**­**

|  |  |
| --- | --- |
|  | **Министерство науки и высшего образования Российской Федерации**  **Федеральное государственное бюджетное образовательное учреждение**  **высшего образования**  **«Заполярный государственный университет**  **им. Н.М. Федоровского**  **(ЗГУ им. Н.М. Федоровского)** |

**­**

**Факультет «Электроэнергетики, экономики и управления»**

**Кафедра «Информационные системы и технологии»**

Дисциплина «Разработка Web-приложений»

Расчетно-пояснительная записка

Тема: «Сервис привязки участников ВОВ к историческим документам»

Студент: Иванов И.И.

Группа: ИС-21

Преподаватель: к.э.н.,

доцент Беляев И.С.

2024г.

# **ВВЕДЕНИЕ**

В настоящее время в связи с увеличением периода, прошедшего с окончания Великой Отечественной войны, всё более доступными становятся исторические архивы. В 2011 году около 30 млн. страниц архивных документов было оцифровано, к 2022 году их число достигло почти 110 млн. Доподлинно известно, что количество военнослужащих в РККА к июню 1945 насчитывало 11,5 млн. человек, причём за всё время ВОВ погибшими признано 8,86 млн. военнослужащих. Возникают проблемы как с централизированным хранением данных о ветеранах и архивных документов, так и с обеспечением взаимосвязи между участником ВОВ и историческим материалом, содержащим о нём информацию. Исходя из данных проблем, предлагается система, где к каждому ветерану посредством формирования заявок привязываются исторические документы, проверка подлинности которых будет проверять отдельным специализированным сервисом.

Целью работы является реализация системы для поиска исторических документов и формирования заявок на привязку выбранных документов к ветерану ВОВ, включающую в себя веб-сервис, веб-приложение, десктопное приложения и асинхронный сервис проверки подлинности исторических документов.

Система предназначена для участников проекта и архивистов. Система позволяет выбрать документы из списка, которые желает привязать участник к ветерану. Для этого пользователю необходимо создать заявку, включив туда желаемые документы. Система предоставляет автоматизированный способ создания, учета и ведения заявок. Также она позволяет архивистам принимать или отклонять заявки. Участники имеют возможность добавлять новых ветеранов и исторические документы. Архивисты – редактировать существующие записи ветеранов и документов и удалять в случаях некорректности данных.

Нефункциональные требования к разрабатываемой системе:

* 1. Должна поддерживаться кроссплатформенность.
  2. Интерфейс системы и текст ошибок должны быть русифицируемы.

В ходе работы необходимо выполнить следующие задачи:

* 1. Разработать дизайн приложения.
  2. Создать базу данных в PostgreSQL.
  3. Создать веб-сервис.
  4. Реализовать интерфейс гостя на технологии React.
  5. Развернуть веб-приложение React на Github Pages.
  6. Добавить авторизацию и аутентификацию в веб-сервис.
  7. Реализовать интерфейс участника проекта в React.
  8. Реализовать интерфейс модератора и администратора в React.
  9. Создать десктопное приложение на Tauri.
  10. Создать асинхронный сервис проверки подлинности документа.
  11. Подготовить набор документации, включающий РПЗ, ТЗ и набор диаграмм.

# **БИЗНЕС-ПРОЦЕСС**

Существует множество проблем в области сохранения исторической памяти. Далеко не каждая система [1] включает полный цикл проверки исторических документов, участников событий и т.п. К архивным материалам будем относить: боевые сводки, наградные листки, журналы боевых действий и др. Необходимо автоматизировать процесс и объединить механизмы по обработке, проверки подлинности и удобному представлению базы данных ветеранов ВОВ и исторических документов. Подобный функционал реализован информационной системой «Память народа», которая позволяет участникам проекта прикреплять архивные документы, относящиеся к определённой категории («Документы частей», «Награждения» и др.). В случае добавления документа, прямо связанного с ветераном, требуется его указать в документе или если ветеран не добавлен в систему, заполнить о нём информацию. Таким образом, посетители портала и специалисты-историки смогут ознакомиться со списком всех материалов: базе данных ветеранов и связанных с ними архивных документов. При разработке проекта, тождественному по цели «Памяти народа», необходимо опираться на функциональность последнего, перенимая положительный пользовательский опыт. Участники проекта должны иметь возможность добавлять участников войны и сканы с описанием документов, а также оставлять заявки на привязку определенного ветерана к историческим документам.

Когда участник проекта определится с документами и ветераном, он формирует заявку, которая отправляется на модерирование. После этого её уже нельзя редактировать. Можно также посмотреть историю своих заявок. Архивисты могут принимать заявки или же отклонять их. Благодаря расширению охвата исследований разнообразных исторических документов, некоторые из них могут устареть или позже быть признанными недействительными [2]. В связи с этим архивисты должны иметь возможность редактировать информацию о документах и ветеранах, чтобы поддерживать данные актуальными. Таким образом, будет поддерживаться относительный баланс между добавлением документов и ветеранов ВОВ со стороны участников и их обработкой – архивистами. Функции пользователей с разными ролями описаны на диаграммах прецедентов (рис. 1).

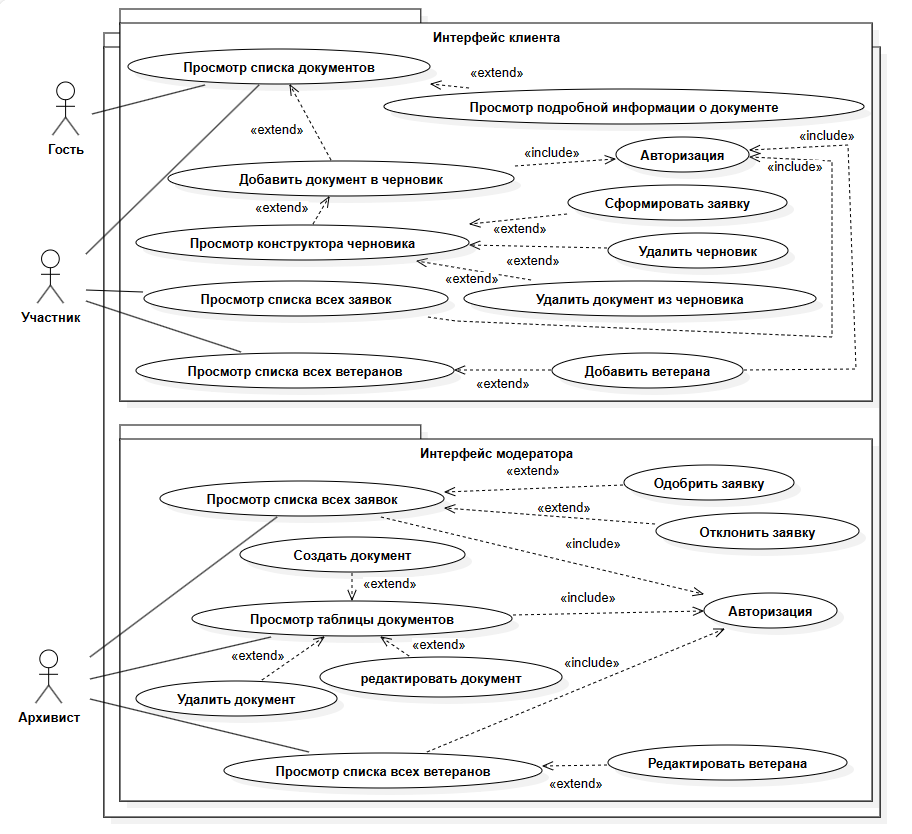


Рисунок 1 - Диаграмма прецедентов

Гостям доступен просмотр документов. Зарегистрированные гости – участники проекта. Они могут добавлять документы в заявку, просматривать список своих заявок и сформировывать текущую заявку, добавлять документы в заявку, удалять их оттуда, либо полностью очистить черновую заявку. Отправленные заявки обрабатываются архивистами. В результате обработки заявки её либо одобряют, либо отклоняют. Также, архивисты могут просматривать список документов в табличном виде, редактировать отдельные документы, либо же создавать новые [3]. Участники также умеют добавлять документы, просматривать и добавлять ветеранов. Процесс оформления заявки отражен на диаграмме деятельности (рис. 2).

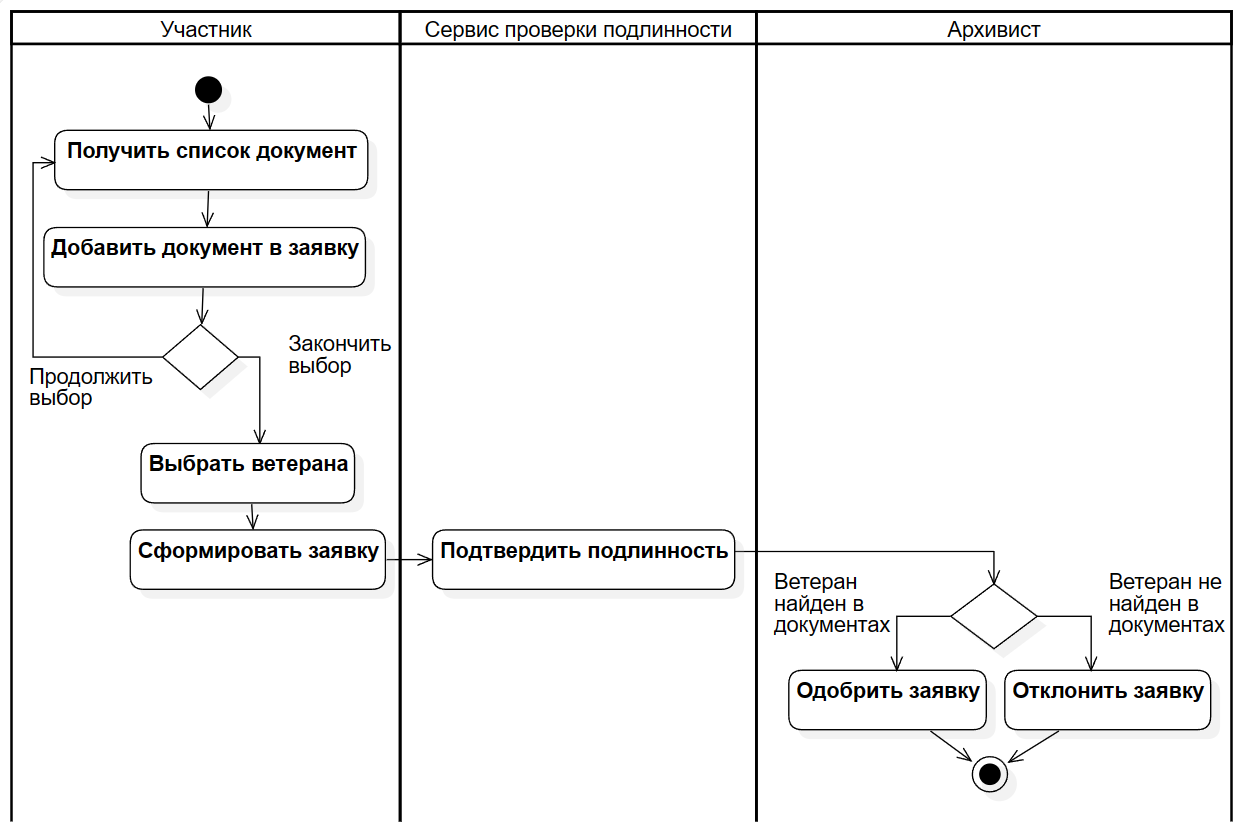


Рисунок 2 - Диаграмма деятельности

Участник выбирает документ, затем формирует на основе выбранных документов заявку. Эту заявку затем обрабатывает асинхронный сервис, а затем и архивист. Возможные состояния заявки отражены на диаграмме состояний (рис. 3).



Рисунок 3 - Диаграмма состояний заявки

При выборе первого документа формируется черновик. Последующие выбранные документы добавляются в этот черновик. Участник затем формирует заявку, удаляет её или выходит из приложения. Сформированную заявку обрабатывает архивист. Они могут одобрить или отклонить её.

# **АРХИТЕКТУРА**

Архитектура системы отображена на диаграмме развертывания (рис. 4).

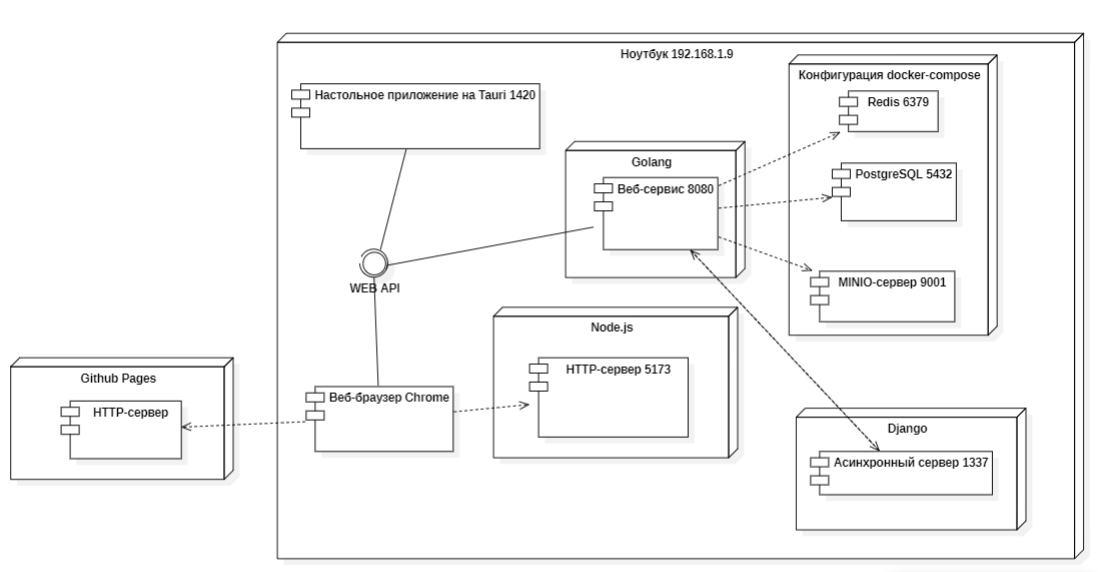


Рисунок 4 - Диаграмма развертывания

Фронтенд был реализован с помощью JS-библиотек – React [5,6] и Redux более удобного управления состояниями приложения, Redux отделяет логику обновлений состояний от основной и упрощает отслеживание изменения данных [8]. Бекенд разворачивается в кластере докер контейнеров. Такое решение было принято в связи с тем, что технология докер контейнеров позволяет быстро и удобно разворачивать целые системы на любых устройствах, поддерживающих докер. Контейнеры включают в себя все зависимости и библиотеки, необходимые для выполнения приложения. Это гарантирует, что приложение будет работать одинаково на разных средах, избегая проблем с зависимостями, а также контейнеры обеспечивают высокий уровень изоляции, что позволяет запускать приложения в изолированных средах без влияния на другие приложения. Еще одним неоспоримым преимуществом докер контейнеров является то, что контейнеры легко масштабируются в зависимости от изменяющихся требований. Конфигурация контейнеров описана в docker-compose.yml файле [4]. Асинхронный сервис проверки подлинности разворачивается отдельно и необязательно на том же компьютере, что и кластер основных контейнеров.

Данные хранятся в СУБД PostgreSQL [7], их структура отражена на ER диаграмме (рис. 5). СУБД PostgreSQL является одним из стандартов индустрии, поэтому было решено использовать её. Часть данных хранится в MinIO – высокопроизводительном объектном хранилище, в частности, там лежат изображения консультаций. Redis используется для хранения недействительных токенов. Интерфейс гостя развернут на GitHub pages.

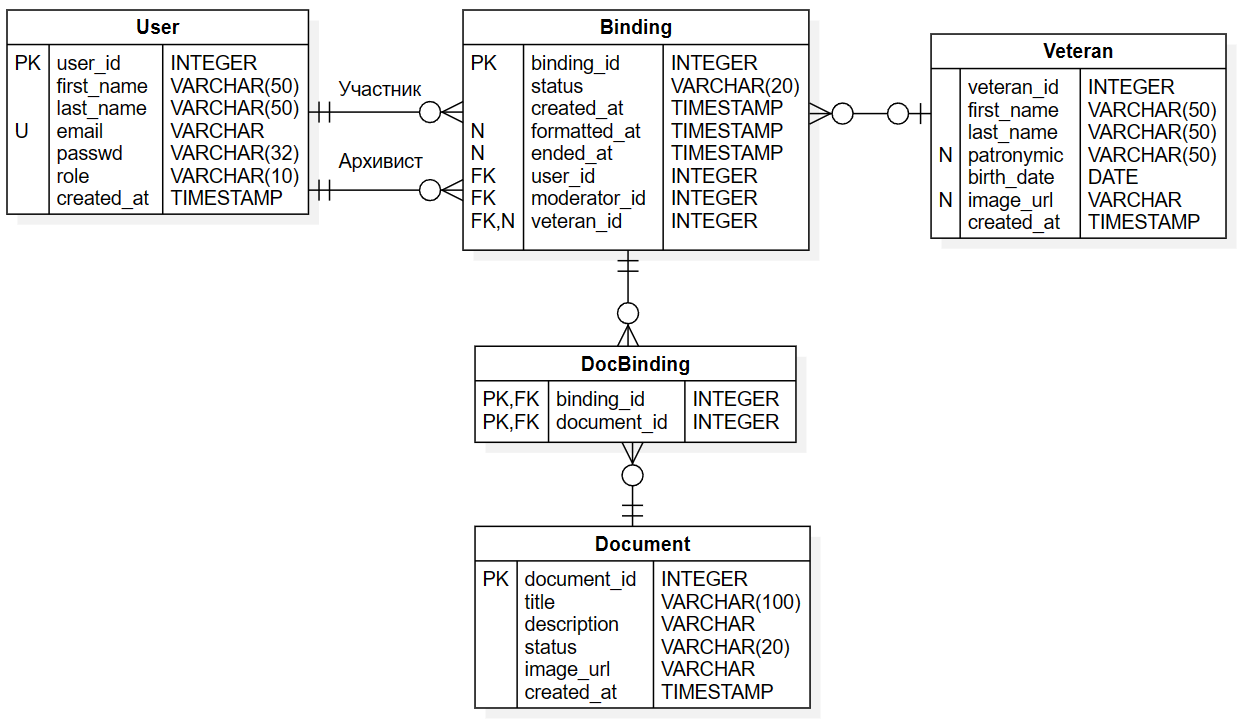


Рисунок 5 - ER диаграмма

Рассмотрим структуры данных. Помимо базовых полей с датами создания, формирования и завершения, в заявке также есть поле, устанавливающее связь с ветераном, выбранным в данной заявке. Модель ветерана включает его ФИО, дату рождения и изображение при наличии. Модель документов представляет собой набор полей, необходимых исключительно для бизнес-логики: заголовок, текстовое описание и изображение. Модель пользователя содержит контактные данные и поля, необходимые для авторизации и аутентификации. Для хранения в одной заявке нескольких документов используется промежуточная таблица doc-binding, реализующая связь М-М.

Устройство бекенда приложения изображено на диаграмме классов бекенда (рис 6.). Модели имеют связи с таблицами в базе данных. Также некоторые модели имеют связи с внешними сервисами. В частности, документы и ветераны имеют связь с сервером статических файлов, т.к. в них хранится ссылка на их изображение, которое лежит на сервере статических файлов [9].

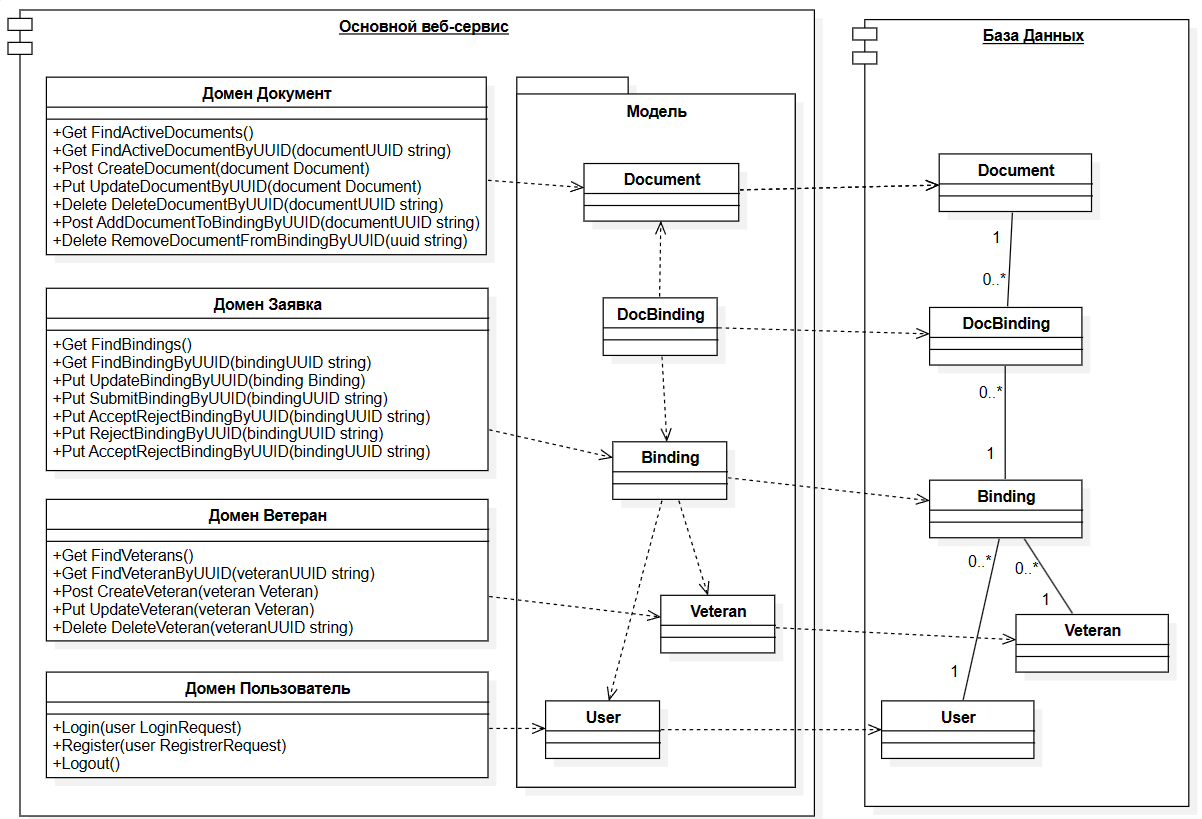


Рисунок 6 *-* Диаграмма классов бекенда

Связь фронтенда и бекенда отражена на диаграмме классов фронтенда (рис. 7). Ключевые страницы имеют связь с API аутентификации, т.к. доступ к ним осуществляется только для авторизированных пользователей с определенными правами (ролями).

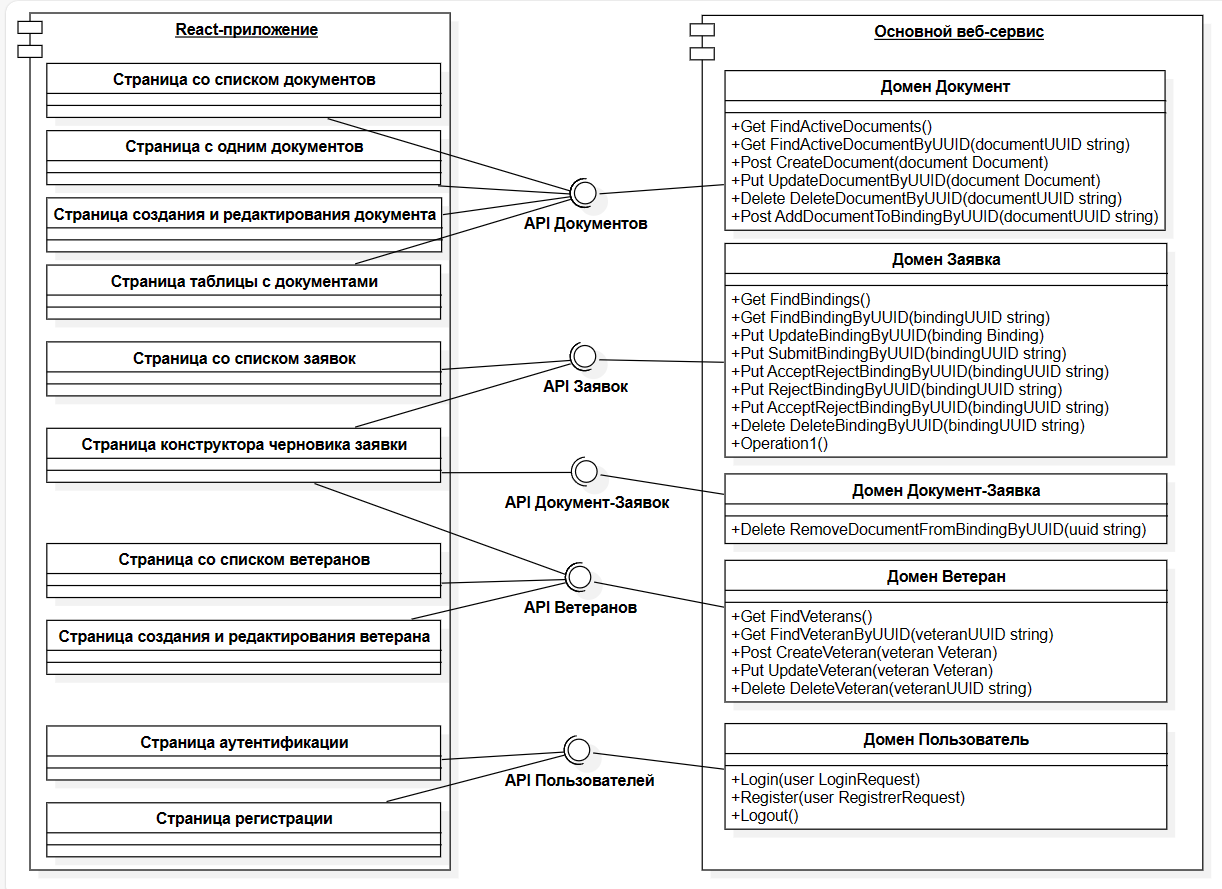


Рисунок 7 *-* Диаграмма классов фронтенда

# **АЛГОРИТМЫ**

Алгоритм работы системы отображен на диаграмме последовательности (рис. 8). В основе системы лежит веб-сервис, реализующий внутри себя всю бизнес-логику. Он предоставляет доступ к методам из следующих доменов: пользователя, документа, заявки. Методы следуют правилам REST API.

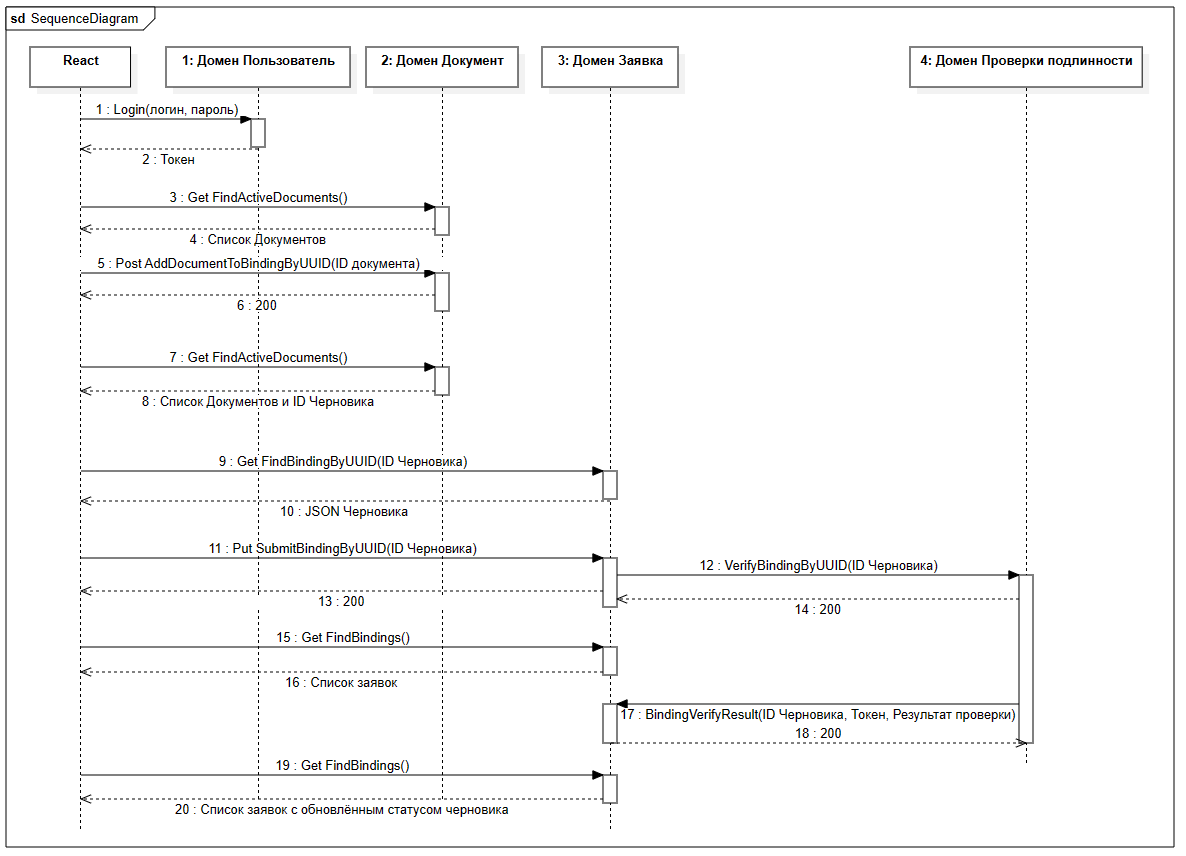


Рисунок 8 - Диаграмма последовательности

В начале бизнес-процесса происходит аутентификация участника. Для этого он отправляет через графический интерфейс запрос, передавая в нем логин и пароль. Если аккаунт с такими данными существует, то участник получает токен в ответном запросе. Если же такого аккаунта не существует, или пароль введен неверно, участник получит ошибку. В таком случае ему надо либо пройти регистрацию, либо ввести свои данные для входа верно. Затем графический интерфейс запрашивает у веб-сервиса список документов, которые возвращаются в JSON формате. Участник проекта выбирает документ, которые хочет привязать за определенным ветераном, и, нажимая на кнопку «Добавить» в графическом интерфейсе, отправляет запрос на добавление документа в свою черновую заявку. Этот процесс может продолжаться по многу раз.

Когда участник определится с выбором документов и ветерана, он переходит в конструктор заявки и нажимает на кнопку «Отправить» в графическом интерфейсе. После этого приложение запрашивает id черновой заявки участника проекта и затем отправляет запрос на формирование этой заявки. В этот момент основной веб-сервис выполняет асинхронный запрос к сервису проверки подлинности на то, чтобы проверить качество выбранных документов и получить разрешение на формирование заявки. Когда заявка будет одобрена, участник сможет увидеть через некоторое время, что его заявка получила новый статус.

Процесс рассмотрения заявок происходит также через графический интерфейс архивиста. Архивисты могут просматривать списки всех заявок и, нажимая на соответствующие элементы управления, отправлять запросы на одобрение или отклонение заявок в основной веб-сервис. В эти запросы также можно включить фильтры по имени создателя заявки, диапазону дат и статусу заявок, в котором должны быть заявки. Кроме обработки заявок, архивисты также могут через графический интерфейс управлять непосредственно документами. Им доступны такие функции, как создание и редактирование документов, просмотр списка документов в табличном формате и их удаление. Для каждой из этих функция присутствует свой метод, отправляемый на основной веб-сервис.

# **ОПИСАНИЕ ИНТЕРФЕЙСА**

Главное меню приложения включает пункты, которые доступны в зависимости от роли (рис. 9-10).

****

Рисунок 9 - Главное меню (гость)



Рисунок 10 - Главное меню (авторизированный участник)

Меню видно для всех пользователей и гостей. В зависимости от роли его содержимое меняется. Для гостей, например, там отображаются только кнопки «Войти», «Ветераны» и «Документы», а для участников проекта – «Ветераны», «Документы», «Заявки» и иконка выхода.

На странице с формой авторизации (рис. 11) отображается форма, через которую гость входит в свой аккаунт. При успешном вводе логина и пароля гость получает JWT, который сохраняется в local storage и используется в хедере Authorization при отправлении запросов.

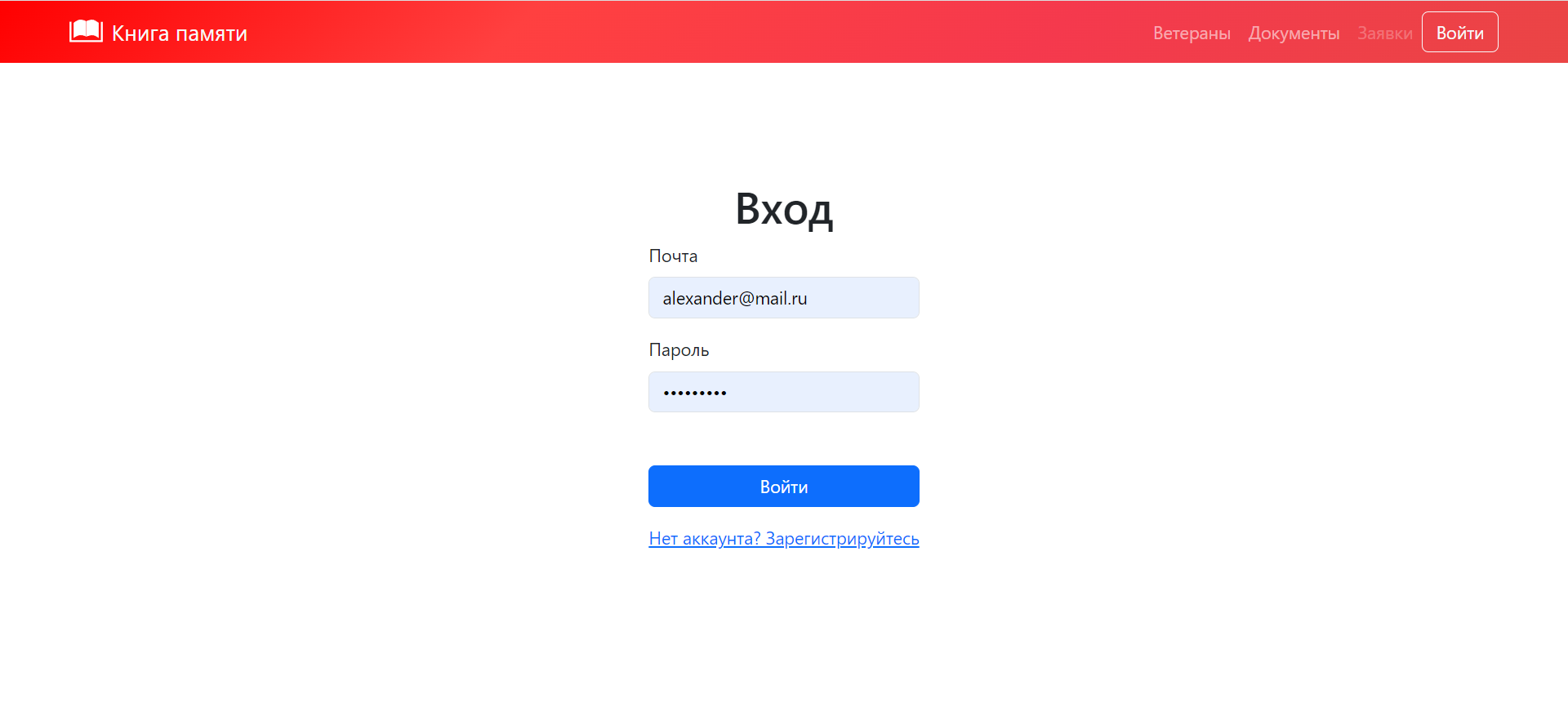
****

Рисунок 11 - Страница авторизации

На странице с формой регистрации (рис. 12) гости могут завести аккаунт. Для этого нужно указать имя с фамилией, почту и пароль. Если введенный логин, телефон, email уже заняты, то система попросит сменить данные.

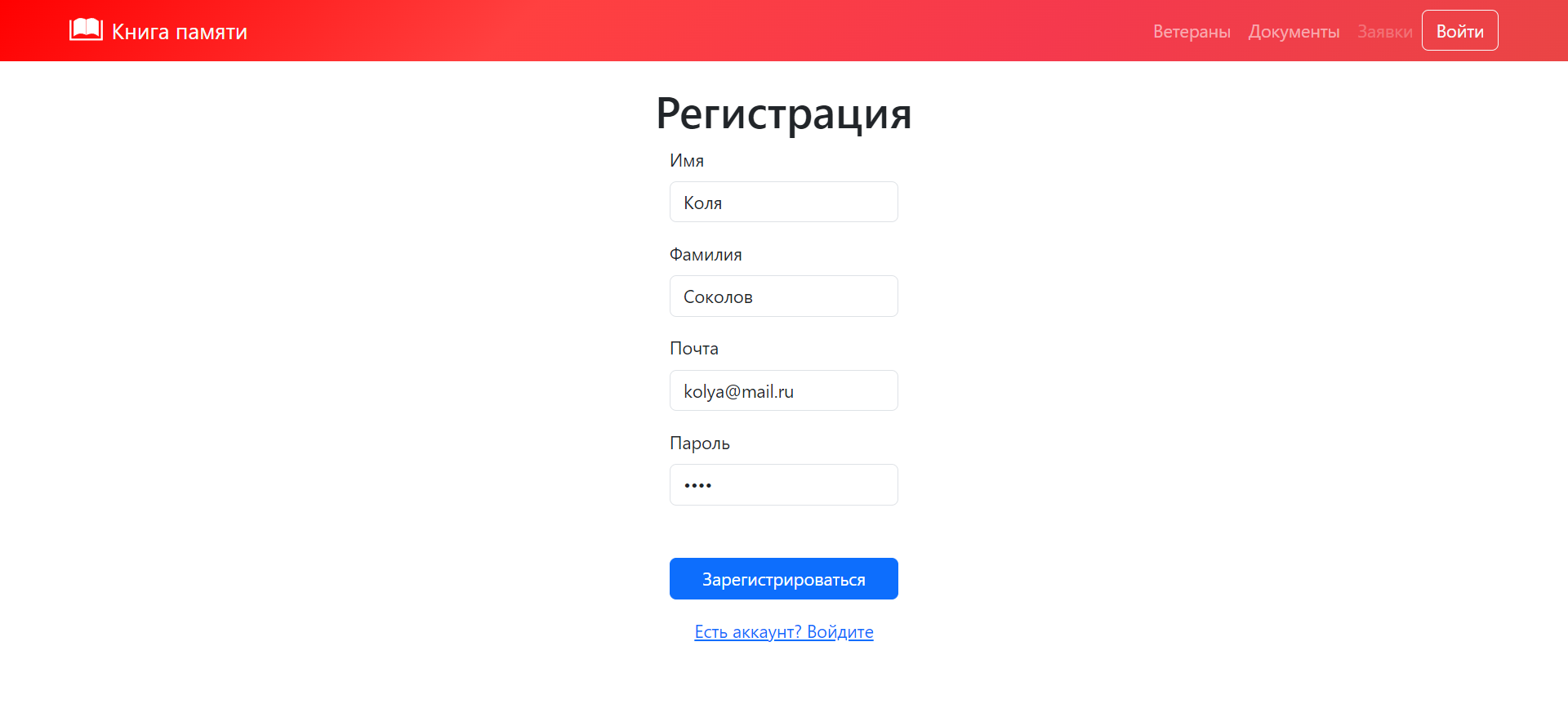
****

Рисунок 12 - Страница регистрации

На странице со списком документов (рис. 13) данные отображаются в виде списка карточек. На каждую карточку можно нажать и тогда участник проекта переносится на страницу с подробной информацией об историческом документе. Также на карточках есть кнопка «Добавить в заявку», которая добавляет документ в черновую заявку. Над списком находится поиск документов по названию вне зависимости от регистра и положения слова. В правом нижнем углу расположена иконка корзины, которая становится активной при наличии хотя бы одной консультации в заказе.

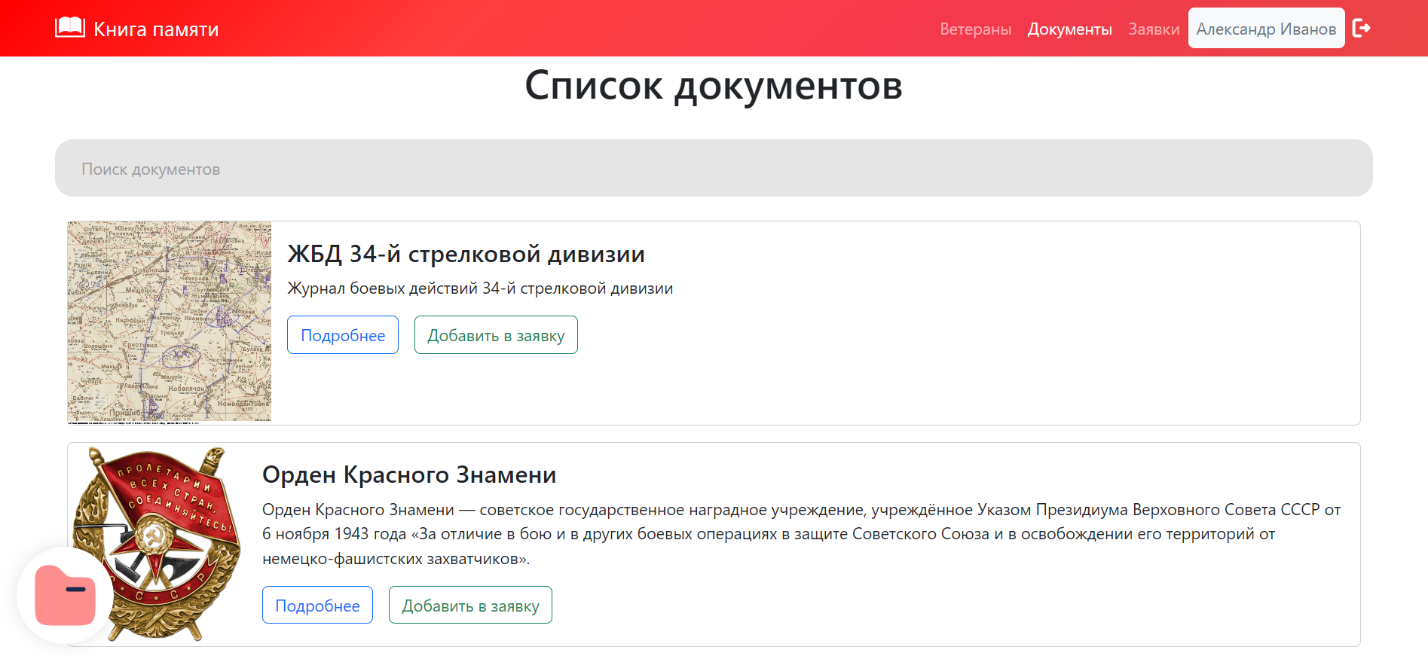
****

Рисунок 13 - Страница со списком документов

На странице с подробным описанием консультации (рис. 14) отображается подробная информация о документе: название, полное текстовое описание и изображение.

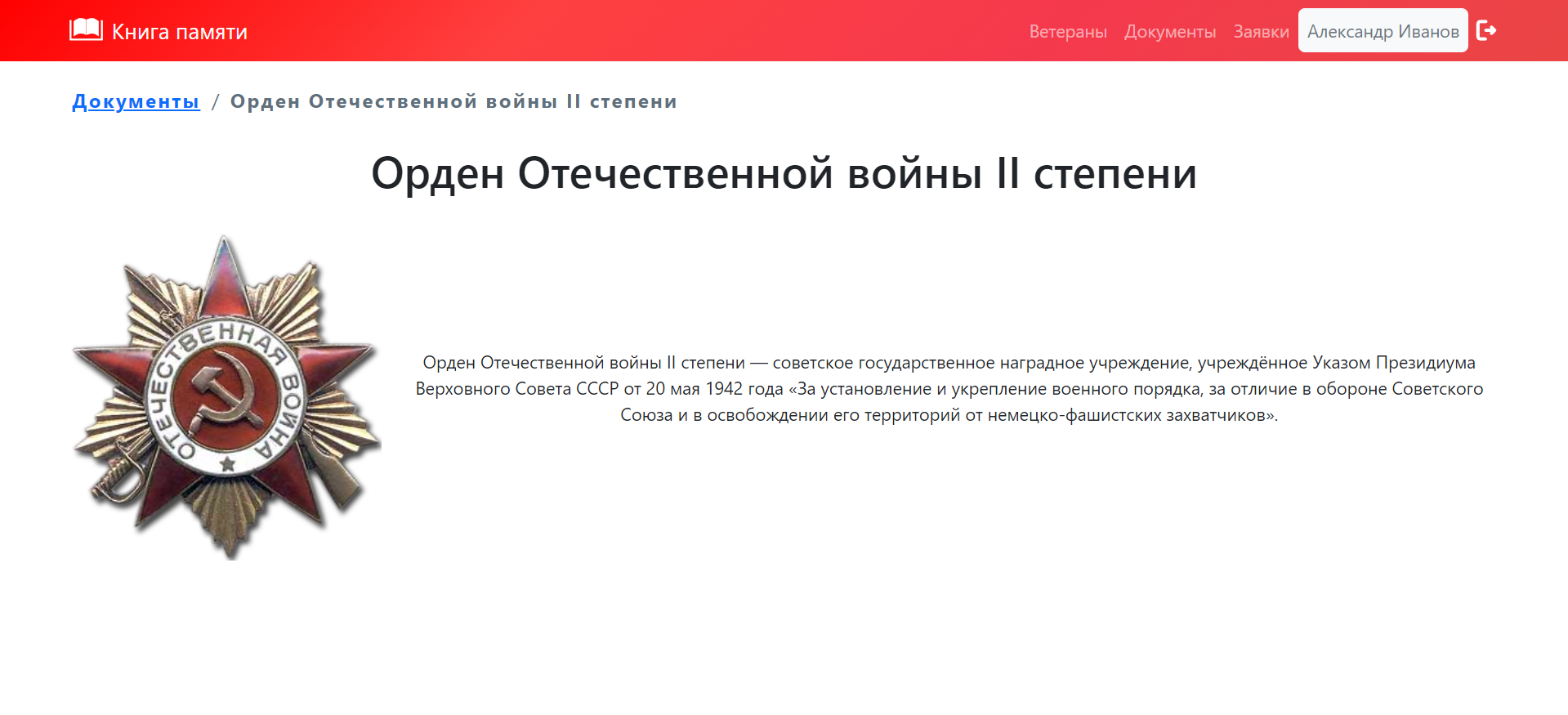


Рисунок 14 - Страница с подробным описанием документа

На странице со списком заявок (рис. 15, 16) отображается список заявок. В зависимости от роли этот список будет функционально отличаться. Так, для пользователей отображается список созданных ими заявок: статус, дата создания, дата формирования, дата завершения. При нажатии на одну из строк таблицы появляется кнопка «Просмотреть».

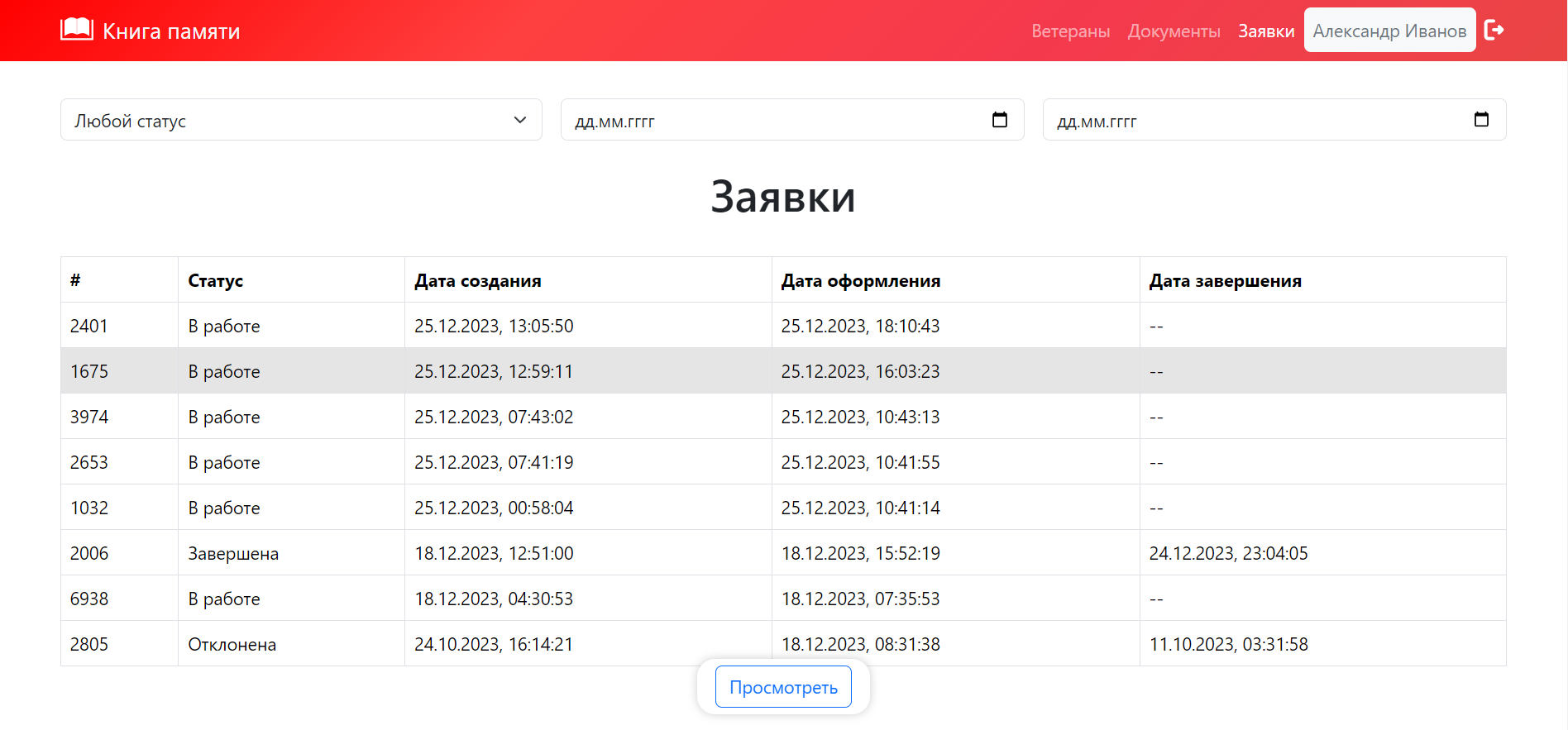
****

Рисунок 15 - Страница со списком заявок (участник)

Для архивиста функционал этой страницы шире. Для него отображается список всех заявок всех участников с дополнительным полем создателя заявки. А также архивист может отфильтровать заявку по: начальной дате формирования, конечной дате формирования, имени отправителя и статусу заявки. Те из заявок, администрировать которые назначен авторизированный архивист, подсвечены иконкой с восклицательным знаком. При нажатии на такую строку появляется ещё несколько элементов управления: кнопка «Просмотреть», «Принять» и «Отклонить».

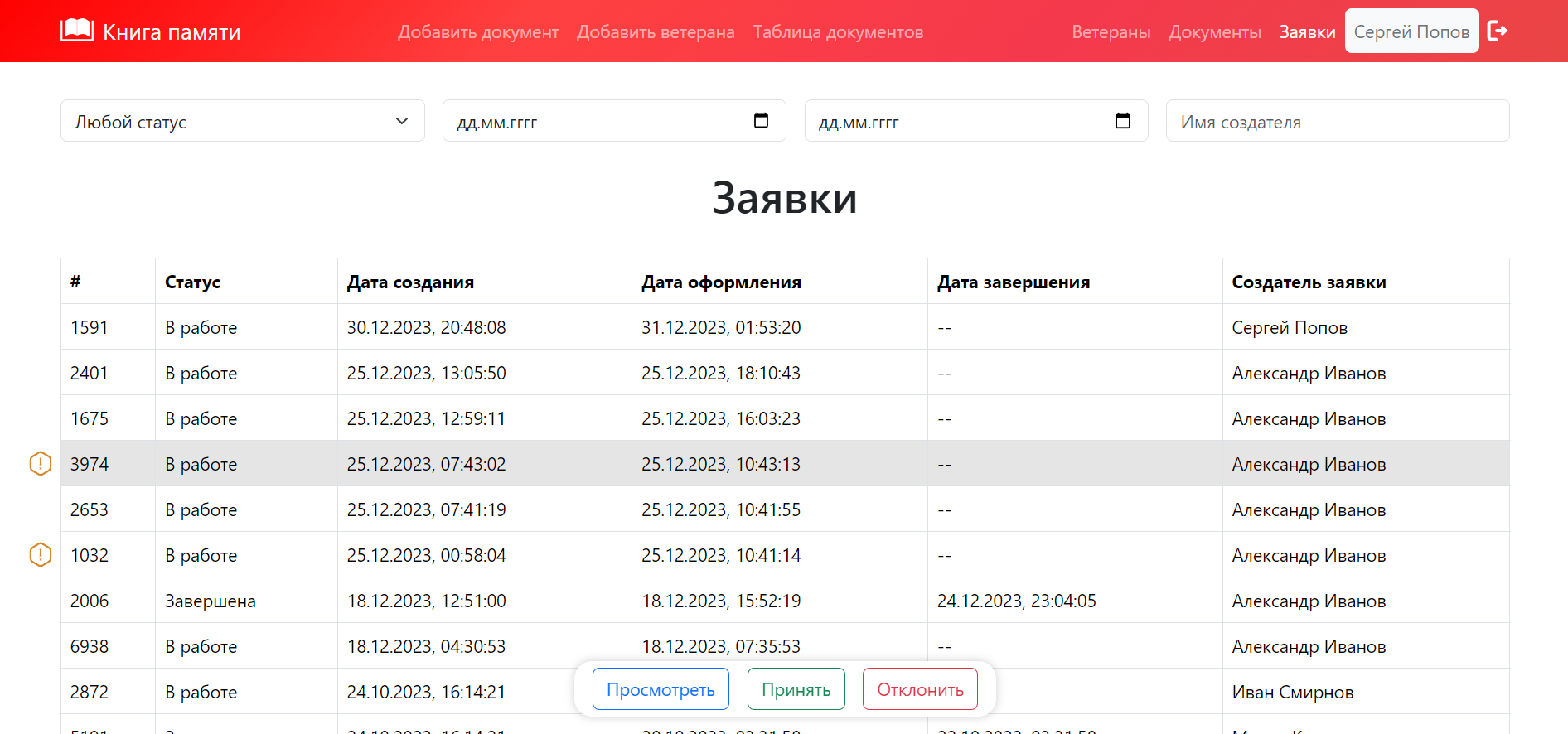


Рисунок 16 - Страница со списком заявок (архивист)

На странице с таблицей документов (рис. 17) архивист может в компактном и удобном формате просмотреть список всех документов, существующих в системе. Отображаются следующие поля: изображение, название, описание. Также можно перейти к окну редактированию документа (рис. 18).

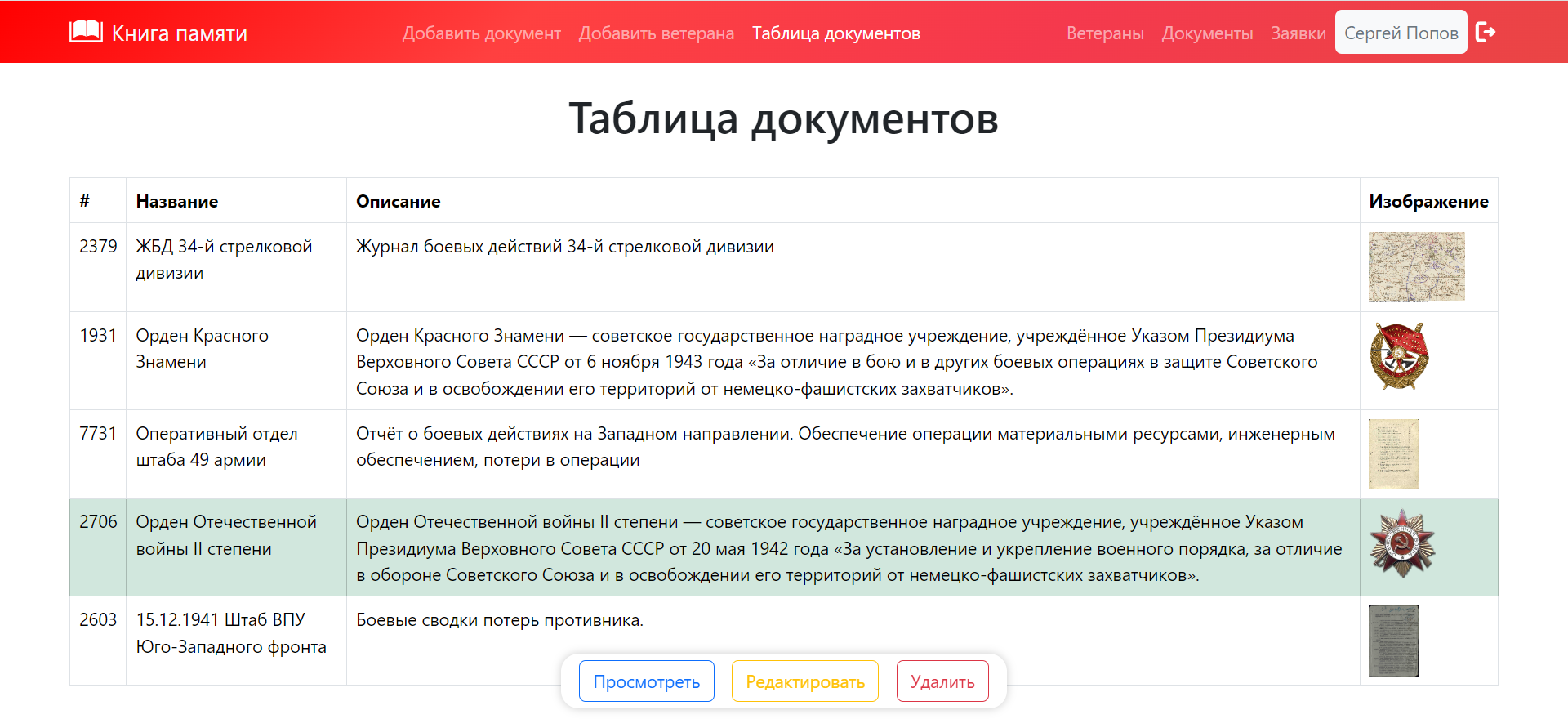


Рисунок 17 - Страница с таблицей консультаций

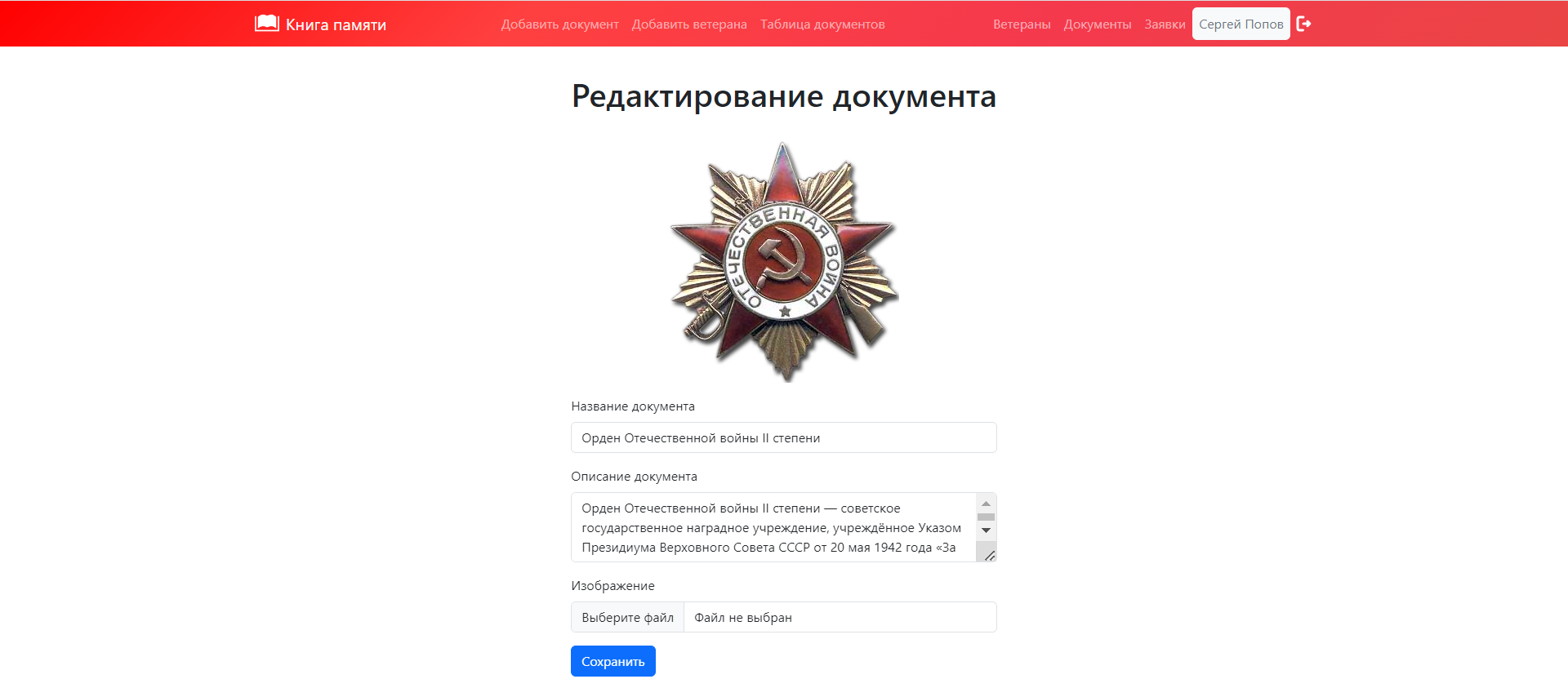


Рисунок 18 - Страница редактирования документа

Участник проекта имеет возможность добавить исторический документ на соответствующей странице (рис. 19). Для этого необходимо загрузить фотографию или скан, заполнить название документа и описать его содержание. После сохранения (кнопка «Сохранить») документ отобразиться в списке.

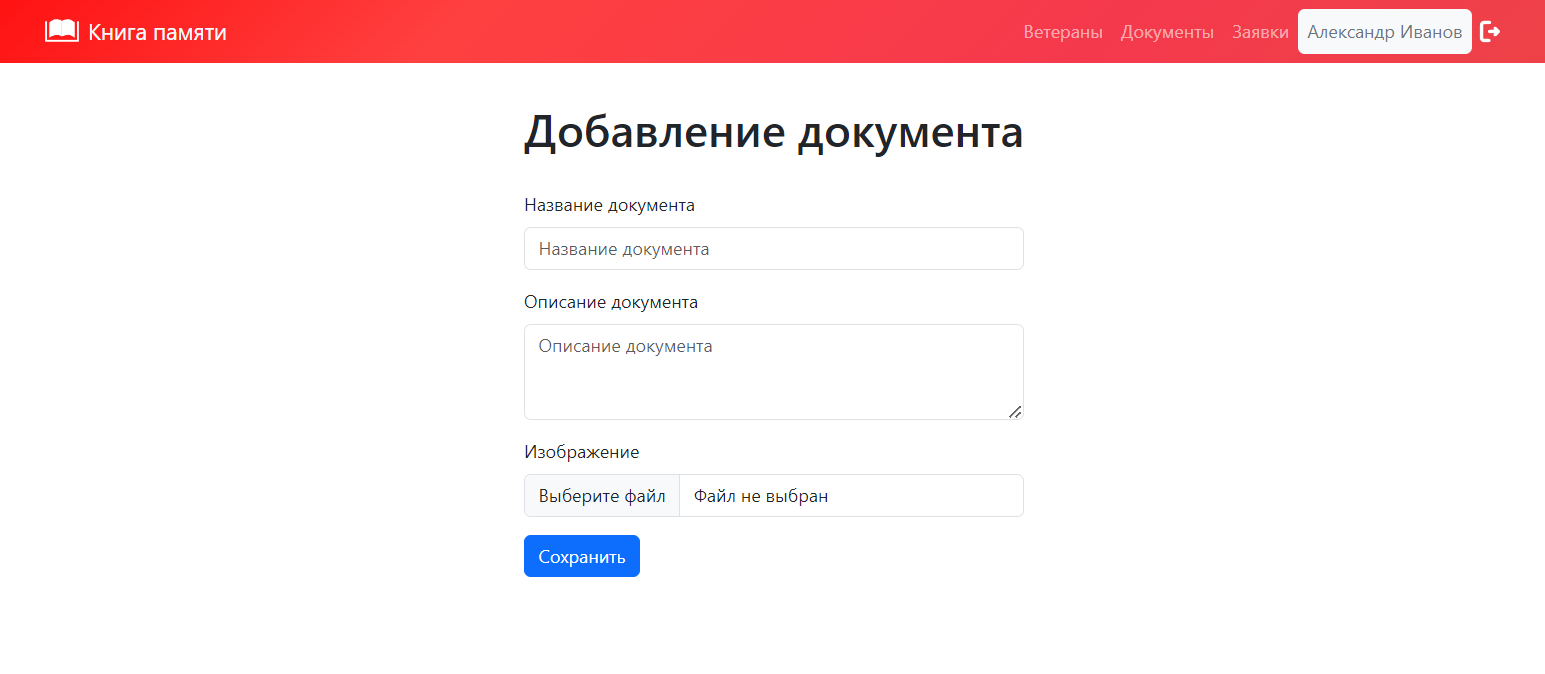


Рисунок 19 - Страница добавления документа

Для участника проекта также доступна страница корзины (рис. 20), где в выпадающем списке можно выбрать ветерана. Если один из документов добавлен по ошибке, то на странице корзины его возможно удалить, а по нажатию на кнопку «Подробнее» на карточке документа осуществляется переход на более подробную информацию о нём. Черновую заявку можно сформировать (кнопка «Оформить заявку») и удалить (кнопка «Удалить заявку»).

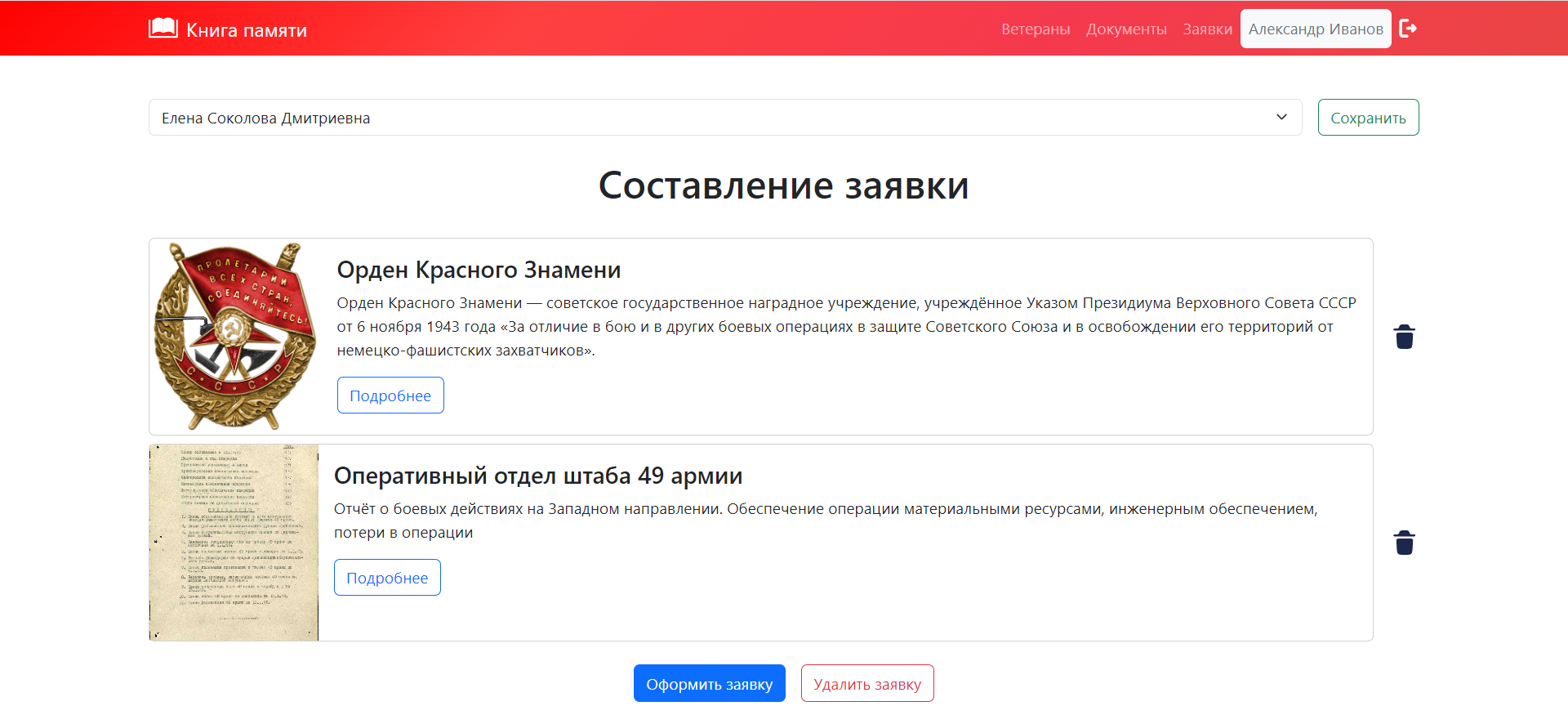


Рисунок 20 - Страница конструктора черновика заявки

Страница со списком ветеранов представляет список карточек ветеранов с полями ФИО и датой рождения участника ВОВ. Можно осуществлять поиск среди ветеранов по ФИО (рис. 21). У архивиста интерфейс более расширенный (рис. 22): он имеет возможность перейти на страницы редактирования ветерана (рис. 23) по дополнительным кнопкам.

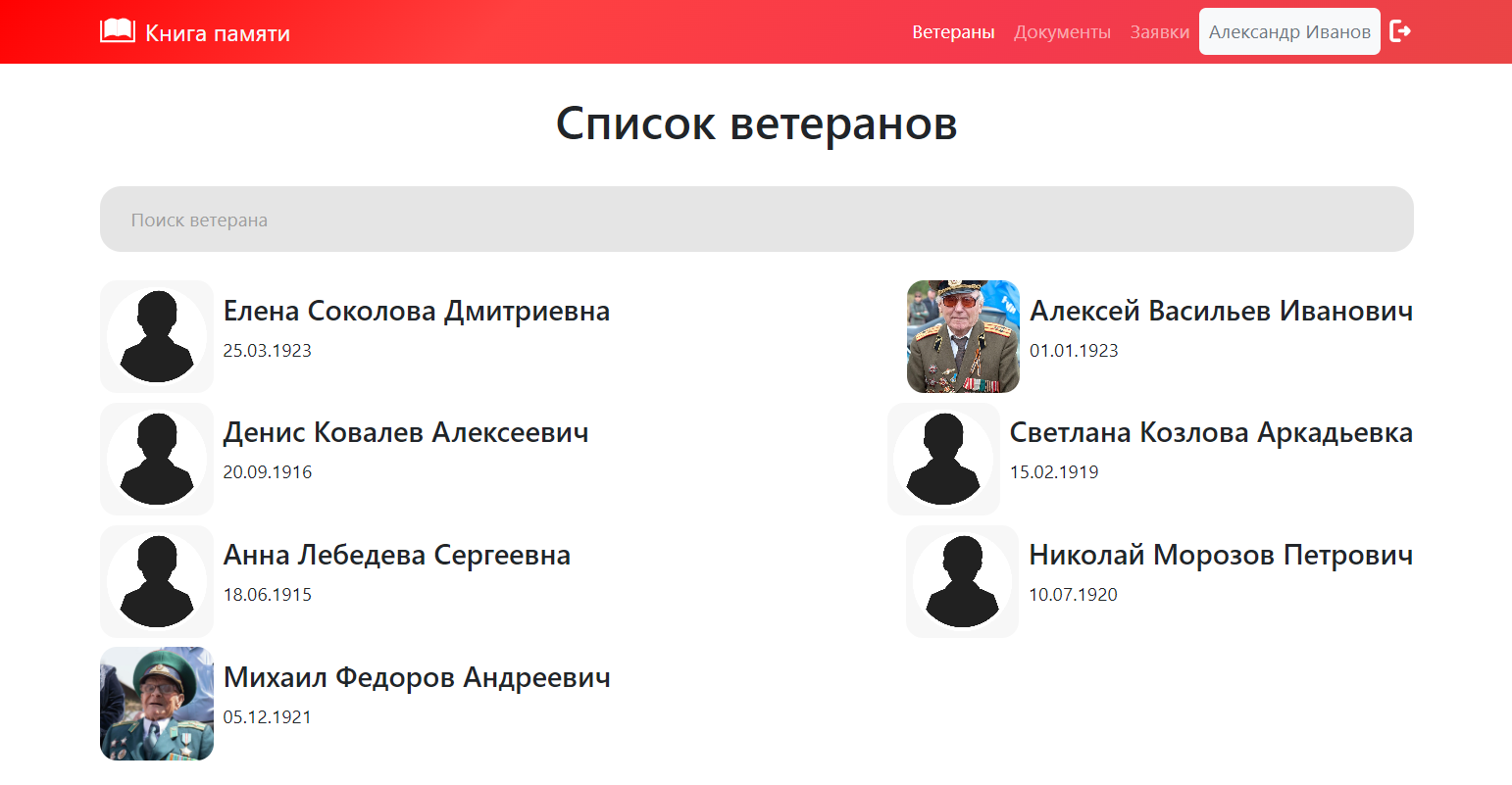
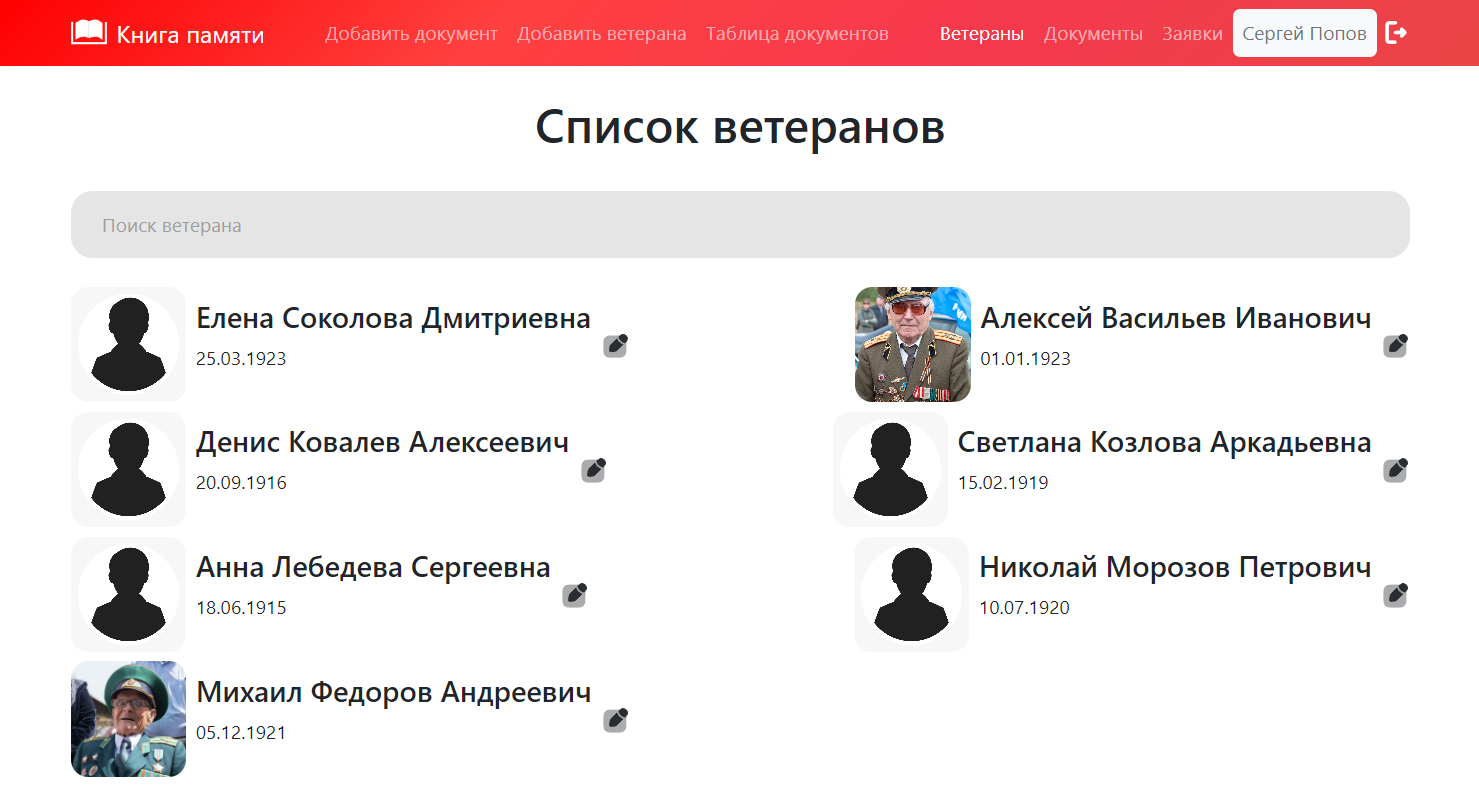


Рисунок 21 - Страница со списком ветеранов (участник)

 Рисунок 22 - Страница со списком ветеранов (архивист)

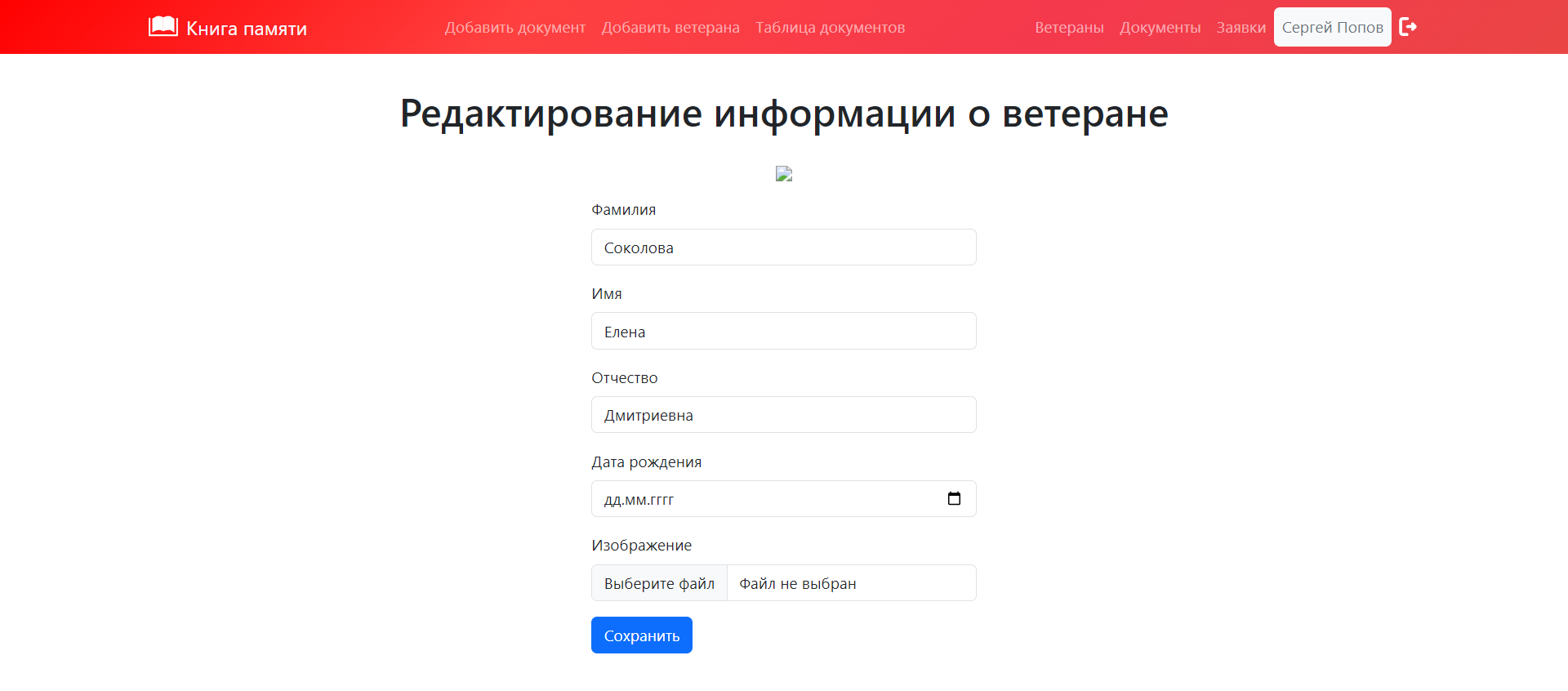


Рисунок 23 - Страница редактирования ветерана

Страница с добавлением ветерана (рис. 24) позволяет участникам проекта заполнять информацию о ветеранах, которых пока отсутствуют в приложении. В форме необходимо заполнить ФИО участника ВОВ, указать его дату рождения и загрузить фотографию при наличии. Если все данные заполнены корректно, то по нажатию на кнопку «Сохранить» информация появится в системе. После успешного добавления ветеран отобразиться в выпадающем списке в конструкторе заявки.

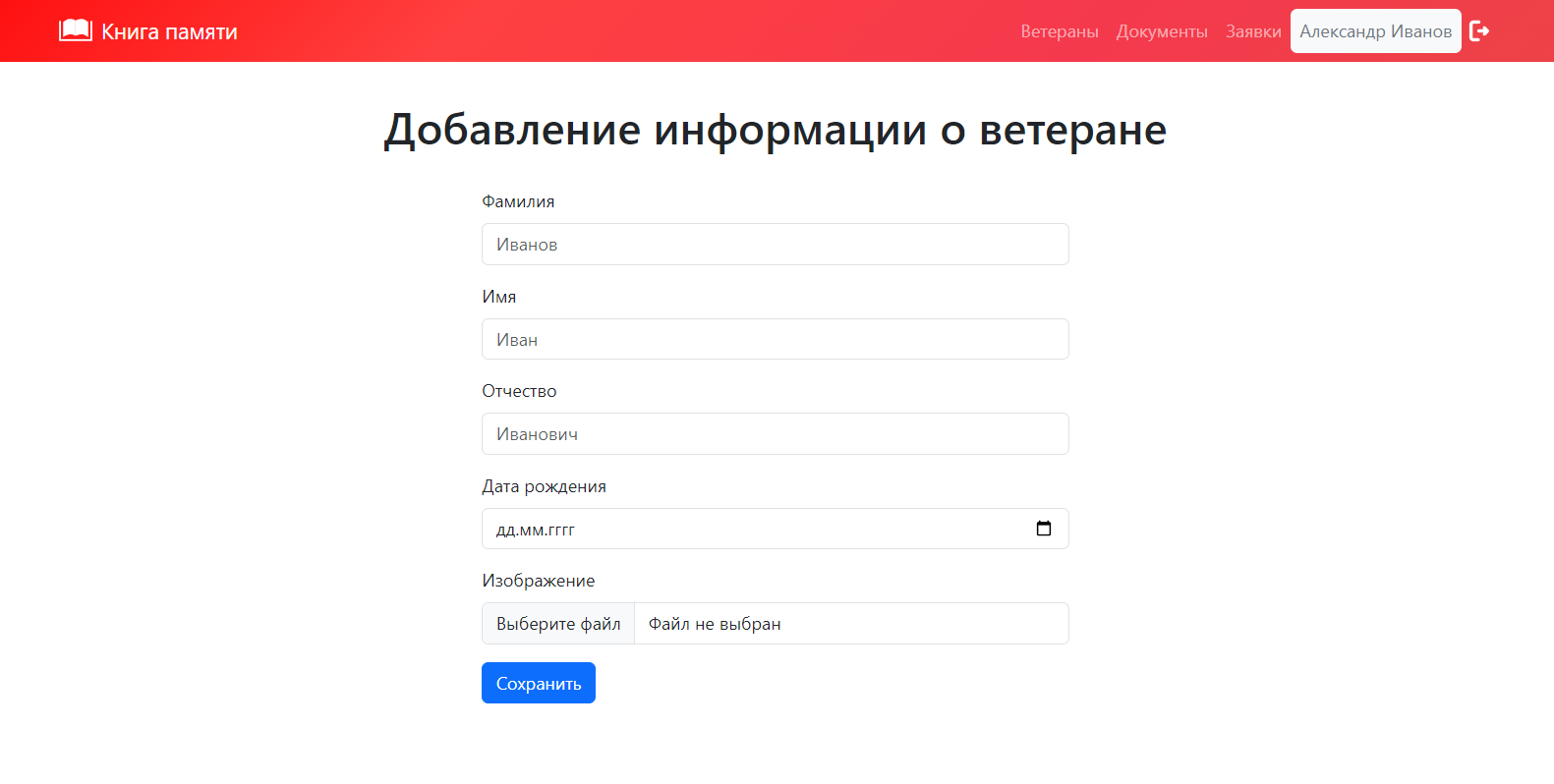


Рисунок 24 - Страница добавления ветерана

# **ЗАКЛЮЧЕНИЕ**

В ходе работы были достигнуты следующие результаты:

1. Был разработан дизайн приложения с помощью набора стилей CSS и HTML тегов.
2. База данных была создана и расположена в docker контейнере.
3. Был создан веб-сервис.
4. Разработан интерфейс гостя с использованием технологии React Framework и подключен к веб-сервису.
5. Приложение интерфейса было развернуто на сервисе Github Pages по ссылке https://azat-bilalov.github.io/book-of-memory-frontend.
6. В веб-сервис добавлена авторизация с использованием JWT токенов, а методы задокументированы через Swagger.
7. Был реализован интерфейс участника проекта. Доступ к нему имеют только авторизированные пользователи.
8. Выделенный сервис был разработан и развернут в отдельном docker кластере.
9. Реализован интерфейс архивиста для подтверждения заявок и создания новых консультаций
10. Было реализовано десктопное приложение на Tauri, повторяющее интерфейс веб-приложения на React.
11. Подготовлен набор документации, включающий РПЗ, ТЗ и набор UML диаграмм.
12. Исходный код проекта доступен в GitHub:

https://github.com/Azat-Bilalov/book-of-memory-server

# **СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ**

* 1. Сохранение культурного наследия исторических поселений в России: история и современные проблемы [Электронный ресурс] // Cyberleninka. URL: https://cyberleninka.ru/article/n/sohranenie-kulturnogo-naslediya-istoricheskih-poseleniy-v-rossii-istoriya-i-sovremennye-problemy (дата обращения 29.12.2023)
  2. Мемориализация участников в Великой Отечественной войны в книгах памяти [Электронный ресурс] // Cyberleninka. URL: https://cyberleninka.ru/article/n/memorializatsiya-uchastnikov-velikoy-otechestvennoy-voyny-v-knigah-pamyati-rayonov-i-sel-kalmykii (дата обращения 29.12.2023).
  3. JavaScript полное руководство [Книга] // Д.Флэнаган. (дата обращения 29.12.2023).
  4. Полное практическое руководство по Docker [Электронный ресурс] // Habr. URL: https://habr.com/ru/articles/310460/ (дата обращения: 29.12.2023).
  5. Руководство по React [Электронный ресурс] // Habr. URL: https://habr.com/ru/companies/ruvds/articles/428077/ (дата обращения: 29.12.2023).
  6. Quick Start – React [Электронный ресурс] // React. URL: https://react.dev/learn (дата обращения: 29.12.2023).
  7. Документация PostgreSQL [Электронный ресурс] // PostgreSQL. URL: https://postgrespro.ru/docs/postgresql (дата обращения: 29.12.2023).
  8. Документация Redux [Электронный ресурс] // Redux URL: <https://reactdev.ru/libs/redux/> (дата обращение 29.12.2023)
  9. Документация MinIO [Электронный ресурс] // MinIO URL: <https://min.io/docs/minio/linux/index.html> (дата обращения: 29.12.2023)

**Приложение. Техническое задание**

|  |  |
| --- | --- |
| zgu | **Министерство науки и высшего образования Российской Федерации**  **Федеральное государственное бюджетное образовательное учреждение**  **высшего образования**  **«Заполярный государственный университет**  **им. Н.М. Федоровского**  **(ЗГУ им. Н.М. Федоровского)** |

**­**

**Факультет «Электроэнергетики, экономики и управления»**

**Кафедра «Информационные системы и технологии»**

Дисциплина «Разработка Web-приложений»

Техническое задание

Тема: «Сервис привязки участников ВОВ к историческим документам»

Студент: Иванов И.И.

Группа: ИС-21

Преподаватель: к.э.н.,

Доцент Беляев И.С.

2024г.

1. **Цель работы**

Реализовать систему привязки участников ВОВ к архивным документам, включающую в себя веб-сервис, веб-приложение, десктопное приложение и выделенный сервис проверки подлинности документов.

1. **Назначение**

Система разработана для участников и архивистов архивной платформы Великой Отечественной войны. В данной платформе предусмотрен ограниченный доступ к архивным документам участников событий. Для получения разрешения на доступ к определенному документу участник обязан подать заявку через автоматизированную систему, указав выбранный документ. Система осуществляет создание, учет и отслеживание статуса заявок. Участники могут добавлять новые документы в архив, которые позже будут подтверждаться выделенным сервисом проверки подлинности. Архивисты имеют возможность принимать или отклонять заявки, а также вносить изменения в уже существующие документы и добавлять новые в архив.

1. **Задачи**:
   1. Разработать дизайн приложения.
   2. Создать базу данных в PostgreSQL.
   3. Создать веб-сервис на технологии Golang 1.20.
   4. Реализовать интерфейс гостя на технологии React.
   5. Развернуть веб-приложение React на Github Pages.
   6. Добавить авторизацию и аутентификацию в веб-сервис.
   7. Реализовать интерфейс гостя в React.
   8. Реализовать интерфейс модератора в React.
   9. Создать десктопное приложение на Tauri.
   10. Создать выделенный сервис для проверки подлинности документов на Django.
   11. Подготовить набор документации, включающий РПЗ, ТЗ и набор диаграмм.
2. **Методы веб-сервиса:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **№** | **Метод** | **Описание** | **URL** | **Входные данные** | **Выходные данные** |
| * 1. **Домен услуг (документов)** | | | | | |
| 4.1.1 | GET | Возвращает список активных документов  Общедоступный метод | /documents | Authorization header  {  "document": string  } | {  {  "document\_id": string,  "title": string,  "description": string,  "image": string  }[],  "entered\_binding\_id": string  } |
| 4.1.2 | GET | Возвращает активный документ по его UUID  Общедоступный метод | /documents/:uuid |  | {  "document\_id": string,  "title": string,  "description": string,  "image": string  } |
| 4.1.3 | POST | Создает новый документ  Доступно участнику проекта и архивисту | /documents | Authorization header  {  "title": string,  "description": string,  "image": file  } | {  "document\_id": string,  "title": string,  "description": string,  "image": string  } |
| 4.1.4 | PUT | Обновляет существующий документ по его UUID  Доступно только архивисту | /documents/:uuid | Authorization header  {  "title": string,  "description": string,  "image": file  } | {  "document\_id": string,  "title": string,  "description": string,  "image": string  } |
| 4.1.5 | DELETE | Удаляет существующий документ по его UUID  Доступно только архивисту | /documents/:uuid | Authorization header |  |
| 4.1.6 | POST | Добавляет документ к заявке по UUID  Доступно только участнику проекта | /bindings/documents/:uuid | Authorization header |  |
| 4.1.7 | DELETE | Удаляет документ из заявки по UUID заявке  Доступно только участнику проекта | /bindings/documents/:uuid | Authorization header |  |
| * 1. **Домен заявок** | | | | | |
| 4.2.1 | GET | Возвращает список заявок участника  Доступно участнику и архивисту  Архивистувозвращается список всех заявок, участнику – только его заявки | /bindings | Authorization header  {  "status": string,  "date\_from": string,  "date\_to": string  } | {  "binding\_id": string,  "status": string,  "date": string,  "veteran\_id": string,  "user\_id": string  }[] |
| 4.2.2 | GET | Возвращает заявку по ее UUID  Доступно участникам проекта и архивисту  Архивистувозвращается список всех заявок, пользователю – только его заявки | /bindings/:uuid | Authorization header | {  "binding\_id": string,  "status": string,  "date": "string",  "veteran": Veteran,  "user":  {  "first\_name": string,  "last\_name": string,  "email": string  },  "documents": Document[],  } |
| 4.2.3 | PUT | Обновляет существующую заявку по UUID  Доступно только участникам  Выполняется только в случае, если статус заявки "entered" | /bindings/:uuid | Authorization header  {  "veteran\_id": string  } | {  "binding\_id": string,  "status": string,  "date": "string",  "veteran": Veteran,  "user": UserResponse,  "documents": Document[],  } |
| 4.2.4 | PUT | Подтверждает заявку по ее UUID  Доступно только участникам  Выполняется только в случае, если статус заявки "entered" | /bindings/:uuid/submit | Authorization header | {  "binding\_id": string,  "status": string,  "date": "string",  "veteran": Veteran,  "user": UserResponse,  "documents": Document[],  } |
| 4.2.5 | PUT | Принимает или отклоняет заявку по ее UUID  Доступно только архивисту  Выполняется только в случае, если статус заявки "in\_progress" | /bindings/:uuid/accept-reject | Authorization header  {  "status": "Accept" | "Reject"  } | {  "binding\_id": string,  "status": string,  "date": "string",  "veteran": Veteran,  "user": UserResponse,  "documents": Document[],  } |
| 4.2.6 | DELETE | Удаляет заявку по ее UUID  Доступно только участникам  Выполняется только в случае, если статус заявки "entered" | /bindings/:uuid | Authorization header |  |
| 4.2.7 | PUT | Устанавливает статус заявки подлинности заявки  Вызывается внешним сервисом. | /binding/:uuid/verify | {  "binding\_id": string,  "token": string,  "verification\_result": string  } |  |
| * 1. **Домен ветеранов** | | | | | |
| 4.3.1 | GET | Возвращает список ветеранов  Общедоступный метод | /veterans | {  "veteran": string  } | {  "veteran\_id": string,  "first\_name": string,  "last\_name": string,  "patronymic": string,  "birth\_date": string,  "image\_url": string,  }[] |
| 4.3.2 | GET | Возвращает ветерана по UUID  Общедоступный метод | /veterans/:uuid |  | {  "veteran\_id": string,  "first\_name": string,  "last\_name": string,  "patronymic": string,  "birth\_date": string,  "image\_url": string,  "documents": Document[]  } |
| 4.3.3 | POST | Создаёт нового ветерана  Доступно участникам и архивистам | /veterans | {  "first\_name": string,  "last\_name": string,  "patronymic": string,  "birth\_date": string,  "image": file  } | {  "veteran\_id": string,  "first\_name": string,  "last\_name": string,  "patronymic": string,  "birth\_date": string,  "image\_url": string  } |
| 4.3.4 | PUT | Обновляет ветерана  Доступно архивисту | /veterans | {  "first\_name": string,  "last\_name": string,  "patronymic": string,  "birth\_date": string,  "image": file  } | {  "veteran\_id": string,  "first\_name": string,  "last\_name": string,  "patronymic": string,  "birth\_date": string,  "image\_url": string  } |
| 4.3.5 | DELETE | Удаляет ветерана по его UUID  Доступно только архивисту | /veterans/:uuid |  |  |
| * 1. **Домен аутентификация** | | | | | |
| 4.4.1 | POST | Регистрирует нового участника в базе данных | /auth/register | {  "email": char(100),  "name": char(100),  "password": char(100)  } | jwt\_token = {  "access\_token": string,  "expires\_in": integet,  "token\_type": string  } |
| 4.4.2 | POST | Выполняет аунтификацию | /auth/login | {  "email": char(100),  "password": char(100)  } | jwt\_token |
| 4.4.3 | POST | Выполняет выход из аккаунта  Доступна только авторизированным пользователям | /auth/logout | jwt\_token |  |
| * 1. **Методы асинхронного сервиса проверки подлинности** | | | | | |
| 4.5.1. | POST | Проверяет подлинность документов в заявке  Вызывается основным сервисом | /verification | {  "binding\_id": string  } |  |

1. **Функциональные требования:**
   1. Главное меню. На этой странице находится текст с приветствием, если участник не авторизирован, и меню со списком доступных пунктов.
      1. Доступна всем пользователям.
      2. Действия
         1. Войти – переход на страницу 5.2. Только для гостей.
         2. Зарегистрироваться – переход на страницу 5.3. Только для гостей.
         3. Открыть список документов – переход на страницу 5.4. Доступна всем пользователям.
         4. Открыть список выбранных документов – переход на страницу 5.7. Только для авторизированных пользователей.
         5. Список заявок – переход на страницу 5.6. Только для авторизированных пользователей.
         6. Создать документ – переход на страницу 5.8. Только для архивиста.
         7. Таблица документов – переход на страницу 5.9. Только для архивистов.
   2. Страница с формой авторизации.
      1. Доступна гостям
      2. Действия
         1. Войти – производит запрос (метод 4.4.2.).
         2. Зарегистрироваться – перенаправляет на страницу 5.3.
   3. Страница с формой регистрации.
      1. Доступна гостям
      2. Действия
         1. Зарегистрироваться – производит запрос (метод 4.4.1.), в котором передаются введенные данные формы.
         2. Войти - перенаправляет пользователя на страницу 5.2.
   4. Страница со списком документов. На этой странице располагается список всех документов и панель их фильтрации.
      1. Доступна всем пользователям.
      2. Выводится информация о документах в виде карточек (методы 4.1.1).
         1. Прикреплённое к документу изображение.
         2. Название документа.
         3. Краткое описание документа.
      3. Действия
         1. Отфильтровать документы по указанному названию (методы 4.1.1).
         2. Добавить документ в заявку (метод 4.1.7). Только для авторизированных пользователей.
         3. Удалить документ из заявки (метод 4.1.8). Только для авторизированных пользователей.
         4. Подробнее – перенаправляет на страницу 5.5.
   5. Страница с подробным описанием документа
      1. Доступна всем пользователям.
      2. Выводится информация о документе в построчном формате (метод 4.1.2).
         1. Прикрепленное к документу изображение.
         2. Название документа.
         3. Описание документа.
      3. Действия
         1. Редактировать – перенаправляет на страницу 5.8. Только для архивистов.
   6. Страница со списком заявок
      1. Доступна только авторизированным пользователям.
      2. Выводится информация о заявках в табличном формате (метод 4.2.1).
         1. Статус заявки.
         2. Дата создания.
         3. Дата формирования.
         4. Дата завершения.
      3. Действия
         1. Открыть заявку – перенаправляет на страницу 5.7.
         2. Установить заявке статус «отклонена» или «завершена» (метод 4.2.5). Только для архивистов.
         3. Отфильтровать список заявок по диапазону дат и/или статусу заявки (метод 4.2.1).
   7. Страница с подробным описанием заявки. Участник может изменить документы в введённой заявке. Архивист может просматривать заявки всех пользователей.
      1. Доступна только авторизированным пользователям.
      2. Выводится ветераны выпадающим списком (метод 5.3.1).
      3. Выводится информация о заявке в карточной форме. Список документов, выбранных в заявке, в виде карточек (метод 4.2.2.).
         1. Прикреплённое к документу изображение.
         2. Название документа.
         3. Описание документа.
      4. Действия
         1. Удалить документ из заявки (метод 4.1.8). Если заявка находится в статусе черновика. Только для владельца заявки.
         2. Выбрать ветерана (метод 4.2.3). Если заявка находится в статусе черновика. Только для владельца заявки.
         3. Сформировать заявку (метод 4.2.4). Если заявка находится в статусе черновика. Только для владельца заявки.
         4. Удалить заявку (метод 4.2.6). Если заявка находится в статусе черновика. Только для владельца заявки.
         5. Открыть документ – перенаправляет на страницу 5.5.
   8. Страница редактирования/создания документа. Архивист может изменить существующий или создать новый документ. Участник проекта может только добавлять новый документ.
      1. Доступна авторизированным пользователям.
      2. Выводится информация о документе в построчном формате (метод 4.1.2).
         1. Прикреплённое к документу изображение.
         2. Название документа.
         3. Описание
      3. Изменение полей документа. Можно изменять все вышеперечисленные поля документа (методы 4.1.4 и 4.1.5).
   9. Страница с таблицей документов. Предоставляет архивисту удобный способ отображения всех документов.
      1. Доступна только архивисту.
      2. Выводит информация о документах в табличном виде (метод 4.1.1).
         1. Прикреплённое к документу изображение.
         2. Название документа.
         3. Описание.
         4. Статус.
      3. Действия
         1. Удалить документ (метод 4.1.6)
         2. Редактировать документ – перенаправляет на страницу 5.8.
   10. Страница со списком ветеранов. На этой странице располагается список всех ветеранов и панель их фильтрации.
       1. Доступна всем пользователям.
       2. Выводится информация о ветеранах в виде карточек (методы 4.3.1).
          1. Фотография ветерана.
          2. ФИО ветерана.
          3. Дата рождения ветерана.
       3. Действия
          1. Отфильтровать ветеранов по указанному названию (методы 4.3.1).
          2. Удаление ветерана (метод 4.3.4). Только для архивиста.
          3. Редактирование ветерана – перенаправляет на страницу 5.11. Только для архивиста.
   11. Страница редактирования/создания ветерана. Архивист может изменить существующего или создать нового ветерана. Участник может только добавлять нового ветерана.
       1. Доступна авторизированным пользователям.
       2. Выводится информация о ветеране в построчном формате (метод 4.3.2).
          1. Фотография ветерана.
          2. ФИО ветерана.
          3. Дата рождения ветерана.
       3. Изменение полей. Можно изменять все вышеперечисленные поля ветерана (методы 4.3.3 и 4.3.4).
2. **Требования к программному обеспечению:**
   1. Серверная часть
      1. ОС: Linux (2.x.x) /Windows (7 и выше)
      2. Docker (4.23.x и выше)
      3. Веб-сервер Minio (8.4.x и выше)
      4. Веб-сервер Redis (7.2 и выше)
      5. Golang (1.21.x и выше) с фреймворком Echo (4.11.x и выше)
      6. Python 3.11.5
      7. СУБД PostgreSQL (16.0 и выше)
   2. Клиентская часть
      1. ОС: Windows/Linux
      2. Веб-браузер: Chrome (119 и выше)
3. **Требования к аппаратному обеспечению:**
   1. Серверная часть
      1. Процессор минимум 4-ядерный с частотой от 2 ГГц
      2. Оперативная память от 8 Гб
      3. Место на жестком диске от 1 Гб
   2. Клиентская часть
      1. Процессор с частотой от 2ГГц
      2. Оперативная память от 1 Гб