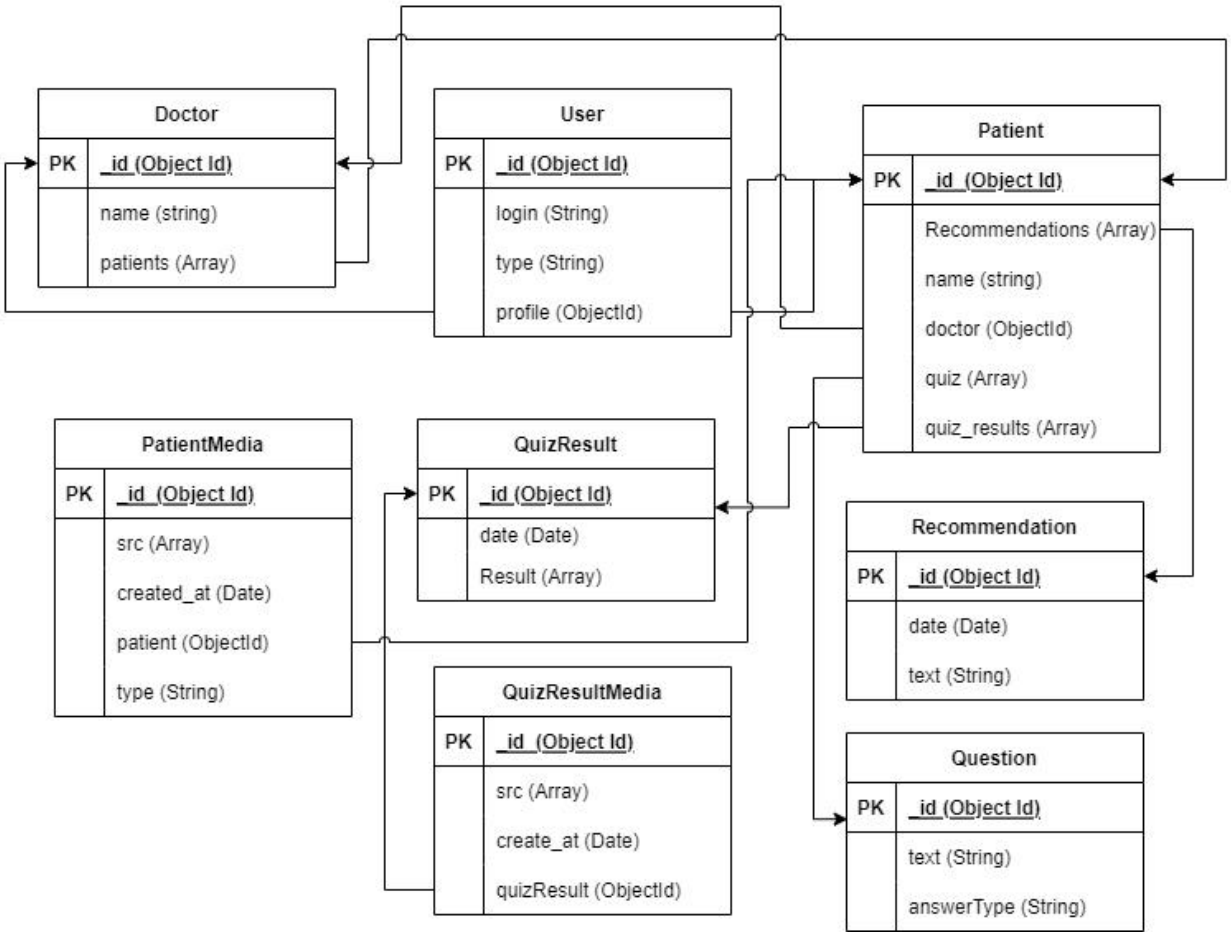
Основной сценарий:

Доктор нажимает кнопку "получить" (экран «страница доктора с выбранным пациентом»)

Доктор нажимает на кнопку «скачать» (экран «страница с полученными подписанными документами»)

## 4. Модель данных

A. Графическое представление




B. Описание коллекций

| **User** | | |
|---|---|---|
| **Наименование** | **Тип** | **Размеры (байты)** |
| login | String | [3; 15] |
| role | String | 6 или 7 |
| profile | ObjectId | 12 |

Хранит логин, роль и ссылку на доктора/пациента.
Память: 15+7+12 = 34 байт

| **Doctor** | | |
|---|---|---|
| **Наименование** | **Тип** | **Размеры** |
| name | String | 20 |
| patients | Array of ObjectIds | N*12 |

Хранит информацию о имени доктора и его пациентах.

Память: 20+N*12 байт

| Patient | | |
|---|---|---|
| **Наименование** | **Тип** | **Размеры** |
| name | String | 20 |
| doctor | ObjectId | 12 |
| quiz | Array of ObjectIds | N*12 |
| quiz_results | Array of ObjectIds | N*12 |
| recommedations | Array of ObjectIds | N*12 |

Хранит имя, лечащего врача, опрос, результат опроса и рекомендации пациента.
Память: $20 + 12 + N_1 * 12 + N_2 * 12 + N_3 * 12 = 32 + 12(N_1 + N_2 + N_3)$ байт

| PatientMedia | | |
|---|---|---|
| **Наименование** | **Тип** | **Размеры** |
| src | Array of String | N*28 |
| created_at | Date | 29 |
| patient | ObjectId | 12 |
| type | String | 6 или 8 |

Хранит путь к файлам, дату создания, ссылку на пациента и подписаны ли файлы.
Память: $N * 28 + 29 + 12 + 8 = 49 + N * 28$ байт

| Question | | |
|---|---|---|
| **Наименование** | **Тип** | **Размеры** |
| answerType | String | 7 |
| text | String | 20 |

Хранит тип ответов и текст вопроса.
Память: $7 + 20 = 27$ *байт*

| QuizResult | | |
|---|---|---|
| **Наименование** | **Тип** | **Размеры** |
| date | Date | 29 |
| Result | Array of Object ({answer: String, questionId: | N*(25+12) |

| | | |
|---|---|---|
| | ObjectId}) | |

Хранит дату ответа на опрос и результат (ответ и ссылку на вопрос).
Память: $29 + N * 37$ байт

| **QuizResultMedia** | | |
|---|---|---|
| **Наименование** | **Тип** | **Размеры** |
| src | Array of String | N*28 |
| create_at | Date | 29 |
| quizResult | ObjectId | 12 |

Хранит приложенные к опросу файлы (пути, дата создания и ссылку на ответ).
Память: $N * 28 + 29 + 12 = 41 + N * 28$ байт

| **Recommendation** | | |
|---|---|---|
| **Наименование** | **Тип** | **Размеры** |
| date | Date | 29 |
| text | String | 50 |

Хранит информацию о рекомендации (дата создания и текст рекомендации).
Память: $29 + 50 = 79$ *байт*

C. Оценка удельного объема информации, хранимой в модели.

| Коллекция | Верхняя граница по памяти (байты) для одного экземпляра | Зависимость размера |
|---|---|---|
| User | 34 | Линейная зависимость от кол-ва докторов и пациентов |
| Doctor | $20 + N_p * 12$ | Линейно зависит от кол-во пользователей |
| Patient | $32 + 12(N_q + N_{qr} + N_r) = 32 + 12(2N_r + N_q)$ | Линейной зависит от кол-ва вопросов, от рекомендаций, от ответа на опрос |
| PatientMedia | $49 + N_f * 28$ | Линейно зависит от кол-ва файлов |
| QuizResult | $29 + N_a * 37$ | Линейно зависит от кол-ва ответов |
| QuizResultMedia | $41 + N_f * 28$ | Линейно зависит от |

| | | кол-ва фалов |
|---|---|---|
| Question | 27 | Линейно зависит от кол-ва вопросов |
| Recommendation | 79 | Линейно зависит от кол-ва рекомендаций |

Кол-во Users = кол-во Doctors + кол-во Patients.

Пусть будет два пациента и два врача. У каждого по врачу.

Коллекция Users увеличится на 34*4 байт.

Коллекция Doctors на 2*(20+12) байт.

У пациента может быть записан только один доктор. При создании опроса к пациенту добавляются в массив Id вопросов, а после ответа на них – ещё и Id ответов (все ответы на один опрос – один экземпляр QuizResult). На каждый ответ – рекомендация.

Получается, что если в опросе – 12 вопросов и нужно загрузить 2 файла, то коллекция Patients увеличится в объеме на 2(кол-во пациентов)*(32 + 10*12 (Id вопросов) + 12(Id ответа) + 12(Id рекомендации)) байт, QuizResults увеличится на 2*(29 + 10*12(ответы)) байт, Recommendations на 2*79 байт, Question на 2*27*12 байт, QuizResultMedia на 2*(41 + 2*28) байт. Итого:

$2 * ((32 + 10 * 12 + 12 + 12) + (29 + 10 * 12) + 79 + (27 * 12) + (41 + 2 * 28) + (20 + 12) + (34 * 2)) = 1850$ байт.

При отправке каждому пациенту пяти документов на подпись увеличится коллекция PatientMedias на $2 * (49 + 5 * 28) = 378$ байт.

Итого: $378 + 1850 = 2228 \approx 2$ кбайта.

D. Избыточность модели (отношение между фактическим объемом модели и “чистым” объемом данных).

Vf = Фактический объем = 2396 байт.

Vc = Чистый объем = 2228 байт.

Vc/Vf = 0.92988313856

E. Направление роста модели при увеличении количества объектов каждой сущности.

| Коллекция | Увеличение БД, байты | Объяснение |
|---|---|---|
| User | 34 | Зависит от Patient и Doctor |
| Doctor | $20 + N_p * 12 + 34$<br>При $N_p = 1$:<br>$20 + 12 + 34 + 12 = 78$ | Объем БД увеличивается на документ в *Users,* на документ в *Doctors* и на 12 байт (добавление к пациенту) |
| Patient | $32 + 12(2N_{qr,r} + N_q)$<br>$+34$<br>$+ N_q * 27$<br>$+49 + N_f * 28$<br>$+29 + N_a * 37$<br>$+79$<br>$+49 + N_f * 28$<br>$+12 * 4$<br>При $N_{qr,r} = 1$,<br>$N_q = N_a = 5$,<br>$N_f = 5$:<br>937 | Объем БД увеличивается на документ в *Users*, на документ в *Patients*, на $N_q$ документов в *Questions*, на $N_{qr,r}$ документов в *Recommendations* и *QuizResults*, на документ в *QuizResultMedia*, на документ в *PatientMedias*, на 12*4 байт (добавление к доктору в массив patients, в массивы пациента) |
| PatientMedia | $49 + N_f * 28$<br>$N_f = 5$:<br>49+5*28=189 | Объём БД увеличивается на документ в *PatientMedias* |
| QuizResult | $29 + N_a * 37$<br>$+41 + N_f * 28$<br>$+79$<br>$+12 * 2$<br>При $N_a = N_f = 5$: | Объём БД увеличивается на документ в *QuizResults, QuizResultMedias,* |

| | | |
|---|---|---|
| | 29+5*37+41+5*28+12*2+79 =498 | *Recommendations,* на 12*2 байт(добавление в массивы пациента), (подразумеваем, что $N_a$ вопросов уже есть и не учитываем) |
| QuizResultMedia | $41 + N_f * 28$<br>При $N_f = 5$:<br>41+5*28=181 | Объём БД увеличивается на документ в *QuizResultMedias* (добавление в *QuizResultMedias* подразумевает существование связного документа в *QuizResults*, поэтому не учитываем его) |
| Question | $N_q * 27$<br>$+29 + N_a * 37$<br>$+41 + N_f * 28$<br>$+79$<br>$+ (N_q + 2) * 12$<br>$N_a = N_q = N_f = 5$:<br>5*27+29+5*37+41+5*28+79 =609 | Объём БД увеличивается на $N_q$ документов в *Questions*, на документ в *QuizResults, QuizResultMedias, Recommendations*, на $(N_q + 2) * 12$ байтов (добавление в массивы пациента) |
| Recommendation | $79 + 12 = 91$ | Объём БД увеличивается на документ в Recommendations и на 12 байт (ObjectId в массив recommendations пациента) |

V = 2228 байт

| Коллекция | Увеличение БД, % |
|---|---|

| | |
|---|---|
| User | 1.5 |
| Doctor | 3.5 |
| Patient | 42 |
| PatientMedia | 8.4 |
| QuizResult | 22.4 |
| QuizResultMedia | 8.1 |
| Question | 27.3 |
| Recommendation | 4 |

Самые ресурсоемкие – создание опроса (вопросы) и ответов на опрос (много связей с другими сущностями).

F. Запросы к модели, с помощью которых реализуются сценарии использования.

I. Текст запросов, кол-во задействованных коллекций

| Описание | Текст | Кол-во задействованных коллекций |
|---|---|---|
| Получение пользователя по логину | Users.findOne({ login: body.login })<br>Для доктора:<br>Doctors.findById(user.profile)<br>Patients.find({ doctor: found._id }).select({ '_id': 1, 'name': 1 })<br>Для пациента:<br>Patients.findById(user.profile)<br>Если есть назначенный доктор:<br>Doctors.findById(found.doctor).select({ 'name': 1, '_id': 1 })<br>Иначе: Doctors.find({}).select({ 'name': 1, '_id': 1 }) | 3 |
| Получение пользователя | Patients.findById(patientId).populate('quiz').populate ('recommendations').populate('quiz_results') | 1 |
| Получение опроса по Id пациента | Patients.findById(patientId).populate('quiz') | 1 |
| Создание опроса | Получение пациента:<br>Patients.findById(body.patientId)<br>Создание вопросов | 2 |
| Получение ответов(фай | QuizResultMedia.findOne({quizResult: QuizResultId}) | 1 |

| | | |
|---|---|---|
| лы) на опросы | | |
| Создание ответа на опрос | Создание QuizResult, QuizResultMedia<br>Поиск пациента: Patients.findById(JSON.parse(patientId)) | 3 |
| Получение неподписанных файлов | PatientMedia.findOne({ patient: patientId, type: 'Unsigned' }) | 1 |
| Создание неподписанных документов | Создание PatientMedia | 1 |
| Получение подписанных файлов | PatientMedia.findOne({ patient: patientId, type: 'Signed' }) | 1 |
| Создание подписанных документов | Создание PatientMedia | 1 |
| Получение рекомендаций | Patients.findById(patientId).populate('recommendations') | 1 |
| Создание рекомендации | Создание Recommendation<br>Поиск пациента: Patients.findById(patientId) | 2 |
| Поменять доктора | Поиск пациента: Patients.findById(patientId) | 1 |

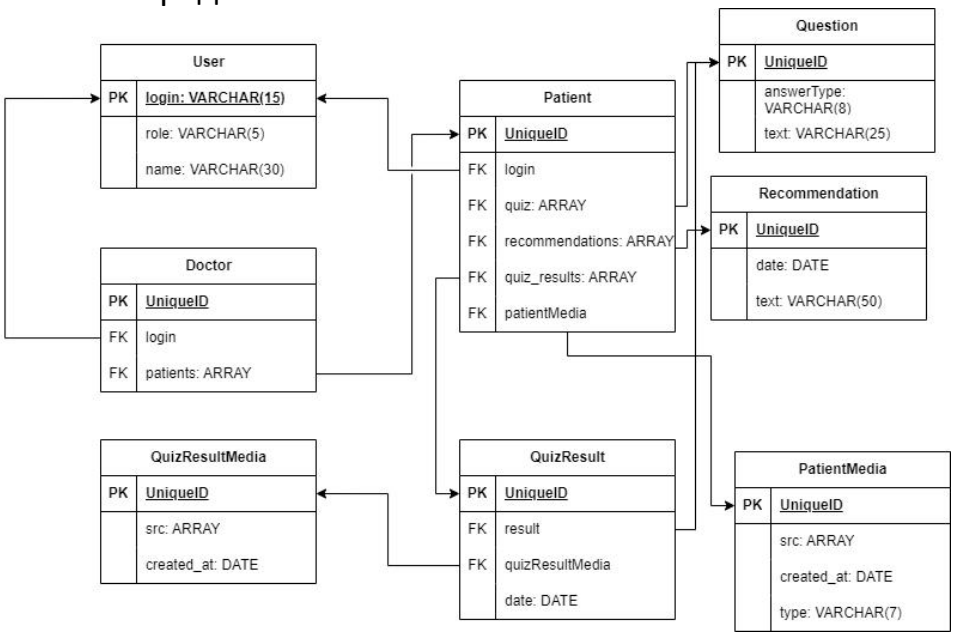
Пример хранения:

```
_id: ObjectId('63e8eb3236ff44f3cdab4af5')
name: "Kek W"
▾ quiz: Array
    0: ObjectId('63ff7e60481a4410d77d33bc')
    1: ObjectId('63ff7e60481a4410d77d33be')
▾ quiz_results: Array
▾ recommendations: Array
    0: ObjectId('63e917d6cb519d4a316d6b42')
    1: ObjectId('63e9185898898c308749405a')
__v: 5
doctor: ObjectId('63e8eb3236ff44f3cdab4af3')
```

```
_id: ObjectId('63e8eb3236ff44f3cdab4af4')
name: "Zidan Zidanov"
▾ quiz: Array
▾ quiz_results: Array
▾ recommendations: Array
__v: 0
```

## I. SQL

### A. Графическое представление




### B. Описание назначений таблиц

| **User** | | |
|---|---|---|
| **Наименование** | **Тип** | **Размеры (байты)** |
| login | VARCHAR | [3; 15] |
| role | VARCHAR | 6 или 7 |
| name | VARCHAR | 20 |

Хранит логин, роль и ссылку на доктора/пациента.
Память: 15+7+20 = 42 байт

| Doctor | | |
|---|---|---|
| **Наименование** | **Тип** | **Размеры** |
| login | VARCHAR | 15 |
| patients | Array | N*8 |

Хранит информацию о имени доктора и его пациентах.
Память: 15+N*8 байт

| Patient | | |
|---|---|---|
| **Наименование** | **Тип** | **Размеры** |
| login | VARCHAR(15) | 15 |
| patientMedia | Foreign Key | 8 |
| quiz | Array | N*8 |
| quiz_results | Array | N*8 |
| recommedations | Array | N*8 |

Хранит имя, лечащего врача, опрос, результат опроса и рекомендации пациента.
Память: $15 + 8 + N_1 * 8 + N_2 * 8 + N_3 * 8 = 23 + 8(N_1 + N_2 + N_3)$ байт

| PatientMedia | | |
|---|---|---|
| **Наименование** | **Тип** | **Размеры** |
| src | Array of VARCHAR(28) | N*28 |
| created_at | DATE | 3 |
| type | VARCHAR(8) | 6 или 8 |

Хранит путь к файлам, дату создания, ссылку на пациента и подписаны ли файлы.
Память: $N * 28 + 29 + 8 = 37 + N * 28$ байт

| Question | | |
|---|---|---|
| **Наименование** | **Тип** | **Размеры** |
| answerType | VARCHAR(8) | 7 |
| text | VARCHAR(20) | 20 |

Хранит тип ответов и текст вопроса.
Память: $7 + 20 = 27$ *байт*

| QuizResult | | |
|---|---|---|
| **Наименование** | **Тип** | **Размеры** |
| date | DATE | 3 |
| Result | Array of Object ({answer: VARCHAR(25), questionId: Foreign Key}) | N*(25+8) |
| quizResultMedia | Foreign Key | 8 |

Хранит дату ответа на опрос и результат (ответ и ссылку на вопрос).
Память: $11 + N * 33$ байт

| QuizResultMedia | | |
|---|---|---|
| **Наименование** | **Тип** | **Размеры** |
| src | ARRAY of VARCHAR(28) | N*28 |
| create_at | DATE | 3 |

Хранит приложенные к опросу файлы (пути, дата создания и ссылку на ответ).
Память: $N * 28 + 3$ байт

| Recommendation | | |
|---|---|---|
| **Наименование** | **Тип** | **Размеры** |
| date | DATE | 3 |
| text | VARCHAR(50) | 50 |

Хранит информацию о рекомендации (дата создания и текст рекомендации).

Память: $3 + 50 = 53$ *байт*

C. Оценка удельного объема информации

Пусть будет 2 пациента, 2 врача (у каждого – по пациенту), каждому пациенту: 12 вопросов в опросе, 2 файла, приложенных к ответу, по рекомендации, 5 документов на подпись.

| Таблица | Объем, байты |
|---|---|
| User | $4 * 42 = 168$ |
| Patient | $2 * (23 + 8 * (12 + 1 + 1)) = 270$ |
| Doctor | $2 * (15 + 8) = 46$ |
| QuizResult | $2 * (11 + 33 * 12) = 814$ |

| | |
|---|---|
| QuizResultMedia | $2 * (3 + 28 * 2) = 118$ |
| PatientMedia | $2 * (37 + 5 * 28) = 354$ |
| Question | $2 * 27 * 12 = 648$ |
| Recommendation | $2 * 53 = 106$ |
| Сумма, байты | |
| $168 + 270 + 46 + 814 + 118 + 354 + 648 + 106 = 2524$ | |

D. Избыточность модели
Vf = Фактический объем = 3324 байт.
Vc = Чистый объем = $2524$ байт.
Vc/Vf = 0.759

E. Запросы к модели

| Описание | Текст | Кол-во задействованных коллекций |
|---|---|---|
| Получение пользователя по логину | Для доктора:<br>SELECT id, name FROM DOCTORS WHERE login = body.login;<br>SELECT id, name FROM PATIENTS WHERE doctor = doctor.id;<br><br>Для пациента:<br>SELECT id, name FROM PATIENTS WHERE login = body.login;<br>Если есть назначенный доктор:<br>SELECT name, id from DOCTORS WHERE id = patient.doctor;<br>Иначе:<br>SELECT name, id from DOCTORS; | 2 |
| Получение пользователя | SELECT * FROM PATIENTS ‘p’ WHERE id = patientId<br>JOIN QUESTIONS ‘q’ on p.quiz = q.id<br>JOIN RECOMMENDATIONS ‘r’ on p.recommnedations = r.id<br>JOIN QUIZRESULTS ‘qr’ on p.quiz_results = qr.id; | 4 |
| Получение опроса по Id пациента | SELECT * FROM PATIENTS ‘p’ WHERE id = patientId<br>JOIN QUIZZES ‘q’ on p.quiz = q.id; | 2 |
| Создание опроса | Получение пациента:<br>SELECT * FROM PATIENTS ‘p’ WHERE id = patientId; | 2 |

| | | |
|---|---|---|
| | Создание вопросов:<br>INSERT INTO QUESTIONS VALUES (answerType, text) | |
| Получение ответов(файлы) на опросы | SELECT * FROM QUIZRESULTMEDIAS WHERE id = QuizResultId; | 1 |
| Создание ответа на опрос | Создание QuizResult, QuizResultMedia:<br>INSERT INTO QUIZRESULTS VALUES (date, result, quizResultMedia);<br>INSERT INTO QUIZRESULTMEDIAS VALUES (src, created_at);<br><br>Поиск пациента: SELECT * FROM PATIENTS ‘p’ WHERE id = patientId; | 3 |
| Получение неподписанных файлов | SELECT * FROM PATIENTMEDIAS WHERE id = patientMedia AND type = ‘Unsigned’ | 1 |
| Создание неподписанных документов | INSERT INTO PATIENTMEDIAS VALUES (src, created_at, type); | 1 |
| Получение подписанных файлов | SELECT * FROM PATIENTMEDIAS WHERE id = patientMedia AND type = ‘Signed | 1 |
| Создание подписанных документов | INSERT INTO PATIENTMEDIAS VALUES (src, created_at, type); | 1 |
| Получение рекомендаций | SELECT * FROM PATIENTS ‘p’ WHERE id = patientId JOIN RECOMMENDATIONS ‘r’ on p.recommendations = r.id; | 1 |
| Создание рекомендации | INSERT INTO RECOMMENDATIONS VALUES (text, date);<br>Поиск пациента:<br>SELECT * FROM PATIENTS ‘p’ WHERE id = patientId; | 2 |
| Поменять доктора | UPDATE PATIENTS<br>SET<br>doctor = doctorId<br>WHERE<br>id = patientId | 1 |

**Сравнение моделей.**

Сравнив объемы SQL и NoSQL баз данных, можно сказать, что во втором случае БД занимает меньше места. Коллекций использовано одинаковое количество. Из-за того, что SQL не обходится без JOIN, NoSQL выигрывает по времени. Однако NoSQL использует дублирование данных, в результате чего запросы на удаление/изменение могут также быть затратными по времени.

В результате, обе модели имеют свои преимущества и минусы. При наличии ограничений на типы данных и структуру хранящейся информации, следует выбрать реляционную модель. При высоких требованиях к скорости работы и нечетких требованиях к типам данных следует выбрать нереляционную модель.

## 5. Разработанное приложение

### Краткое описание

Приложение реализовано на React+typescript+Nodejs, для с работы с MongoDB используется библиотека mongoose[3]. При помощи нее  создаются подключение к бд, создание коллекций для хранения информации о пациентах, доктора, подписанных и неподписанных документах, а также об опросах и рекомендациях. Для загрузки файлов использовалась библиотека multer[4]. На фронтенде для создания интерфейса была использована библиотека MUI.

### Схема экранов приложения

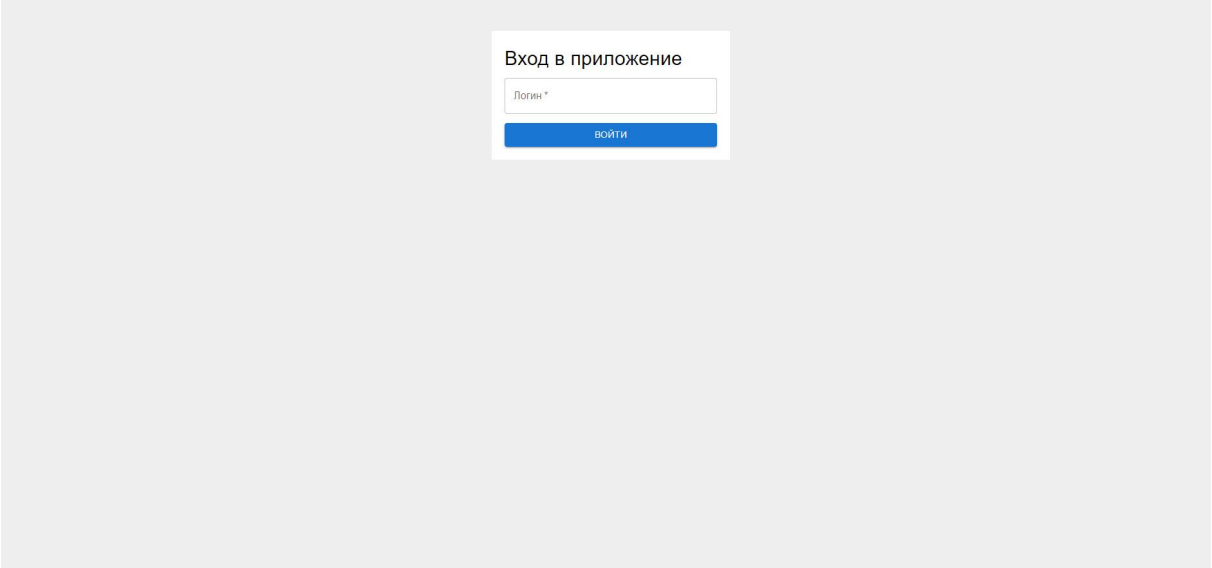



Рис. 2 Экран входа

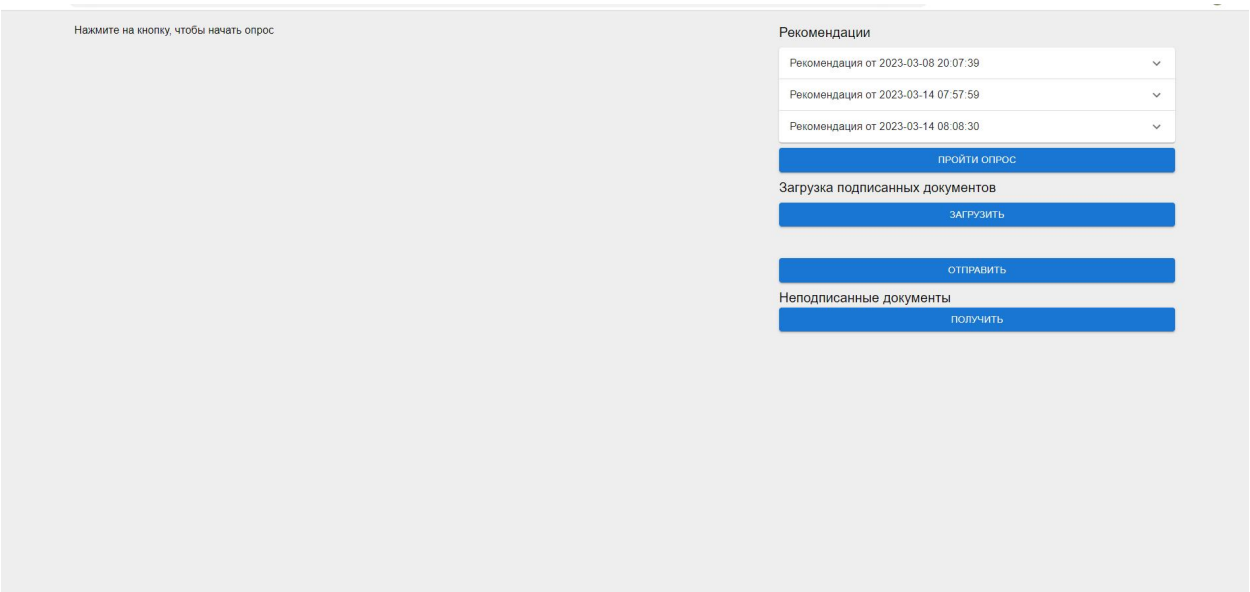



Рис. 3 Экран пациента

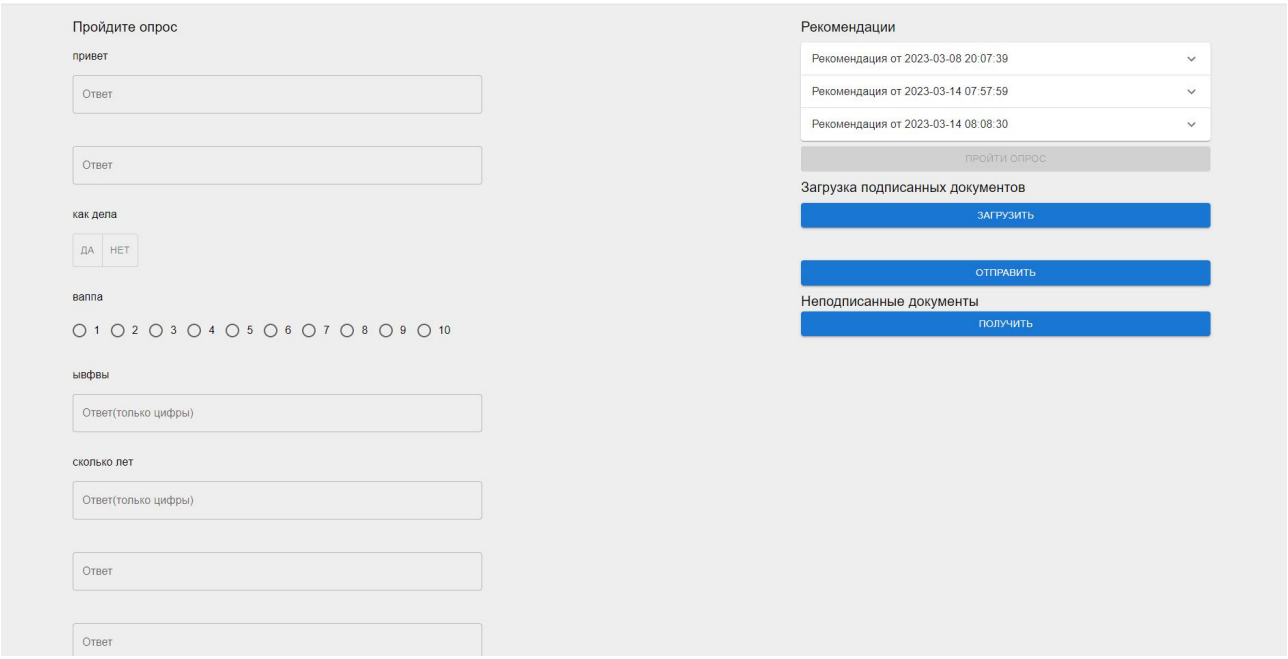



Рис. 4 Экран пациента с открытым опросом

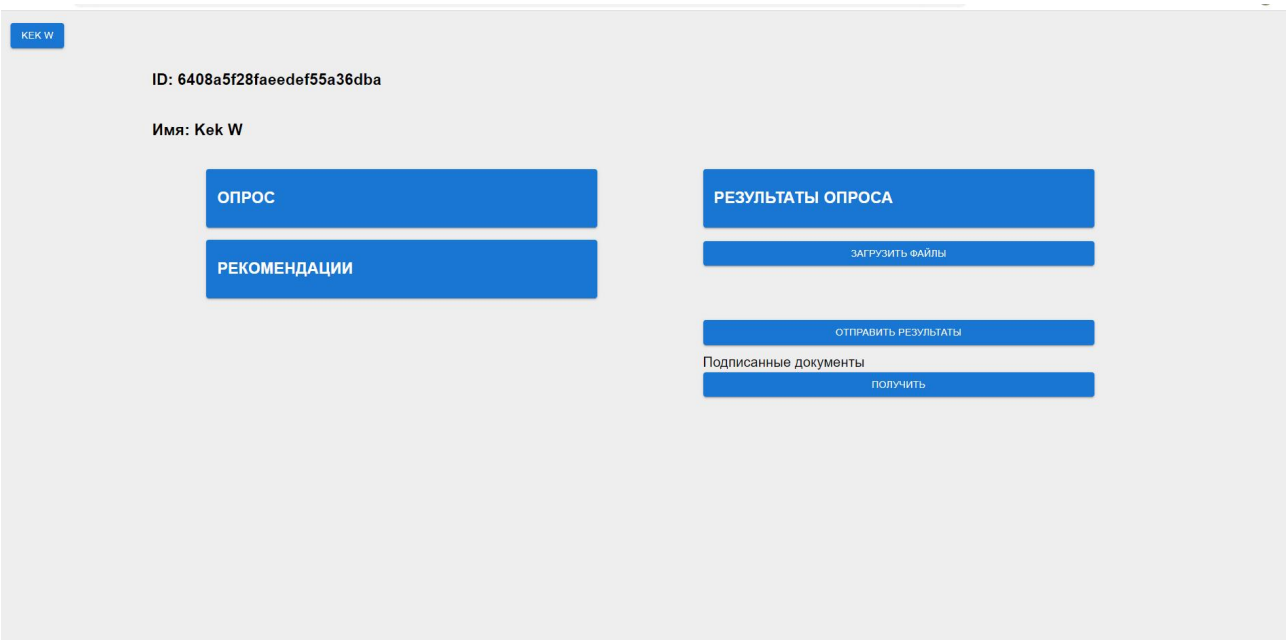



Рис. 5 Экран доктора

## Использованные технологии

СУБД: MongoDB

Backend: Nodejs, Mongoose, multer

Frontend: React, typescript, mui, axios, redux-toolkit, sass .

## Ссылки на приложение

Ссылка на github: https://github.com/moevm/nosql2h22-monitoring

# 6.Вывод

В ходе работы было разработано web-приложение мониторинга пациентов, позволяющее пользователям взаимодействовать с базой данных: просмотр содержимого СУБД, добавление новых элементов, как текстовых, так и файловых.

## Будущее развитие решения

Планируется улучшить пользовательский интерфейс, добавить больше логики: статистика опросов, графики прохождения опрос, подписания документов при помощи цифровых подписей.