

СОДЕРЖАНИЕ

ВВЕДЕНИЕ.....	4
1. СИСТЕМНЫЙ АНАЛИЗ И ПОСТАНОВКА ЗАДАЧИ.....	6
1.1 Анализ выбранной тематики.....	6
1.2 Исходные данные к проекту.....	8
2. ПРОЕКТИРОВАНИЕ СИСТЕМЫ.....	9
2.1 Проектирование архитектуры ПО.....	9
2.2 Структура программного, аппаратного и информационного обеспечения системы.....	13
2.3 Макетирование пользовательского интерфейса.....	18
3. РЕАЛИЗАЦИЯ СИСТЕМЫ.....	19
4. ТЕСТИРОВАНИЕ СИСТЕМЫ.....	22
ЗАКЛЮЧЕНИЕ.....	30
СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ.....	31
ПРИЛОЖЕНИЕ А СХЕМА ПРОГРАММЫ	

					КР.ПО4. 190329-04 81 00		
<i>Изм</i>	<i>Лист</i>	<i>докум №</i>	<i>Подп.</i>	<i>Дата</i>			
Разраб.		Байдук Я. А.			<i>Веб-приложение «Онлайн библиотека». Пояснительная записка</i>		
Проверил		Кочурко П. А.					
Н. контр.		Кочурко П. А.					
Утв.							
						<i>Лит</i>	<i>Лист</i>
						<i>К</i>	<i>Листов</i>
						БрГТУ	

ВВЕДЕНИЕ

В последнее время список сервисов и ресурсов, которые выросли в интернете небывалое множество. Интернет превратился из однообразных статических страничек в мощный инструмент интерактивности и общения с конечными пользователями. В связи с этим веб-приложения в настоящее время приобрели небывалую популярность, потому что они предлагают массу важных преимуществ, которые отсутствуют в обычных приложениях. Глядя на все происходящее в индустрии программного обеспечения можно увидеть очень большое развитие веб-приложений как отдельного звена во всей цепочке данного направления, и это не считая скорого прихода таких технологий как CSS3 и HTML5. С обычными приложениями все не так радужно, они конечно развиваются, но не так бурно, как веб-технологии. Большое количество компании переходят с обычных приложений на веб-приложения, именно потому что видят в них будущее, готовы пользоваться ими и перейти на них с обычных приложений, а за этим следует и капиталовложение, что дает мощный толчок технологиям. В связи с этим можно выделить небольшой перечень причин, по которым так происходит.

1. Установка веб-приложений дешевле и намного проще.

Благодаря использованию именно веб-приложений предприятия и компании могут снизить затраты на содержание ИТ отделов, которые отвечают за установку программного обеспечения и его сопровождение. В этом случае у пользователя просто-напросто лишь компьютер с браузером и соединение с интернетом или корпоративной сетью.

2. Обновление веб-приложений дешевле и намного проще.

Всегда большое значение имеет стоимость обслуживания ПО. Обновление ПО очень похоже на его установку, поэтому преимущества, которые были упомянуты имеют место и в данной ситуации. Для того чтобы совершить обновление веб-приложения, его необходимо обновить не только на сервере, и все сразу же смогут работать с новой версией.

3. Веб-приложения более универсальны и практичны для конечного пользователя.

Вам достаточно будет установить веб-приложение на сервер, работающей под любой современной ОС, и вы сможете пользоваться им через интернет на любом компьютере или мобильном устройстве, работающем под управлением MacOS,

					КР.ПО4190329-04 81 00	Лист
Изм	Лист	№ докум.	Подп.	Дата		4

Windows, Linux или какой-либо другой ОС. Если приложения сделаны качественно то они будут работать одинаково хорошо в любом браузере, будь то Mozilla Firefox, Opera, Google Chrome, Internet Explorer или Safari.

4. Веб-приложения облегчают организацию хранения данных.

Если есть необходимость обращаться к одним и тем же данным из разных мест, то намного проще организовать их хранение в одном месте, вместо того чтобы разбрасывать по разным базам данных. Благодаря этому отпадает необходимость синхронизации и повысится степень их защищенности.

Именно по этим причинам веб-приложения завоевали огромную популярность, которая привела к такому уровню развития данной отрасли. Именно поэтому мы с вами сейчас все пользуемся всемирной паутиной интернет.

Основная цель работы – создание веб-приложения, предоставляющего возможности хранения книг, их добавления и чтения.

					КР.ПО4190329-04 81 00	Лист
Изм	Лист	№ докум.	Подп.	Дата		5

1 СИСТЕМНЫЙ АНАЛИЗ И ПОСТАНОВКА ЗАДАЧИ

1.1 Анализ выбранной тематики

Электронная библиотека — упорядоченная коллекция разнородных электронных документов (в том числе книг, журналов), снабжённых средствами навигации и поиска. Может быть веб-сайтом, где постепенно накапливаются различные тексты (чаще литературные, но также научные и любые другие, вплоть до компьютерных программ) и медиафайлы, каждый из которых самодостаточен и в любой момент может быть востребован читателем. Электронные библиотеки могут быть универсальными, стремящимися к наиболее широкому выбору материала (как Библиотека Максима Мошкова или Либрусек), и более специализированными, как Фундаментальная электронная библиотека или проект Сетевая Словесность, нацеленный на соби́рание авторов и типов текста, наиболее ярко заявляющих о себе именно в Интернете.

Электронные библиотеки следует отличать от смежных структурных типов сайта, особенно литературного. В отличие от литературного журнала, родившегося как тип печатного издания, но успешно и без принципиальных изменений структуры перебравшегося в Интернет, электронная библиотека не подразделяется на выпуски и обновляется перманентно по мере появления новых материалов. В отличие от сайта со свободной публикацией, электронная библиотека, как правило, подбирается координатором проекта по своему усмотрению и, что гораздо более важно, не предусматривает создания вокруг публикуемых текстов коммуникативной среды. При этом в практике отдельных Интернет-проектов могут возникать и гибридные формы, и промежуточные решения: так, открытие в электронной библиотеке Сетевая Словесность гостевых книг для каждого публикуемого автора в известной степени вносит в проект элемент формирования коммуникативной среды, состоящей из авторов и читателей, что для электронных библиотек вообще нехарактерно.

Достоинства электронных библиотек:

1. Цифровая литература практически вечная — она не стареет и не рвётся.
2. Электронные библиотеки позволяют пользоваться литературой разных библиотек по всему миру.
3. Большинство интересных книг можно скачать абсолютно бесплатно, а если и нужно что-либо платить, то намного меньше, чем за печатное издание.

4. Чтобы найти интересующее издание не нужно открывать каждый сайт, а можно использовать для этого поисковик.

5. Любимые книги в электронном формате можно хранить на флэшке и носить с собой, а для большого количества печатных книг желательно иметь большой книжный шкаф.

6. Читатели могут отыскать в электронных библиотеках редкую литературу или документы.

7. Для удобного чтения книг для пользователей были разработаны программы – закладки по тексту, быстрый поиск, переход по страницам.

8. Даже если читатель не знает иностранного языка, любое издание в электронном формате можно перевести.

Количество пользователей Интернета становится всё больше и больше с каждым годом, также как и посетителей электронных библиотек. В последнее время люди не тратят своё время на поиски желаемой литературы в книжных магазинах за свои деньги.

Недостатки электронных библиотек:

1. Иногда электронные библиотеки нарушают права авторов книг :)
2. С маленького экрана электронного устройства не очень удобно читать книгу на протяжении длительного времени.
3. Электронные устройства (электронные книги, планшеты, смартфоны и др.) имеют большую стоимость.
4. Читателю следует разбираться в форматах, которые имеют электронные издания.
5. Читать обычную книгу намного приятнее, чем её цифровой аналог, так как можно ощутить запах книги, листать её.

Но стоит заметить и неоспоримые вещи – любая книга в электронной библиотеке доступна большому количеству читателей, а также каждому пользователю доступны любые из многих книг.

1.2 Исходные данные к проекту

Необходимо провести анализ предметной области, найти и проанализировать аналогичное программное обеспечение, решающее схожие задачи, из представленных на рынке. На основании данного анализа обосновать целесообразность разработки и поставить задачу.

Программный продукт должен выполнять следующие функции:

1. Регистрация и авторизация пользователя;
2. Добавление книг в библиотеки;
3. Чтение книг в режиме светлая/тёмная тема;

Разработать архитектуру системы на уровне подсистем и модулей, спроектировать структуру программного, аппаратного, информационного обеспечения системы. Произвести логическое проектирование базы данных, макетирование пользовательского интерфейса на основе шаблонов пользовательского опыта. Описать реализацию спроектированной системы: физическое проектирование баз данных, имплементация модулей на уровне классов и методов выбранного ЯП, экранные формы как реализация представленных ранее эскизов. Описать процесс испытания системы как набор приёмочных и прочих тестов. Также необходимо создать публичный репозиторий на github с исходным кодом.

					КР.ПО4190329-04 81 00	Лист
Изм	Лист	№ докум.	Подп.	Дата		8

2 ПРОЕКТИРОВАНИЕ СИСТЕМЫ

2.1 Проектирование архитектуры ПО

Для пользователей, которые будут пользоваться данным приложением, будет существовать несколько уровней привилегий (по возрастанию):

- **Гость**

Имеет возможность просмотреть главную страницу а также возможность зарегистрироваться или авторизоваться

- **Пользователь**

Имеет возможности, которые имеет гость, а также возможность читать и оценивать книги.

- **Администратор**

Имеет возможности пользователя, а также возможность добавления своих книг в библиотеку.

Разработаем общие алгоритмы, которые необходимы для выполнения курсовой работы.

Алгоритм регистрации пользователя

Исходные данные: данные, которые ввёл пользователь в форму (логин, пароль, фамилия, имя).

Алгоритм:

1. Если длина почты будет меньше 3 или больше 20 символов – подсвечивать ошибку.
2. Если длина пароля будет меньше 6 – подсвечивать ошибку.
3. Добавление информации в базу данных с помощью SQL-запроса. По умолчанию, новый пользователь будет иметь наименьший уровень привилегий.
4. Переадресация на страницу авторизации.

Выходные данные: нет

Алгоритм авторизации пользователя

Исходные данные: данные, которые ввёл пользователь в форму (логин, пароль).

Алгоритм:

1. Поиск информации в базе данных с помощью SQL-запроса.
2. Если логин или пароль не найден в базе данных – перенаправить пользователя на страницу с ошибкой. Иначе, перенаправить пользователя на главную страницу.
3. Переадресация на главную страницу.

Выходные данные: установка в сессию данных о пользователе.

Алгоритм изменения фамилии и имени

Исходные данные: данные, которые ввёл пользователь в форму (фамилия, имя).

Алгоритм:

1. Поиск информации в базе данных с помощью SQL-запроса.
2. Изменение фамилии и имени на новые значения.
3. Переадресация на страницу профиля.

Выходные данные: нет

Алгоритм изменения пароля

Исходные данные: данные, которые ввёл пользователь в форму (старый и новый пароль).

Алгоритм:

1. Поиск информации в базе данных с помощью SQL-запроса.
2. Если старый и новый пароль совпадают, то пароль меняется, иначе ошибка
3. Переадресация на страницу профиля.

Выходные данные: нет

Алгоритм выхода из профиля

Исходные данные: нет.

Алгоритм:

1. Активация кнопки «Выход из профиля».
2. Очищение сессии.

3. Переадресация на главную страницу.

Выходные данные: нет

Алгоритм удаления профиля

Исходные данные: нет.

Алгоритм:

1. Активация кнопки «Удалить профиль профиля».
2. Поиск информации в базе данных с помощью SQL-запроса.
3. Очищение сессии.
4. Переадресация на главную страницу.

Выходные данные: нет

Алгоритм вывода всех книги

Исходные данные: нет.

Алгоритм:

1. Цикл перебора всех книг.
2. Вывод книг.

Выходные данные: список статей.

Алгоритм содержания книги

Исходные данные: нет.

Алгоритм:

1. Поиск книги по его id в базе данных.
2. Вывод содержимого книги, находящиеся в пути pkg/books/text.

Выходные данные: содержимое книги.

Алгоритм создание книги (доступен только для администратора)

Исходные данные: книга.

Алгоритм:

1. Ввод названия книги, его автора, путь, где лежит книга на локальном компьютере и имя файла.
2. Добавление информации в базу данных с помощью SQL-запроса.
3. Если книга по заданному пути не найдена, то вывести ошибку.

4. Переадресация на станицу с книгами.

Выходные данные: нет.

Алгоритм добавления оценки

Исходные данные: оценка пользователя.

Алгоритм:

1. Поиск книги по его id в базе данных.
2. Добавление пользовательской оценки от 0 до 5.
3. Расчёт новой средней оценки на основе оценки пользователя.
4. Переадресация на станицу новости.

Выходные данные: нет.

2.2 Структура программного, аппаратного и информационного обеспечения системы

Для разработки Frontend будет использоваться язык разметки HTML, язык стилей CSS, а также скриптовой язык JavaScript.

HTML – это язык разметки, который используется для отображения веб-страниц. Для создания HTML-сайта потребуется нанять профессионального разработчика, который воспользуется HTML, CSS, JavaScript и другими технологиями для построения веб-ресурса.

Весь контент HTML-сайты хранят в статичных файлах, в то время как системы управления, в том числе WordPress, используют для хранения базы данных. То есть сайты на чистом HTML-коде состоят из отдельных страниц, которые существуют реально.

Преимущества сайтов на чистом HTML-коде

- меньший вес;
- экономичный расход ресурсов сервера;
- не требуется обновление движка или отдельных модулей;
- практически неуязвимы к взлому;
- упрощенная система создания бэкапа;
- высокий уровень безопасности данных;
- высокая стабильность (если ошибка допущена на одной странице, она не может затронуть работу всего сайта).

CSS (англ. Cascading Style Sheets, каскадные таблицы стилей) — это простой язык дизайна, предназначенный для упрощения процесса презентации веб-страниц.

CSS обрабатывает внешний вид веб-страницы. Используя CSS, вы можете контролировать цвет текста, стиль шрифтов, расстояние между параграфами, размеры и расположение колонок, используемые фоновые изображения и цвета, макеты дизайна, варианты отображения на разных устройствах и размерах экрана. А также множество других эффектов.

Преимущества CSS:

CSS экономит время. Вы можете написать CSS один раз, а затем использовать одну и ту же таблицу на нескольких HTML-страницах. Вы можете определить стиль для каждого HTML-элемента и применить его ко многим веб-страницам.

Страницы загружаются быстрее. Если вы используете CSS, вам не нужно каждый раз писать атрибуты HTML-тегов. Просто напишите одно CSS правило для тега и примените его ко всем вхождениям этого тега. Таким образом, меньшее количество кода означает более быстрое время загрузки.

Простота обслуживания. Чтобы внести глобальные изменения, просто измените стиль, и все элементы на всех веб-страницах будут обновляться автоматически.

Улучшенные стили для HTML. CSS имеет гораздо более широкий набор атрибутов, чем HTML, поэтому вы можете сделать гораздо лучший вид своей HTML-страницы по сравнению с атрибутами HTML.

Совместимость нескольких устройств. Таблицы стилей позволяют оптимизировать контент для более чем одного типа устройств. Используя один и тот же HTML-документ, можно представить различные версии веб-сайта для карманных устройств, таких как PDA и сотовые телефоны, или для печати.

Глобальные веб-стандарты. Теперь атрибуты HTML устарели, и рекомендуется использовать CSS. Поэтому неплохо было бы начать использовать CSS во всех HTML-страницах, чтобы сделать их совместимыми с будущими браузерами.

Оффлайн-просмотр. Веб-приложения могут хранить CSS локально с помощью оффлайн кэша. Используя это, мы можем просматривать сайты находясь оффлайн. Кэш также обеспечивает быструю загрузку и лучшую общую производительность веб-сайта.

Независимость от платформы. Скрипт обеспечивает независимость от платформы и поддерживает новейшие браузеры.

JavaScript – язык программирования, являющийся прототипно-ориентированным. Он отражает язык ECMAScript, чьим прототипом изначально и являлся. Первая вариация появилась ещё в 1995 году и с тех пор постоянно совершенствовалась, пока не пришла к нынешнему виду.

Чаще этот язык используется в разработке приложений и браузерах с целью придания им интерактивности и «живости».

Преимущества JavaScript

1. Ни один современный браузер не обходится без поддержки JavaScript.
2. С использованием написанных на JavaScript плагинов и скриптов справится даже не специалист.
3. Полезные функциональные настройки.
4. Постоянно совершенствующийся язык – сейчас разрабатывается бета-вариация проекта, JavaScript2.
5. Взаимодействие с приложением может осуществляться даже через текстовые редакторы – Microsoft Office и Open Office.
6. Перспектива использования языка в процессе обучения программированию и информатике.

Для Backend-разработки буду использовать язык GoLang. GoLang — это многопоточный компилируемый язык (интерпретатор в экосистеме также предусмотрен, но на практике необходимости в нем нет: компиляция происходит мгновенно). По производительности Go почти не уступает C++ и в десятки раз превосходит скриптовые языки — такие, как JavaScript, Python, Ruby, PHP. При этом он имеет очень простой и понятный синтаксис. Код на GoLang пишется легко и приятно, чужой читается быстро и без проблем. А освоить этот язык в состоянии даже школьники.

Преимущества языка Go:

- *Высокая производительность.*
- *Простой синтаксис.*
- *Строгая статическая типизация.* У каждой переменной — свой неизменяемый тип. Если вы изначально определили ее как целое число, значит, в течение всей программы она может быть только целым числом и ничем иным. Это свойство работает на простоту кода, делает его легко читаемым, а, кроме того, сводит к минимуму риск ошибок из-за невнимательности.
- *Низкая требовательность к памяти.* В GoLang есть «сборщик мусора» — периодически те объекты, потребность в которых уже не возникнет, удаляются. Таким образом происходит автоматическая очистка памяти.

- *Большая библиотека.* В ней есть все, что необходимо для разработки. Можно также использовать библиотеки других языков — С и С++.
- *Простота параллельных вычислений.* Язык идеально подходит для создания программ, рассчитанных на многоядерные процессоры. Многопоточность, реализуемая через так называемые горутины (go routines), которые взаимодействуют посредством каналов, позволяет вызывать несколько функций практически одновременно. Эта возможность очень актуальна при создании больших и сложных программ. Средства параллельного программирования экономят время разработчика и помогают равномерно распределять ресурсы процессора.

Visual Studio Code (VS Code) — редактор исходного кода, разработанный Microsoft для Windows, Linux и macOS. Позиционируется как «лёгкий» редактор кода для кроссплатформенной разработки веб- и облачных приложений. Включает в себя отладчик, инструменты для работы с Git, подсветку синтаксиса, IntelliSense и средства для рефакторинга. Имеет широкие возможности для кастомизации: пользовательские темы, сочетания клавиш и файлы конфигурации. Распространяется бесплатно, разрабатывается как программное обеспечение с открытым исходным кодом, но готовые сборки распространяются под проприетарной лицензией.

Visual Studio Code основан на Electron и реализуется через веб-редактор Monaco, разработанный для Visual Studio Online.

Преимущества Visual Studio Code

- множество настроек (как всей программы, так и интерфейса);
- расширяемая библиотека дополнений и готовых решений;
- мультифункциональность (редактор поддерживает почти все языки, используемые для создания приложений);
- простота и гибкость.

Также я буду использовать базу данных MySQL. MySQL является наиболее приспособленной для применения в среде web СУБД (системой управления базами данных). Не секрет, что для исполнения приложений клиента на большинстве хостинг-площадок провайдеры предоставляют небольшое количество ресурсов (как вычислительных, так и дисковых). Поэтому для данного применения необходима высокоэффективная СУБД, обладающая при этом высокой надежностью (большинство web-приложений и сайтов должны работать в режиме 24/7).

По всем этим причинам MySQL стала незыблемым стандартом в области СУБД для web, а теперь в ней развиваются возможности для использования ее в любых критичных бизнес-приложениях, то есть конкурирует на равных с такими СУБД таких производителей, как Oracle, IBM, Microsoft и Sybase.

Основные преимущества MySQL:

- многопоточность, поддержка нескольких одновременных запросов;
- оптимизация связей с присоединением многих данных за один проход;
- записи фиксированной и переменной длины;
- ODBC драйвер;
- гибкая система привилегий и паролей;
- гибкая поддержка форматов чисел, строк переменной длины и меток времени;
- интерфейс с языками C и Perl, PHP;
- быстрая работа, масштабируемость;
- совместимость с ANSI SQL;
- бесплатна в большинстве случаев;
- хорошая поддержка со стороны провайдеров услуг хостинга;
- быстрая поддержка транзакций через механизм InnoDB.

В нашем случае, база данных будет храниться на локальном сервере OpenServer.

Что касается аппаратного обеспечения, то курсовой будет выполнен на ПК, который имеет процессор Intel Core i5-8300H CPU с частотой 2.3 ГГц, ОЗУ 8 ГБ, а также тип операционной системы: 64-разрядная ОС Windows10.

2.3 Макетирование пользовательского интерфейса

Интерфейс проекта будет выглядеть следующим образом (header на всех страницах будет одинаковым, отличается будет только содержимое – тело страницы):

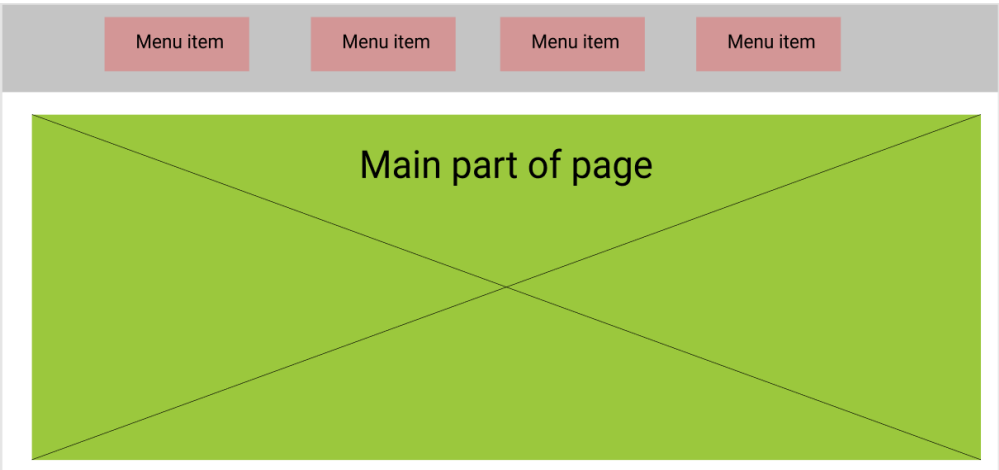


Рисунок 2.1 – Макет внешнего вида сайта

3 РЕАЛИЗАЦИЯ СИСТЕМЫ

Данная система была реализована в 64-разрядная операционная системе Windows 10 на ноутбуке фирмы hp с оперативной памятью 8Гб, процессором Intel(R) Core(TM) i5-8265U CPU @ 1.60GHz 1.80 GHz. Для разработки системы были выбраны такие инструменты: язык программирования Go для создания backend API сервера и HTML, CSS и JavaScript для создания интерфейса приложения.

Структура каталогов Go (см. Рисунок 3.1):

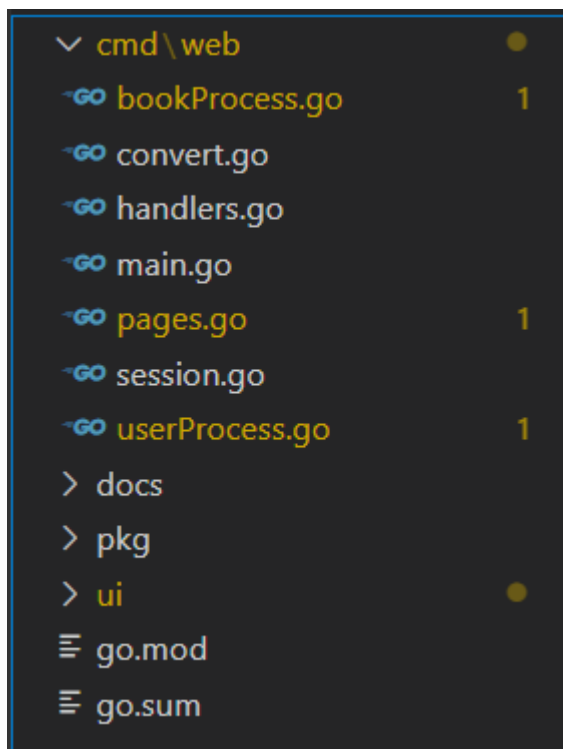


Рисунок 3.1 - структура каталогов сервера

Домашний каталог:

В домашнем каталоге, помимо файлов, которые созданы git, находятся файл go.mod.

Файл go.mod - это корень управления зависимостями в GoLang. Все модули, которые необходимы или будут использоваться в проекте, хранятся в файле go.mod.

Для всех пакетов, которые мы собираемся импортировать / использовать в нашем проекте, будет создана запись этих модулей в go.mod. Наличие файла go.mod экономит усилия по запуску команды go get для каждого зависимого модуля для успешного запуска проекта.

Также, в домашнем каталоге содержится 4 папки: cmd, docs, pkg, ui.

Каталог cmd:

Папка cmd содержит папку web. Это означает, что у нас веб-приложение. Здесь хранится модуль main.go, который является входной точкой в приложение.

Каталог pkg:

Папка pkg содержит вспомогательный код, не зависящий от приложения в проекте. Тут содержатся папки books, хранящий книги, которые загружает администраторов (с расширение .txt), handlers, содержащий подключение к базе данных и папки работы с таблицами, которые есть в базе данных, и models, содержащий в себе структуры для работы с базой данных.

Каталог ui:

Папка ui содержит файлы HTML-шаблона для пользовательского интерфейса, используемые веб-приложением. В частности, папка ui/html будет содержать HTML-шаблоны.

У данной структуры есть два больших преимущества:

- В структуре есть четкое разделение между Go файлами с кодом и файлами пользовательского интерфейса (HTML, CSS, JS) которые никак с Go не связаны. Весь написанный нами код на Go будет находиться исключительно в папках cmd и pkg. Корень проекта останется свободным для хранения ресурсов, не относящихся к Go. Это могут быть файлы пользовательского интерфейса, make-файлы и настройки модулей (включая наш файл go.mod). Данная структура упростит работу с веб-приложением, когда дело дойдет до развития и развертывания приложения в будущем;
- Структура отлично масштабируется, если вы хотите добавить в свой проект еще одно исполняемое приложение. Например, вы можете захотеть добавить CLI (Command Line Interface) для автоматизации некоторых административных задач в будущем. С такой структурой можно создать это CLI приложение в cmd/cli, и оно сможет импортировать и повторно использовать весь код, который была написан в папке pkg.

Для определения маршрутов в приложении используется пакет Go gin-gonic/gin.

Gin это высокопроизводительный микрофреймворк, который используется для создания веб-приложений и микросервисов. С ним очень удобно делать комплексную конвейерную обработку запросов из модулей - многократно используемых кусочков кода. Прописывается промежуточный слой приложения, который затем подключается в один или более обработчик запросов или в группу обработчиков.

Одно из лучших качеств Go - его встроенная библиотека net/http, позволяющая с лёгкостью создавать HTTP сервер.

В Go нет встроенной поддержки обработчика роутов на базе регулярных выражений. Вам нужно писать код для получения этого функционала. Однако, с ростом количества ваших приложений, вы будете вынуждены копировать один и тот же код везде или всё-таки создадите библиотеку.

Для работы с базой данных используется пакет Go: jinzhu/gorm/dialects/mysql (диалектом gorm является MySQL).

Gorm – это реализация доступа к базе данных на языке go. Использует библиотеку ORM (объектно-реляционное сопоставление). Благодаря этому, мы можем использовать объектно-ориентированные методы для более удобного выполнения данных в базе данных CRUD.

Для запуска сервера и обработки запросов используется пакет Go gin-gonic/gin. Чтобы запустить сервер используется функция Run(":8080"), которая в качестве параметров принимает номер порта, на котором он будет запущен, и обработчик.

В качестве стилей приложения используется Bootstrap. Bootstrap - один из самых популярных инструментов, который используется при создании сайтов и веб-приложений.

Преимущества:

- Уменьшение количества времени, затрачиваемого на разработку
- Адаптивность
- Кросс-браузерность
- Легкость в использовании и быстрота в освоении
- Понятный код
- Единство стилей

4 ТЕСТИРОВАНИЕ СИСТЕМЫ

Данная система тестировалась на 64-разрядной операционной системе Windows 10 на ноутбуке фирмы hp с оперативной памятью 8Гб, процессором Intel(R) Core(TM) i5-8265U CPU @ 1.60GHz 1.80 GHz.

В процессе тестирования была создана локальная база данных «уана» с 2-мя таблицами (см. Рисунок 4.1):













Таблица ▲	Действие	Строки ?	Тип	Сравнение	Размер	Фрагментировано
books	     	1	InnoDB	utf8mb4_unicode_ci	48.0 КиБ	-
users	     	2	InnoDB	utf8mb4_unicode_ci	32.0 КиБ	-

Рисунок 4.1 – база данных «уана»

Тест 1: «Регистрация пользователя»

Ожидаемый результат: запись нового пользователя в базу данных.

Описание: тестирование правильности добавление нового пользователя в таблицу «users» (см. рисунок 4.2, 4.3).

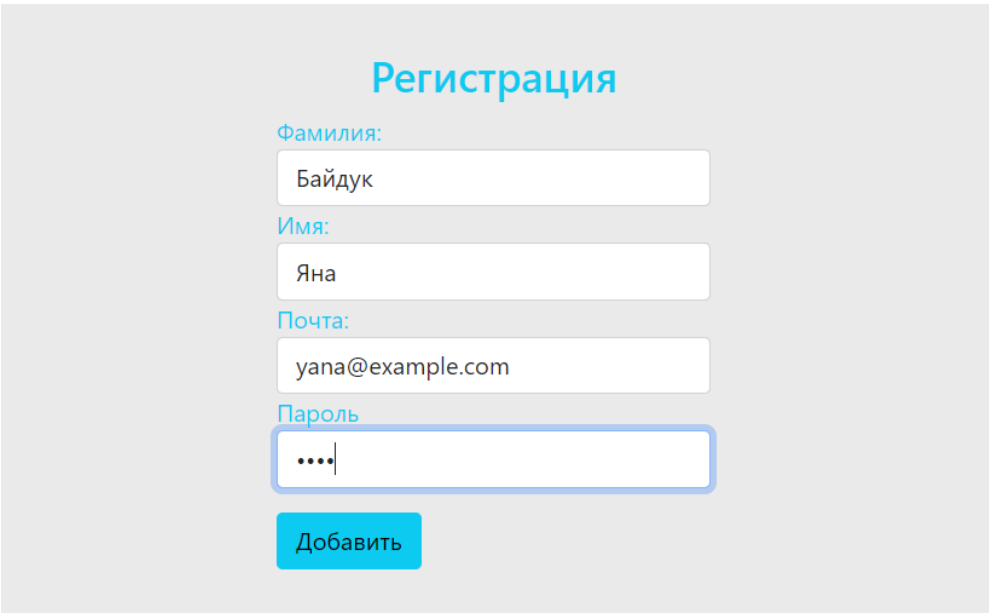


Рисунок 4.2 – форма регистрации пользователя

5	2021-12-20 15:18:56	2021-12-20 15:18:56	NULL	Байдук	Яна	yana@example.com	1111	user
---	---------------------	---------------------	------	--------	-----	------------------	------	------

Рисунок 4.3 – запись в таблице «users»

Вывод: регистрация пользователя работает корректно.

Тест 2: «Авторизация пользователя в системе»

Ожидаемый результат: проверка подлинности данных, введенные пользователем при регистрации.

Описание: тестирование правильности данных в таблице «users», введенные пользователем при регистрации (см. рисунок 4.4, 4.5, 4.6).

Рисунок 4.4 – форма авторизации пользователя



Рисунок 4.5 – меню

Рисунок 4.6 – профиль пользователя

Вывод: авторизация пользователя работает корректно.

Тест 3: «Изменение фамилии и имени пользователя»

Ожидаемый результат: проверка работы обновления фамилии и имени.

Описание: тестирование правильности обновления фамилии и имени в таблице «users», введенные пользователем при регистрации (см. рисунок 4.7, 4.8).

Рисунок 4.7 – форма редактирования фамилии и имени

Рисунок 4.8 – профиль пользователя

Вывод: обновление фамилии и имени пользователя работает корректно.

Тест 4: «Изменение пароля пользователя»

Ожидаемый результат: проверка работы обновления пароля.

Описание: тестирование правильности обновления пароля в таблице «users», введённый пользователем при регистрации (см. рисунок 4.9, 4.10).

Рисунок 4.9 – форма редактирования фамилии и имени

5	2021-12-20 15:18:56	2021-12-20 15:22:51	NULL	Иванова	Яна	yana@example.com	1234	user	
---	---------------------	---------------------	------	---------	-----	------------------	------	------	--

Рисунок 4.10 – запись в таблице «users»

Вывод: обновление пароля пользователя работает корректно.

Тест 5: «Выход из профиля»

Ожидаемый результат: корректное завершение сессии пользователя.

Описание: тестирование правильности завершение сессии пользователя (см. рисунок 4.11, 4.12).

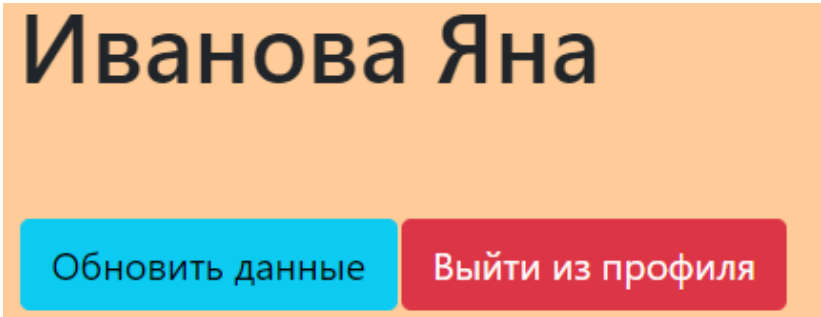


Рисунок 4.11 – профиль пользователя



Рисунок 4.12 – меню

Вывод: обновление социальных сетей пользователя работает корректно.

Тест 6: «Удаление профиля»

Ожидаемый результат: удаление пользователя из базы данных.

Описание: тестирование правильности удаление пользователя (см. рисунок 4.13, 4.14, 4.15).

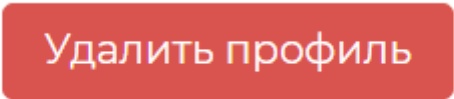


Рисунок 4.13 – профиль пользователя



Рисунок 4.14 – меню

5	2021-12-20 15:18:56	2021-12-20 15:22:51	2021-12-20 15:26:03	Иванова	Яна	yana@example.com	1234	user
---	---------------------	---------------------	---------------------	---------	-----	------------------	------	------

Рисунок 4.15 – запись в таблице «users»

Тест 7: «Вывод данных в профиле для определённого пользователя»

Ожидаемый результат: вывод определённых данных для определённых пользователей.

Описание: тестирование правильности вывода данных для определённых пользователей. (см. рисунок 4.16, 4.17).

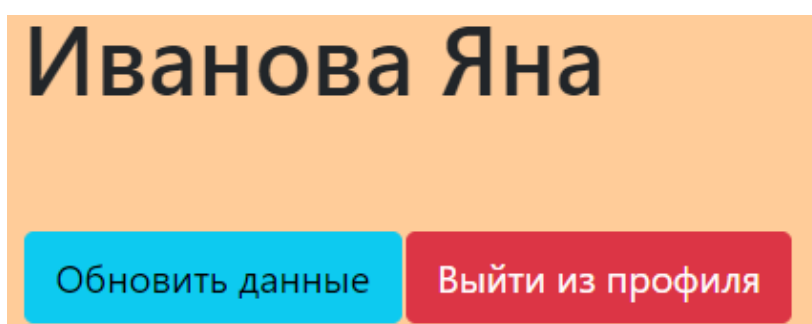


Рисунок 4.16 – профиль пользователя с правами «user»

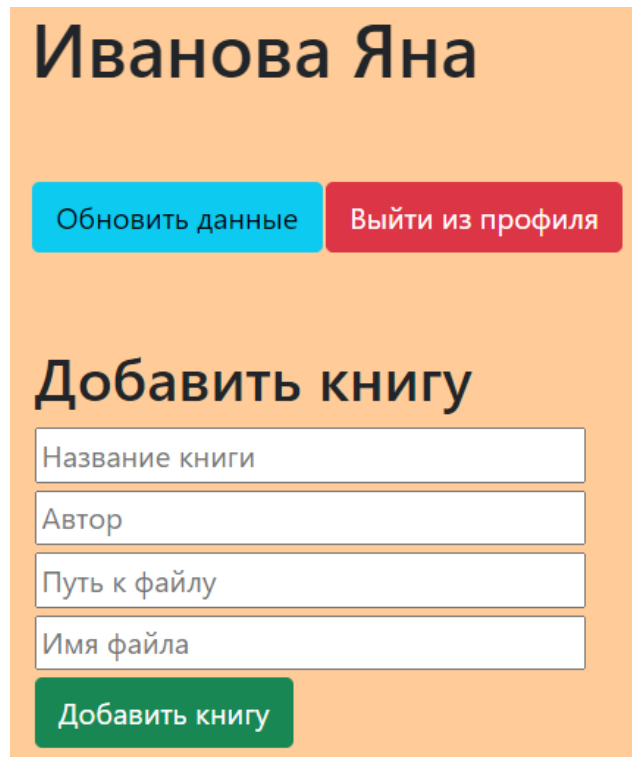


Рисунок 4.17 – профиль пользователя с правами «admin»

Вывод: вывод определённых данных для определённых пользователей работает корректно.

Тест 8: «Добавление книги»

Ожидаемый результат: добавление книги на сайт.

Описание: тестирование правильности добавление книги в таблицу «books» (см. рисунок 4.18, 4.19).

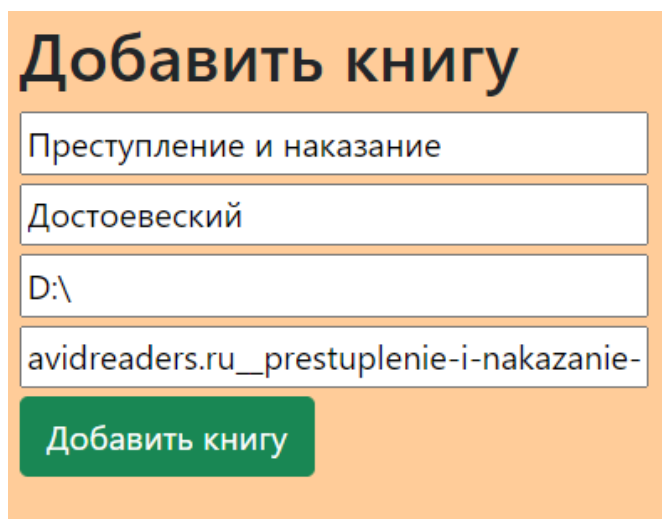


Рисунок 4.18 – форма добавления книги на сервер

	id	created_at	updated_at	deleted_at	name	author	file_name	mark	number_mark
	6	2021-12-20 15:33:53	2021-12-20 15:33:53	NULL	Преступление и наказание	Достоевский	avidreaders.ru__prestuplenie-i-nakazanie-dr-izd.tx...	0.0	0

Рисунок 4.19 – запись в таблице «books»

Вывод: добавление книги работает корректно.

Тест 9: «Вывод списка книг»

Ожидаемый результат: вывода списка книг в вкладке «Все книги».

Описание: тестирование правильности вывода списка книг (см. рисунок 4.20).

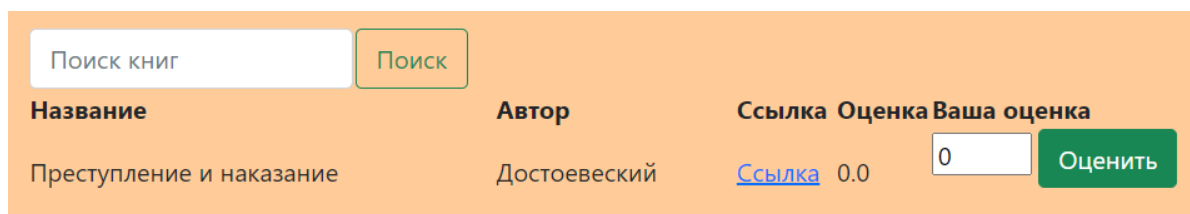


Рисунок 4.20 – вывод новости на главной странице

Вывод: вывод новостей работает корректно.

Тест 12: «Вывод содержания книги»

Ожидаемый результат: вывода содержимой книги.

Описание: тестирование правильности вывода содержания книги (см. рисунок 4.21).

– Я так и знал! – бормотал он в смущении, – я так и думал! Это уж всего сквернее! Вот эдакая какая-нибудь глупость, какая-нибудь пошлейшая мелочь, весь замысел может испортить! Да, слишком приметная шляпа... Смешная, потому и приметная... К моим лохмотьям непременно нужна фуражка, хотя бы старый блин какой-нибудь, а не этот урод. Никто таких не носит, за версту заметят, запомнят... главное, потом запомнят, ан и улика. Тут нужно быть как можно неприметнее... Мелочи, мелочи главное!.. вот эти-то мелочи и губят всегда и все...

Идти ему было немного; он даже знал, сколько шагов от ворот его дома: ровно семьсот тридцать. Как-то раз он их сосчитал, когда уж очень размечтался. В то время он и сам еще не верил этим мечтам своим и только раздражал себя их безобразною, но соблазнительною дерзостью. Теперь же, месяц спустя, он уже начинал смотреть иначе и, несмотря на все поддразнивающие монологи о собственном бессилии и нерешимости, «безобразную» мечту как-то даже поневоле привык считать уже предприятием, хотя все еще сам себе не верил. Он даже шел теперь делать пробу своему предприятию, и с каждым шагом волнение его возрастало все сильнее и сильнее.

С замиранием сердца и нервного дрожью подошел он к преогромнейшему дому, выходившему одною стеной на канаву, а другою в-ю улицу. Этот дом стоял весь в мелких квартирах и заселен был всякими промышленниками – портными, слесарями, кухарками, разными немцами, девицами, живущими от себя, мелким чиновничеством и проч. Входящие и выходящие так и шмыгали под обоими воротами и на обоих дворах дома. Тут служили три или четыре дворника. Молодой человек был очень доволен, не встретив ни которого из них, и неприметно проскользнул сейчас же из ворот направо на лестницу. Лестница была темная и узкая, «черная», но он все уже это знал и изучил, и ему вся эта обстановка нравилась: в такой темноте даже и любопытный взгляд был неопасен. «Если о сю пору я так боюсь, что же было бы, если б и действительно как-нибудь случилось до самого дела дойти?..» – подумал он невольно, проходя в четвертый этаж. Здесь загородили ему дорогу отставные солдаты-носильщики, выносившие из одной квартиры мебель. Он уже прежде знал, что в этой квартире жил

Темная тема | Светлая тема

Рисунок 4.21 – вывод содержания книги

Вывод: вывод содержания книги работает корректно.

Тест 13: «Добавить пользовательскую оценку и рассчитать среднюю оценку»

Ожидаемый результат: добавить пользовательскую оценку и рассчитать среднюю оценку.

Описание: тестирование добавления пользовательской оценки и тестирование новой средней оценки (см. рисунок 4.22).

Преступление и наказание	Достоевский	Ссылка	4.000000	<input type="text" value="0"/>	Оценить
--------------------------	-------------	------------------------	----------	--------------------------------	---------

Рисунок 4.22 – средняя оценка после оценок 3 и 5

Вывод: добавление пользовательской оценки и расчёт её средней оценки работает корректно.

Тест 14: «Смена тем»

Ожидаемый результат: смена на тёмную светлую тему.

Описание: тестирование смены на тёмную и светлую тему (см. рисунок 4.23, 4.24).

– А ты с фактами обращаться умеешь?

– Да ведь нельзя же молчать, когда чувствуешь, ошупом чувствуешь, что вот мог бы делу помочь, кабы... Эх!.. Ты дело-то подробно знаешь?

– Да вот про красильщика жду.

– Да, бишь! Ну, слушай историю: ровно на третий день после убийства, поутру, когда они там нянчились еще с Кохом да Пестряковым, – хотя те каждый свой шаг доказали: очевидность кричит! – объявляется вдруг самый неожиданный факт. Некто крестьянин Душкин, содержатель распивочной, напротив того самого дома, является в контору и приносит ювелирский футляр с золотыми серьгами и рассказывает целую повесть: «Прибежал-де ко мне повечеру, третьего дня, примерно в начале девятого, – день и час! вникаешь? – работник красильщик, который и до этого ко мне на дно забегал, Миколай, и принес мне ефту коробку, с золотыми сережками и с камушками, и просил за них под заклад два рубля, а на мой спрос: где взял? – объявил, что на панели поднял. Больше я его на том не расспрашивал, – это Душкин-то говорит, – а вынес ему билетик – рубль то есть, – потому-де думал, что не мне, так другому заложит; все одно – проплет, а пусть лучше у меня вещь лежит: дальше-де положишь, ближе возьмешь, а объявится что аль слухи пойдут, тут я и представлю». Ну, конечно, бабушкин сон рассказывает, врет, как лошадь, потому я этого Душкина знаю, сам он закладчик и краденое прячет, и тридцатирублевую вещь не для того, чтоб «преставить», у Миколая подтибрил. Просто с

Тёмная тема

Светлая тема

Рисунок 4.23 – средняя оценка после оценок 3 и 5

– А ты с фактами обращаться умеешь?

– Да ведь нельзя же молчать, когда чувствуешь, ошупом чувствуешь, что вот мог бы делу помочь, кабы... Эх!.. Ты дело-то подробно знаешь?

– Да вот про красильщика жду.

– Да, бишь! Ну, слушай историю: ровно на третий день после убийства, поутру, когда они там нянчились еще с Кохом да Пестряковым, – хотя те каждый свой шаг доказали: очевидность кричит! – объявляется вдруг самый неожиданный факт. Некто крестьянин Душкин, содержатель распивочной, напротив того самого дома, является в контору и приносит ювелирский футляр с золотыми серьгами и рассказывает целую повесть: «Прибежал-де ко мне повечеру, третьего дня, примерно в начале девятого, – день и час! вникаешь? – работник красильщик, который и до этого ко мне на дно забегал, Миколай, и принес мне ефту коробку, с золотыми сережками и с камушками, и просил за них под заклад два рубля, а на мой спрос: где взял? – объявил, что на панели поднял. Больше я его на том не расспрашивал, – это Душкин-то говорит, – а вынес ему билетик – рубль то есть, – потому-де думал, что не мне, так другому заложит; все одно – проплет, а пусть лучше у меня вещь лежит: дальше-де положишь, ближе возьмешь, а объявится что аль слухи пойдут, тут я и представлю». Ну, конечно, бабушкин сон рассказывает, врет, как лошадь, потому я этого Душкина знаю, сам он закладчик и краденое прячет, и тридцатирублевую вещь не для того, чтоб «преставить», у Миколая подтибрил. Просто с

Тёмная тема

Светлая тема

Рисунок 4.24 – средняя оценка после оценок 3 и 5

Вывод: смена на тёмную и светлую тему работает корректно.

Тест 15: «Поиск книги по его названию»

Ожидаемый результат: книги, найденные по названию.

Описание: тестирование поиска книги по его названию (см. рисунок 4.25, 4.26).

Поиск книг	Поиск				
Название	Автор	Ссылка	Оценка	Ваша оценка	
Преступление и наказание	Достоевский	Ссылка	4.000000	<input type="text" value="0"/>	Оценить

Рисунок 4.25 – вывод книг с названием «Преступление и наказание»

Поиск книг	Поиск			
Название	Автор	Ссылка	Оценка	Ваша оценка

Рисунок 4.26 – вывод книг с названием «Дубровский»

Вывод: поиск книги по его названию работает корректно.

ЗАКЛЮЧЕНИЕ

В ходе выполнения курсового проекта был разработана онлайн-библиотека, включающий в себя просмотр содержания книги, её добавления, а также авторизацию, аутентификацию, регистрацию. Система была разработана с использованием современных средств разработки.

Был получен опыт проектирования веб-приложения: выбор архитектуры, формат представления данных, способы обмена информацией, описание URI, API, проектирование базы данных. Также были приобретены навыки разработки веб-приложения на языке программирования Go и JS. Были изучены основные принципы разработки REST API сервисов.

Система соответствует требованиям технического задания. Она имеет Backend API сервер написанный на языке Go и клиентскую часть, написанную с помощью HTML, CSS и JS. Система реализована в соответствии с архитектурой «клиент-сервер», которая предполагает инициирование запроса на стороне клиента, его обработка на стороне сервера и формирование ответа, и отправка ответа обратно на сторону клиента. Данный принцип реализовывался при использовании HTTP протокола. Основными методами http-протокола являлись GET – получение данных, POST - создание данных, PUT – обновление данных и DELETE – удаление данных.

Для тестирования системы использовался Postman – приложение, в котором можно выбирать необходимый метод для совершения запроса. Postman имеет несколько преимуществ:

- Бесплатный.
- Простой.
- Поддержка API.
- Расширяемый.
- Интеграция.

В соответствии с требованиями к курсовому проекту исходный код был помещён в публичный репозиторий Github.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. ГОСТ 2.105-95. Единая система конструкторской документации (ЕСКД). Общие требования к текстовым документам.
2. ГОСТ 19.504-79. Единая система программной документации ЕСПД. Руководство программиста. Требования к содержанию и оформлению.
3. ГОСТ 19.701-90. ЕСПД. Схемы алгоритмов, программ, данных и систем. Обозначения условные и правила выполнения.
4. ГОСТ 19.005-85. ЕСПД. Р-схемы алгоритмов и программ. Обозначения условные графические и правила выполнения.
5. ГОСТ 19.101-77. ЕСПД. Виды программ и программных документов.
6. ГОСТ 19.102-77. ЕСПД. Стадии разработки.
7. ГОСТ 19.103-77. ЕСПД. Обозначения программ и программных документов.
8. ГОСТ 19.401-78. ЕСПД. Текст программы. Требования к содержанию и оформлению.
9. ГОСТ 19.402-78. ЕСПД. Описание программы.
10. ГОСТ 7.1-2003. Система стандартов по информации, библиотечному и издательскому делу. Библиографическая запись. Библиографическое описание. Общие требования и правила составления.
11. https://ru.wikipedia.org/wiki/Электронная_библиотека. Определение онлайн библиотеки.
12. <https://coollib.xyz/node/145185>. Преимущества онлайн библиотеки над обычной.
13. <https://vzh.ru/article/html-ili-cms-что-лучше-для-коммерческого-сайта> Описание преимуществ HTML.
14. <http://proglang.su/css/introduction> Описание преимуществ CSS.
15. <https://ipipe.ru/info/javascript> Описание преимуществ JavaScript.
16. <https://techrocks.ru/2021/03/29/golang-online-courses/> Описание преимуществ Go
17. <https://www.methodlab.ru/technology/mysql.shtml> Описание преимуществ MySQL.
18. https://ru.wikipedia.org/wiki/Visual_Studio_Code. Определение VS Code
19. <https://bizzapps.ru/p/vs-code/> Описание преимуществ VS Code.
20. <https://golangify.com/web-project-structure> Файловая структура веб-приложения на GoLang.

21. <https://fokusov.com/posts/razrabotka-web-prilozhenij-i-mikroservisov-na-golang-s-gin> Разработка Web-приложений и микросервисов на Go с Gin.
22. <https://russianblogs.com/article/49761651530/> Определение Gorm.

					<i>КР.ПО4190329-04 81 00</i>	Лист
Изм	Лист	№ докум.	Подп.	Дата		32