ЛАБОРАТОРНАЯ РАБОТА №2



Курс «Технологии разработки программного обеспечения»

Тема: Проектирование и макетирование программного продукта.

Цель: Научиться проектировать простейшие системы и составлять документацию по проектированию программного продукта.

Задание:

1. Спроектировать общую архитектуру приложения на основе спецификаций требований, составленных в лабораторной работе №1.

**Архитектура**

**Архитектурный стиль:** Многослойная архитектура (представление, бизнес-логика, уровень данных) с RESTful API для взаимодействия между клиентской и серверной частями.

Компоненты:

**Клиентская часть (Frontend):**

Модуль отображения каталога книг.

Модуль чтения книг.

Модуль авторизации и управления аккаунтом.

Модуль читательского дневника.

Модуль просмотра комментариев.

Модуль чата для книг (доступен зарегистрированным пользователям).

**Серверная часть (Backend):**

Модуль аутентификации и авторизации.

Модуль управления каталогом книг.

Модуль управления комментариями к книгам.

Модуль управления читательским дневником.

Модуль чата для книг (реализация WebSocket или аналогичная технология).

Модуль доступа к данным (работа с базой данных).

**База данных (Data Layer):**

Хранит информацию о пользователях, книгах, комментариях, записях дневника и сообщениях чата.

1. Разработать макет интерфейса программного продукта для демонстрации заказчику. Проанализировать и учесть его замечания и пожелания.

* При открытии приложения будет появляться окно загрузки с логотипом (рис 1). Светло-фиолетовый круг будет немного уменьшаться в размере и возвращаться.



Рисунок 1

* После будет появляться окно регистрации и входа (рис 2)

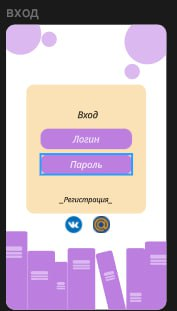
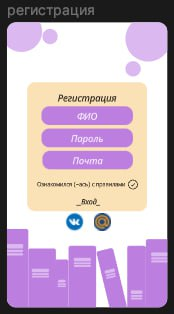


Рисунок 2

Между ними можно переключаться между ними с помощью «\_Вход\_» и «\_Регистрация\_» (рис 2). При «Регистрации», пользователь соглашается с правилами, установленными администратором, и контролируемые модераторами.

* После любой из этих махинаций, пользователь переходит на главную страницу



Рисунок 3

На ней можно сразу искать книги по названию или автору, или просто зайти в библиотеку (рис 4) в которой будет лента из самых популярных произведений этой недели (промежуток времени можно менять на «день», «месяц», «год», нажав на кружочек с гайкой в правом верхнем углу. По умолчанию стоит «неделя»). В левом верхнем углу можно было нажать на стрелку и вернуться обратно на «главный экран».

Так же тут есть кнопка на переход в личный кабинет (рис 3). Если пользователь не зарегистрирован, то всплывает предупреждающее окно (рис 5). При нажатии «да» мы возвращаемся на окна регистрации и входа (рис). При выборе «нет», всплывающее окно уходит, и пользователь просто остается на «главном экране».



Рисунок 4

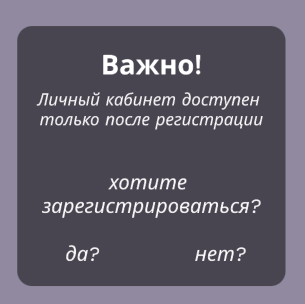


Рисунок 5

* В случае выполнения условия «регистрация», пользватель переходит в свой личный кабинет (рис 6).

Тут отображается личная информация пользователя:

1. Никнейм пользователя
2. установленное им изображение
3. статус. (слово «Статус» пропадает при его замене с системного на пользовательский.
4. Уровень (в соответствии с определенной системой, он повышается (например: за посещение приложения больше 2 дней подряд, прочтении +1 книги или активном участии в чате, комментариях) или понижается (например: при не соблюдении правил, с которыми пользователь будет соглашается при регистрации)
5. В правом верхнем углу карандаш, при нажатии, дает возможность редактировать пункты 1,2,3.
6. В левом верхнем углу стрелочка переносит обратно на «главный экран»
7. В правом нижнем углу книга с узором шестеренки называется «настройки». (Там можно редактировать: уведомления, размер шрифта и темную и светлую тему. После пару обновлений возможны добавления новых пунктов типа: выбрать цветовую палитру приложения)
8. В левом нижнем углу, книга с названием «жалобы». Там пользователь может пожать жалобу на цензуру или плагиат. (открывается после достижения Lv.10 пользователя)
9. Изображение полки с книгами, представляют собой кнопку для перехода в «читательский дневник (рис 7)



Рисунок 6

* В читательском дневнике (рис7) отображаются добавленные пользователем книги. Отображается обложка книги, ее название и оценка пользователя (10/10). Так же быстрый переход в чат произведения (рис8).

В верхнем правом углу волшебная палочка отвечает за редактирование дневника (добавление и удаление книг)

В левом верхнем углу стрелочка переносит обратно на «читательский дневник»

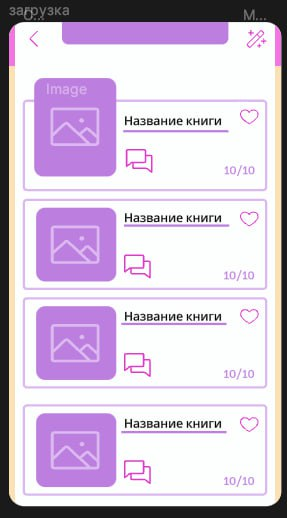


Рисунок 7

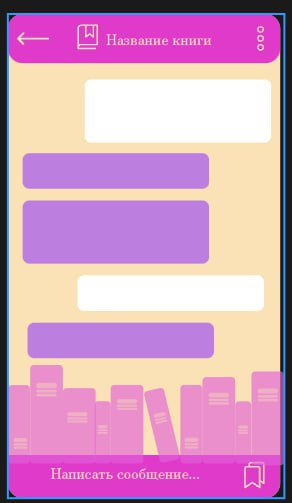


Рисунок 8

* В чате можно писать сообщения. Отвечать на сообщения других пользователей. Вставлять цитаты с помощью флажков в правом нижнем углу. (чат предназначен для обсуждения конкретного произведения)
* Нажав на любую из книг (не важно, где в библиотеке или дневнике), пользователь переходит в превью Книги (рис 9).

На превью есть:

1. Обложка книги
2. Название книги (так же на английском и оригинальном языке)
3. Издательство
4. Пролог
5. В левом верхнем углу стрелочка переносит обратно на предыдущую страницу
6. В правом верхнем углу карандашик дает возможность добавить/удалить книгу в читательском дневнике.
7. Сердечки позволяют кратко отреагировать на книгу
8. В левом нижнем углу переход в чат произведения (рис7)
9. В правом нижнем углу отображены черточки позволяющие перейти в комментарии (рис10).



Рисунок 9

* В комментариях нельзя отвечать на комментарии других пользователей. Можно читать их, писать свои и оставлять реакции в виде «лайка» и «дизлайка», означающие положительный и отрицательный факт.   
  Сверху стрелочка позволяет вернуться к «превью» книги. (рис9)

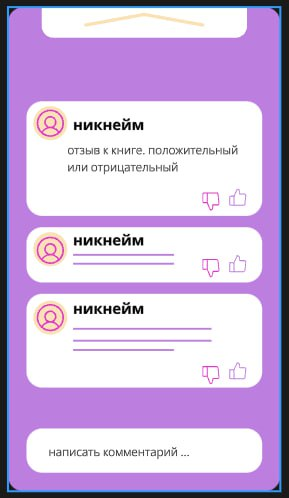


Рисунок 10

1. Оформить разделы документа SDD, относящиеся к предварительному проектированию (приложение 2.1).

**1.Введение**

* **Название приложения:** Онлайн-библиотека “Читай-ка”
* **Цель:** предоставить пользователям возможность читать книги онлайн, вести читательский дневник, участвовать в обсуждениях книг и делиться впечатлениями.
* **Целевая аудитория:** любители чтения, желающие иметь доступ к книгам онлайн и общаться с другими читателями.
* **Обзор:** Приложение состоит из клиентской части (для отображения интерфейса и взаимодействия с пользователем) и серверной части (для хранения данных и обработки запросов).

**2.Компоненты и модули**

* **2.1 Серверная часть (Backend)**
  + **Аутентификация и авторизация:**
    - Предоставляет API для регистрации, входа и выхода пользователей.
    - Генерирует и управляет токенами авторизации.
    - Проверяет права доступа к ресурсам (например, доступ к чату только для зарегистрированных пользователей).
  + **Управление каталогом книг:**
    - Предоставляет API для поиска, просмотра книг.
    - Позволяет добавлять, изменять и удалять книги (только для администраторов).
  + **Управление комментариями:**
    - Предоставляет API для получения списка комментариев к книге.
    - Позволяет добавлять новые комментарии к книге (для всех пользователей).
  + **Управление читательским дневником:**
    - Предоставляет API для добавления, просмотра, изменения и удаления записей дневника.
  + **Чат для книг:**
    - Использует WebSocket (или аналогичную технологию) для обмена сообщениями в реальном времени.
    - Предоставляет API для получения истории сообщений.
    - Ограничивает доступ к чату только для зарегистрированных пользователей.
  + **Доступ к данным:**
    - Интерфейс для взаимодействия с базой данных.
    - Выполняет операции CRUD (создание, чтение, обновление, удаление) для всех сущностей.
* **2.2 Клиентская часть (Frontend)**
  + **Каталог книг:**
    - Отображает список книг (с разбивкой на страницы, сортировкой и фильтрацией).
    - Предоставляет возможность просмотра детальной информации о книге.
    - Предоставляет возможность поиска книг.
  + **Чтение книг:**
    - Отображает текст книги.
    - Позволяет перелистывать страницы.
  + **Авторизация и управление аккаунтом:**
    - Предоставляет формы для регистрации и входа.
    - Позволяет просматривать и редактировать профиль пользователя.
  + **Читательский дневник:**
    - Отображает список записей дневника.
    - Предоставляет формы для добавления, изменения и удаления записей.
  + **Просмотр комментариев:**
    - Отображает список комментариев к книге.
    - Позволяет добавлять новые комментарии.
  + **Чат для книг:**
    - Отображает чат для обсуждения книги.
    - Позволяет отправлять сообщения в реальном времени.

**3.База данных**

* **Схема (пример):**
  + **Пользователи:** (идентификатор пользователя, имя пользователя, адрес электронной почты, хэш пароля, дата регистрации)
  + **Книги:** (идентификатор книги, название, автор, жанр, год публикации, изображение обложки, содержание)
  + **Комментарии:** (идентификатор комментария, идентификатор пользователя, идентификатор книги, содержимое, дата)
  + **ReadingLog:** (logEntryId, userId, bookId, дата, текст, рейтинг)
  + **Сообщения в чате:** (идентификатор сообщения, идентификатор книги, идентификатор пользователя, сообщение, дата)

**4.Безопасность**

* Использование HTTPS для шифрования трафика.
* Хранение паролей в виде хэшей.
* Проверка прав доступа к ресурсам (на основе токенов авторизации).

**5.Нефункциональные требования**

* **Производительность:** Быстрый отклик, обработка большого количества запросов.
* **Масштабируемость:** возможность добавления новых пользователей и контента без потери производительности.
* **Надежность:** Бесперебойная работа приложения.

1. Программно реализовать первую версию спроектированной системы (создать программные модули и закодировать основные классы / структуры и связи между ними).



1. Предварительное проектирование vs. Детальное проектирование:

• Предварительное проектирование: Это начальная стадия проектирования, где определяются основные характеристики и требования к проекту. Цель - провести предварительный анализ и определить основные параметры проекта.

• Детальное проектирование: На этом этапе разрабатываются подробные чертежи и спецификации, необходимые для реализации проекта. Включает в себя детализированные планы, схемы и расчеты.

2.

Архитектурные системные паттерны:

• Многослойная архитектура: Организует систему в слои, каждый из которых отвечает за определенные функции (представление, бизнес-логика, доступ к данным).

• Порты и адаптеры: Разделяет бизнес-логику и инфраструктурные компоненты, обеспечивая независимость бизнес-логики от технических аспектов.

• CQRS (Command Query Responsibility Segregation): Разделяет операции чтения и записи данных, что позволяет оптимизировать производительность и масштабируемость системы.

3.

Паттерны управления:

• Фабричный метод: Создает объекты через интерфейс, позволяя подклассам изменять тип создаваемого объекта.

• Одиночка (Singleton): Гарантирует, что у класса есть только один экземпляр, и предоставляет глобальную точку доступа к нему.

• Стратегия (Strategy): Определяет семейство алгоритмов, инкапсулирует их и делает их взаимозаменяемыми.

4.

Связность модуля:

• Связность: Степень зависимости между элементами внутри модуля. Чем выше связность, тем лучше модуль выполняет одну задачу.

• Типы связности:

• Функциональная: Все элементы модуля работают вместе для выполнения одной задачи (самая сильная связность).

• Последовательная: Элементы модуля выполняют задачи последовательно.

• Коммуникационная: Элементы модуля используют одни и те же данные.

• Процедурная: Элементы модуля выполняют задачи, которые должны быть выполнены в определенном порядке.

• Временная: Элементы модуля выполняют задачи, которые должны быть выполнены в одно и то же время.

• Логическая: Элементы модуля выполняют логически связанные задачи.

• Случайная: Элементы модуля не связаны друг с другом (самая слабая связность).

5.

Сцепление модулей:

• Сцепление: Степень зависимости между модулями. Чем слабее сцепление, тем лучше.

• Типы сцепления:

• Сцепление по данным: Модули обмениваются данными через параметры.

• Сцепление по управлению: Один модуль управляет поведением другого.

• Сцепление по общим данным: Модули используют общие данные.

• Сцепление по содержимому: Один модуль напрямую использует содержимое другого модуля (самое сильное сцепление).

6.

Фундаментальные паттерны проектирования:

• Делегирование: Объект передает задачу другому объекту, который может её выполнить.

• Неизменяемый объект (Immutable Object): Объект, который не может быть изменен после создания.

• Интерфейс (Interface): Определяет набор методов, которые должны быть реализованы классами.

• MVC (Model-View-Controller): Разделяет приложение на три компонента: модель (данные), вид (интерфейс) и контроллер (логика).