|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  | |
|  | **WEB-ПРИЛОЖЕНИЕ ДЛЯ**  **ЗАПОМИНАНИЯ СТИХОВ**  **Пояснительная записка**  **RU.17701729. 02.07-01 81 01-1**  **Листов 22** | | | | |
|  |  | | | |
|  | | | |
|  | | | | |
|  | | |  | |

**2024**

СОДЕРЖАНИЕ

[1. ВВЕДЕНИЕ 3](#_Toc513517902)

[1.1. Наименование разработки 3](#_Toc513517903)

[1.2. Основание для разработки 3](#_Toc513517904)

[2. НАЗНАЧЕНИЕ И ОБЛАСТЬ ПРИМЕНЕНИЯ 4](#_Toc513517905)

[**2.1.** **Функциональное назначение** 4](#_Toc513517906)

[**2.2.** **Эксплуатационное назначение** 4](#_Toc513517907)

[3. ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ 5](#_Toc513517908)

[**3.1.** **Постановка задачи на разработку программы** 5](#_Toc513517909)

[**3.2.** **Описание алгоритма и функционирования программы** 5](#_Toc513517910)

[3.2.1 Клиентская часть 5](#_Toc513517911)

[3.2.2. Серверная часть 6](#_Toc513517912)

[**3.3.** **Организация входных данных** 7](#_Toc513517913)

[3.3.1. Серверная часть 7](#_Toc513517914)

[**3.4.** **Организация выходных данных** 7](#_Toc513517915)

[3.4.1. Серверная часть 7](#_Toc513517916)

[**3.5.** **Конфигурация базы данных** 7](#_Toc513517917)

[**3.6.** **Описание и обоснование выбора и состава технических и программных средств** 8](#_Toc513517918)

[3.6.1. Описание технических и программных средств 8](#_Toc513517919)

[3.6.2. Обоснование выбора технических и программных средств 8](#_Toc513517920)

[4. ТЕХНИКО-ЭКОНОМИЧЕСКИЕ ПОКАЗАТЕЛИ 9](#_Toc513517921)

[4.1. Предполагаемая потребность 9](#_Toc513517922)

[4.2. Ориентировочная экономическая эффективность 9](#_Toc513517923)

[4.3. Экономические преимущества разработки по сравнению с отечественными и зарубежными образцами или аналогами 9](#_Toc513517924)

[5. СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 10](#_Toc513517925)

[ПРИЛОЖЕНИЕ 1. JAVASCRIPT: ОПИСАНИЕ И ФУНКЦИОНАЛЬНОЕ НАЗНАЧЕНИЕ СКРИПТОВ 11](#_Toc513517926)

[ПРИЛОЖЕНИЕ 2. JAVASCRIPT: ОПИСАНИЕ И ФУНКЦИОНАЛЬНОЕ НАЗНАЧЕНИЕ МЕТОДОВ, ПОЛЕЙ И СВОЙСТВ 11](#_Toc513517927)

[ПРИЛОЖЕНИЕ 3. JAVA: МЭППИНГ СЕРВТЛЕТ 13](#_Toc513517928)

[ПРИЛОЖЕНИЕ 4. JAVA: ОПИСАНИЕ И ФУНКЦИОНАЛЬНОЕ НАЗНАЧЕНИЕ КЛАССОВ 14](#_Toc513517929)

[ПРИЛОЖЕНИЕ 5. JAVA: ОПИСАНИЕ И ФУНКЦИОНАЛЬНОЕ НАЗНАЧЕНИЕ МЕТОДОВ, ПОЛЕЙ И СВОЙСТВ 14](#_Toc513517930)

[ЛИСТ РЕГИСТРАЦИИ ИЗМЕНЕНИЙ 22](#_Toc513517931)

1. **ВВЕДЕНИЕ**

## Наименование разработки

**Наименование разработки на русском языке:** Web-приложение для запоминания стихов

**Наименование разработки на английском языке:** Web-application for poems memorizing

1. **НАЗНАЧЕНИЕ И ОБЛАСТЬ ПРИМЕНЕНИЯ**
   1. **Функциональное назначение**

Web-сервис предназначен для облегчения процесса заучивания стихов и текстов, а также повышения его эффективности путем предоставления централизованного места для хранения, повторения, и экзаменования заучиваемых тексов, которое было бы доступно на любых устройствах.

* 1. **Эксплуатационное назначение**

Сервис был задуман преимущественно для помощи учащимся школ, но также он может быть полезен и просто для людей, увлекающихся литературой и желающих развить свою память и эрудицию.

1. **ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ**
   1. **Постановка задачи на разработку программы**

В рамках курсовой работы требуется реализовать web-сервис для заучивания текстов, при регистрации в котором у пользователя появляется возможность:

* добавлять тексты
* повторять тексты в режиме тренировки
* проверять правильность запоминания.

Поставленная задача подразумевает разработку клиентской и серверной части сервиса. Для клиентской части требуется создать современный сайт, который бы обеспечивал возможность комфортного использования на любом устройстве за счет адаптивности и интерактивности. От серверного приложения необходима корректная обработка всех пользовательских запросов даже при невалидных данных или неправомерном доступе.

* 1. **Описание алгоритма и функционирования программы**

### 3.2.1 Клиентская часть

* На главной странице у пользователя имеются три возможности: вход в аккаунт, восстановление пароля и регистрация. При открытии главной страницы сервиса загружаются три JavaScript скрипта: main, sign-in и form-submit, первые два из которых управляют визуальными интерактивными элементами на странице и всплывающими окнами форм соответственно, а последний отвечает за проверку корректности заполняемых форм для действий, связанных с входом в аккаунт, и отправку AJAX запросов на сервер при их валидности. В случае успешного входа в аккаунт обработчики этого события перенаправляют пользователя в его личный кабинет.
* В личном кабинете пользователь может совершать все действия, предусмотренные сервисом. Это обеспечивают три скрипта: sign-in и form-submit и workplace. Первые два уже упоминались ранее, в то время как скрипт workplace управляет интерактивными элементами сайта и отправкой запросов на сервер. После загрузки страницы в скрипте workplace поочередно метод wireEventsToCards привязывает обработчики ко всем карточкам с текстами для действий над ними, wireEventsToSideMenu — к боковому меню для переключения между разделами, wireEventsToPopup — к всплывающему окну, в котором пользователь может просматривать текст, а методы wireEventsToEditMenu, wireEventsToTestMenu, wireEventsToTrainMenu — к контекстным меню для редактирования, экзаменования и тренировки текстов.
* Все запросы добавление новых текстов, редактирование, удаление и т.д. отправляются на сервер по AJAX без перезагрузки страницы, после чего их результаты обрабатываются и вносятся необходимые изменения в структуру страницы для отображения изменений.
* Текст и название записей содержатся в разметке html блока, содержащего разметку записи, и извлекаются из нее для открытия записи во всплывающем окне, а информация о выравнивании текста берется из атрибута text-align. При этом при открытии текста его html блока присваивается переменной openedNote. Для действий с записями, требующих отправки запросов на сервер, используется их идентификатор, который отправляется вместе с другими параметрами запроса. Идентификатор записи берется из атрибута note-id, присутствующего у html блока записи.
* При добавлении нового текста возможна его загрузка из файла. Диалог выбора файла открывается по кнопке в меню, и при выборе файл присваивается скрытому элементу input на странице. Обработка файла происходит без отправки на сервер. Вместо этого он считывается прямо в скрипте workplace с использованием DOM File API. Текст файла считывается с помощью объекта FileReader, используя кодировку UTF-8, и добавляется в поле ввода, а полю с названием текста присваивается название файла.
* В режиме проверки корректности запоминания текста возможен голосовой ввод текста для сравнения. Распознавание текста производится с помощью SpeechKit JavaScript API [14] от Яндекса. При нажатии на кнопку голосового ввода происходит создание сессии распознавания голоса на объекте типа SpeechRecognition. При ее успешной инициализации начинается прослушивание события dataCallback, получающего распознанный текст. По мере распознавания порции текста добавляются в поле ввода.
* Проверка корректности запоминания производится в методе checkTest, где введенный пользователем текст сравнивается с оригинальным. Сравнение производится с помощью алгоритма diff, реализованного в библиотеке jsdifflib [15]. Оба текста разбиваются на слова, после чего полученные массивы используются для создания объекта типа SequenceMatcher, который сравнивает их методом get\_opcodes. Метод возвращает коды сопоставления в виде двумерного массива, строки которого представляются в следующем формате:

[ <флаг>, begA, endA, begB, endB ]

где <флаг> может принимать значение equal, replace, delete или insert, а begA, endA, begB, endB – индексы начала и конца последовательности элементов в первом и втором массиве соответственно, индексы конца не включительно. По полученным кодам в оригинальном тексте производится выделение совпадающих (окрашиваются в зеленый), разнящихся (окрашиваются в желтый) и отсутствующих фрагментов (окрашиваются в красный). Слова в строке оборачиваются в строчные теги, к которым применяются соответствующие стили, окрашивающие их в тот или иной цвет. Также вычисляется процент совпадения текстов методом ratio объекта SequenceMatcher, который отправляется на сервер для обновления прогресса запоминания текста.

* В режиме тренировки пользователю предлагается собрать в конструкторе исходный текст, восстановив верную последовательность из перемешанных слов. Для этого текст разбивается на небольшие фрагменты. Если исходный текст был разделен на параграфы, он разбивается в конструкторе на соответствующие порции, иначе разбиение происходит по предложениям. Пользователь должен собрать поочередно каждый фрагмент из его слов. Метод addWords получает очередной фрагмент текста, делит его на слова, кнопки с которыми в перемешанном порядке добавляет в одну половину всплывающего окна, а во вторую половину добавляя текст с пропусками на месте слов. При этом к словам-кнопкам привязываются обработчики нажатия, которые проверяют, совпадает ли слово кнопки с тем, которое должно быть на месте первого пропуска. Если да, то пропуск заполняется, а слово-кнопка удаляется. При этом если это было последнее слово, то вызывается метод addWords для следующего фрагмента текста или происходит выходит из режима тренировки, если фрагментов больше нет.

### 3.2.2. Серверная часть

* При входе в аккаунт объект User, содержащий информацию о залогиненом пользователе, добавляется к текущей сессии. Если пользователь выбрал опцию “Remember me” в момент входа, создается долговременный файл cookie, содержащий его идентификатор, который позволяет найти пользователя в базе данных после инвалидации сессии.
* За личный кабинет отвечает JSP страница, которая отдает пользователю html код, содержащий его записи. Для получения записей конкретного пользователя используется объект класса NotesDBManager, организующий работу с базой данных и оперирующий объектами класса Note, инкапсулирующего информацию о записи.
* Сервлеты, наследующие абстрактный класс ActionHandler, обрабатывают типовые запросы, требующие обращения к базе данных. Их методы обработки запросов выполняют проверки корректности полученных параметров, конструируют SQL запрос, который требуется выполнить в базе данных, и передают его и объект класса HttpServletResponse унаследованным методам processQuery или processQueryWithReturn, которые выполняют переданный запрос и пишут в объект HttpServletResponse сообщение об удачном выполнении или возникновении ошибки.
* Для предотвращения command-injection attack фильтр HttpFilter заменяет во всех параметрах запросов специальные символы (>, <, &, ") на соответствующие управляющие последовательности.
* Для проверки, аутентифицирован ли пользователь при попытке входа в личный кабинет, фильтр AuthorizationFilter пытается извлечь из сессии ранее записанный в нее объект класса User, хранящий информацию о текущем пользователе, или, если это невозможно, восстановить его по идентификатору из файла cookie с помощью объекта класса UsersDBManager. Если и это не удается, в доступе отказывается и происходит переадресация на главную страницу.
* Для восстановления пароля пользователю на электронную почту отправляется письмо с паролем. Для рассылки используется пакет JavaMail API [22] из Java EE.
  1. **Организация входных данных**

### 3.3.1. Серверная часть

Web-приложение обращается к базе данных для получения информации о пользователях и записях. Проект был разработан для работы с базой данных MySQL. При этом база данных должна быть предварительно сконфигурирована, в ней должны быть созданы таблицы. Сервлеты обращаются за инициализирующими параметрами (url базы данных, имя пользователя и пароль к ней) к объекту класса ServletContext, предоставляемого контейнером сервлетов и доступному в рамках одного web-приложения. Контекст конфигурируется в дескрипторе web.xml, используемом контейнером для развертывания web-приложения. Параметры должны быть сконфигурированы в файле для каждой машины перед деплоем приложения.

* 1. **Организация выходных данных**

### 3.4.1. Серверная часть

Web-приложение добавляет информацию о пользователях и записях в базу данных.

* 1. **Конфигурация базы данных**

Web-приложение манипулирует двумя таблицами — users и notes. Они имеют следующие вид:

Таблица 1

Схема users

|  |  |  |  |
| --- | --- | --- | --- |
| id | name | email | password |
| INT PRIMARY KEY NOT NULL AUTO\_INCREMENT | VARCHAR(50) NOT NULL | VARCHAR(50) NOT NULL | VARCHAR(50) NOT NULL |

Таблица 2

Схема notes

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| id | user\_id | title | text | progress | align |
| PRIMARY KEY NOT NULL AUTO\_INCREMENT | INT NOT NULL | VARCHAR(50) NOT NULL | VARCHAR(10000) NOT NULL | INT DEFAULT 0 | BOOLEAN DEFAULT 0 |

* 1. **Описание и обоснование выбора и состава технических и программных средств**

### 3.6.1. Описание технических и программных средств

#### **Клиентская часть**

Для возможности использования приложения требуется компьютер или мобильный девайс с минимальными техническими характеристиками и браузером, поддерживающим HTML5.

#### **Серверная часть**

Для выполнения web-приложения необходим компьютер с установленным Java SE 8, Apache Tomcat 9 и MySQL. Минимальные технические характеристики ограничены Tomcat 9:

1. 1 Gb оперативной памяти
2. 500 Mb свободного дискового пространства

### 3.6.2. Обоснование выбора технических и программных средств

#### **3.6.2.1. Клиентская часть**

Сайт разрабатывался с использованием фреймворка Material Design Lite [13], который предоставляет набор визуальных адаптивных компонент и модульную сетку для позиционирования элементов. Для повышения эффективности работы с таблицами стилей и улучшения их читаемости при верстке использовался препроцессор Sass, а именно его диалект SCSS, синтаксис которого более близок к обычному CSS. Для компиляции SCSS, расставления вендорных префиксов, оптимизации и минификации стилей и скриптов использовался таск-менеджер Gulp [12].

Сайт использует библиотеку jQuery [17] для упрощения работы с DOM элементами. Для распознавания голоса было решено использовать Yandex SpeechKit JavaScript API [14], потому что оно предоставляет необходимую функциональность и удобное API, а также в силу того, что оно бесплатно для образовательных проектов.

#### **3.6.2.2. Серверная часть**

Web-приложение разрабатывалось на основе Java Servlet API и JSP API, являющиеся базой для большого стека web-технологий на языке Java. В качестве контейнера сервлетов был выбран Apache Tomcat по причинам того, он является одним из наиболее распространенных и популярных HTTP серверов.

1. **ТЕХНИКО-ЭКОНОМИЧЕСКИЕ ПОКАЗАТЕЛИ**

## Предполагаемая потребность

Сервис был задуман преимущественно для помощи учащимся школ для облегчения выполнения домашнего задания, но также он может быть полезен и просто для людей, увлекающихся литературой и желающих развить свою память и эрудицию.

## Ориентировочная экономическая эффективность

Данная работа не является коммерческим проектом, поэтому расчет экономической эффективности не предусмотрен.

## Экономические преимущества разработки по сравнению с отечественными и зарубежными образцами или аналогами

В ходе анализа существующих решений таковых выявлено не было, поэтому проект является уникальным в своем роде.

1. **СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ**
2. Cay S. Horstmann Core Java Volume II (10th edition) – Prentice Hall, 2016, 957 страниц
3. Ian Darwin, Jason Brittain Tomcat: The Definitive Guide, 2nd Edition – O'Reilly Media, 2008, 496 страниц
4. David Flanagan JavaScript: The Definitive Guide, 6th Edition – O'Reilly Media, 2011, 1096 страниц
5. Alan Beaulieu Learning SQL, 2nd Edition – O'Reilly Media, 2009, 338 страниц
6. Gulp API docs [Электронный ресурс] URL <https://github.com/gulpjs/gulp/blob/v3.9.1/docs/API.md>
7. Веб-документация MDN [Электронный ресурс] URL <https://developer.mozilla.org/>
8. W3Schools [Электронный ресурс] URL <https://www.w3schools.com/>
9. Material Design Lite Framework [Электронный ресурс] URL <https://getmdl.io/>
10. Современный учебник Javascript [Электронный ресурс] URL <https://learn.javascript.ru/>
11. jQuery API Documentation [Электронный ресурс] URL <http://api.jquery.com/>
12. Речевые технологии SpeechKit JavaScript API документация [Электронный ресурс] URL <https://tech.yandex.ru/speechkit/jsapi/>
13. jsdifflib - A Javascript visual diff tool & library [Электронный ресурс] URL <https://github.com/cemerick/jsdifflib>
14. Apache Tomcat 9 Documentation [Электронный ресурс] URL <https://tomcat.apache.org/tomcat-9.0-doc/>
15. Java Server-side Programming Tutorials [Электронный ресурс] URL <https://www.ntu.edu.sg/home/ehchua/programming/>
16. JavaMail API – Oracle [Электронный ресурс] URL <https://javaee.github.io/javamail/>

# ПРИЛОЖЕНИЕ 1. JAVASCRIPT: ОПИСАНИЕ И ФУНКЦИОНАЛЬНОЕ НАЗНАЧЕНИЕ СКРИПТОВ

|  |  |
| --- | --- |
| **Скрипт** | **Описание** |
| main | Управление визуальными элементами главной страницы |
| sign-in | Управление всплывающими формами входа в аккаунт |
| form-submit | Отправка форм на сервер |
| workplace | Обработка действий в личном кабинете |

# ПРИЛОЖЕНИЕ 2. JAVASCRIPT: ОПИСАНИЕ И ФУНКЦИОНАЛЬНОЕ НАЗНАЧЕНИЕ МЕТОДОВ, ПОЛЕЙ И СВОЙСТВ

**Скрипт main**

|  |  |  |
| --- | --- | --- |
| Методы | | |
| **Имя** | **Аргументы** | **Назначение** |
| updateNavigation |  | Преобразует боковую навигацию с названиями разделов по мере прокрутки страницы |
| smoothScroll | target | Прокручивает страницу до переданного объекта |

**Скрипт sign-in**

|  |  |  |
| --- | --- | --- |
| Методы | | |
| **Имя** | **Аргументы** | **Назначение** |
| ModalSignin | element | Конструирует объект всплывающей формы входа |
| ModalSignin.prototype.init |  | Привязывает события к элементам формы |
| ModalSignin.prototype.togglePassword | target | Переключает видимость пароля в поле ввода |
| ModalSignin.prototype.showSigninForm | type | Открывает форму входа, регистрации или восстановления пароля в зависимости от переданного типа |
| putCursorAtEnd | element | Перемещает курсор в конец переданного поля ввода |

**Скрипт workplace**

|  |  |
| --- | --- |
| Поля | |
| **Имя** | **Назначение** |
| apikey | Ключ подключения к сервису SpeechKit |
| streamer | Объект типа SpeechRecognition для распознавания голоса |
| is\_listening | Индицирует включено ли распознавание голоса |
| snackbarContainer | Всплывающее уведомление |
| popup | Всплывающее окно для просмотра и действий с записями |
| openedNote | Просматриваемая в данный момент запись |
| train\_text | Разделенный на фрагменты текущий текст, открытый в режиме тренировки |
| train\_text\_pos | Индекс текущего фрагмента текста на тренировке |

|  |  |  |
| --- | --- | --- |
| Методы | | |
| **Имя** | **Аргументы** | **Назначение** |
| wireEventsToTestMenu |  | Добавляет поведение к меню режима тестирования |
| wireEventsToTrainMenu |  | Добавляет поведение к меню режима тренировки |
| wireEventsToEditMenu |  | Добавляет поведение к меню редактирования записи |
| wireEventsToSideMenu | indicator | Устанавливает индикатор прогресса на указанный в его атрибуте процент |
| setIndicator | element | Перемещает курсор в конец переданного поля ввода |
| wireEventsToCards |  | Добавляет обработчики событий к еще непривязанным блокам с записями |
| openNote | noteCard | Переносит запись из переданного записи во всплывающее окно |
| deleteNote | noteCard | Удаляет переданную запись со страницы и делает запрос на удаление серверу |
| markNote | noteCard, with\_query | Помечает переданную запись выученной, внося соответствующие изменения в страницу, и делает запрос на сервер, если второй параметр истинен |
| edit |  | Переключает всплывающее окно с открытым в нем текстом в режим редактирования и открывает соответствующее меню |
| test |  | Переключает всплывающее окно с открытым в нем текстом в режим тестирования и открывает меню |
| checkTest | original\_text, typed\_text | Сравнивает введенный пользователем во время проверки текст с оригинальным и выводит анализ во всплывающее окно |
| train |  | Инициализирует сессию тренировки для открытого во всплывающем окне текста |
| shuffle | arr | Переставляет элементы массива в рандомном порядке |
| checkWord | word,check | Если переданная кнопка-слово совпадает со следующим пропуском, или если второй параметр истинен, проявляет слово на месте пропуска и удаляет переданную кнопку со страницы |
| addWords | text | Добавляет в окно конструктора переданный текст |

# ПРИЛОЖЕНИЕ 3. JAVA: МЭППИНГ СЕРВЛЕТ

|  |  |  |
| --- | --- | --- |
| **Класс** | **Тип** | **URL паттерн** |
| ChangePasswordHandler | Servlet | /u/changepassword/ |
| DeleteAccount | Servlet | /deleteaccount/ |
| DeleteHandler | Servlet | /u/delete/ |
| EditNoteHandler | Servlet | /u/editnote/ |
| LearnedHandler | Servlet | /u/learned/ |
| Login | Servlet | /login/ |
| Logout | Servlet | /logout/ |
| NewNoteHandler | Servlet | /u/newnote/ |
| RecoverPassword | Servlet | /recoverpassword/ |
| Register | Servlet | /register/ |
| UpgradeProgressHandler | Servlet | /u/updateprogress/ |
| Workplace | JSP | /u/ |
| AuthorizationFilter | Filter | /u/ |
| HttpFilter | Filter | /\* |

# ПРИЛОЖЕНИЕ 4. JAVA: ОПИСАНИЕ И ФУНКЦИОНАЛЬНОЕ НАЗНАЧЕНИЕ КЛАССОВ

|  |  |
| --- | --- |
| **Класс** | **Назначение** |
| ActionHandler | Обобщенный класс для типовых запросов, требующих обращения к базе данных |
| AuthorizationFilter | Проверка санкционированности доступа в личный кабинет |
| ChangePasswordHandler | Обработка запроса на изменение пароля пользователя |
| DBManager | Обобщенный класс для манипулирования базой данных |
| DeleteAccount | Обработка запроса на удаление аккаунта |
| DeleteHandler | Обработка запроса на удаление записи |
| EditNoteHandler | Обработка запроса на редактирование записи |
| HttpFilter | Фильтрация параметров запроса |
| LearnedHandler | Обработка запроса на отметку записи выученной |
| Login | Обработка запроса входа в аккаунт |
| Logout | Обработка запроса на выход из аккаунта |
| NewNoteHandler | Обработка запроса добавления новой записи |
| Note | Представляет запись пользователя |
| NotesDBManager | Манипуляции с записями в базе данных |
| RecoverPassword | Обработка запроса на восстановление пароля |
| Register | Обработка запроса на регистрацию нового пользователя |
| UpgradeProgressHandler | Обработка запроса обновления прогресса изучения текста |
| User | Инкапсулирует информацию о пользователе |
| UsersDBManager | Манипуляции с пользователями в базе данных |
| Utils | Вспомогательный класс |

# ПРИЛОЖЕНИЕ 5. JAVA: ОПИСАНИЕ И ФУНКЦИОНАЛЬНОЕ НАЗНАЧЕНИЕ МЕТОДОВ, ПОЛЕЙ И СВОЙСТВ

**Класс ActionHandler**

|  |  |  |  |
| --- | --- | --- | --- |
| Поля | | | |
| **Имя** | **Доступ** | **Тип** | **Назначение** |
| dbConnection | protected | Connection | Соединение с базой данных |
| successMessage | protected | String | Отправляется в ответ при успешном выполнении запроса |
| errorMessage | protected | String | Отправляется в ответ в случае ошибки при выполнении запроса |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Методы | | | | |
| **Имя** | **Доступ** | **Тип** | **Аргументы** | **Назначение** |
| init | public | void |  | Устанавливает соединение с базой данных |
| destroy | public | void |  | Закрывает соединение с базой данных |
| processQueryWithReturn | protected | void | String sqlQuery, HttpServletRequest request, HttpServletResponse response | Выполняет требуемую операцию над базой данных и пишет результат в ответ |
| processQuery | protected | void | String sqlQuery, HttpServletRequest request, HttpServletResponse response | Выполняет требуемую операцию над базой данных и пишет об успешности в ответ |
| doGet | protected | void | HttpServletRequest request, HttpServletResponse response | Передает обработку GET запросов обработчику для POST |

**Класс AuthorizationFilter**

|  |  |  |  |
| --- | --- | --- | --- |
| Поля | | | |
| **Имя** | **Доступ** | **Тип** | **Назначение** |
| usersDBManager | private | UsersDBManager | Получение пользователей из базы данных |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Методы | | | | |
| **Имя** | **Доступ** | **Тип** | **Аргументы** | **Назначение** |
| init | public | void | FilterConfig config | Инициализирует соединение с базой данных |
| destroy | public | void |  | Закрывает соединение с базой данных |
| doFilter | protected | void | ServletRequest  req,  ServletResponse resp,  FilterChain chain | Проверяет санкционированность доступа к аккаунту |

**Класс ChangePasswordHandler**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Методы | | | | |
| **Имя** | **Доступ** | **Тип** | **Аргументы** | **Назначение** |
| init | public | void |  | Устанавливает сообщение об ошибке и успехе |
| doPost | public | void | HttpServletRequest request, HttpServletResponse response resp | Меняет пароль пользователя на новый, если старый был указан правильно |

**Класс DBManager**

|  |  |  |  |
| --- | --- | --- | --- |
| Поля | | | |
| **Имя** | **Доступ** | **Тип** | **Назначение** |
| connection | publuc | Connection | Соединение с базой данных |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Методы | | | | |
| **Имя** | **Доступ** | **Тип** | **Аргументы** | **Назначение** |
| close | public | void |  | Закрывает соединение с базой данных |

**Класс DeleteAccount**

|  |  |  |  |
| --- | --- | --- | --- |
| Поля | | | |
| **Имя** | **Доступ** | **Тип** | **Назначение** |
| usersDBManager | private | UsersDBManager | Получение пользователей из базы данных |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Методы | | | | |
| **Имя** | **Доступ** | **Тип** | **Аргументы** | **Назначение** |
| init | public | void |  | Инициализирует соединение с базой данных |
| destroy | public | void |  | Закрывает соединение с базой данных |
| doPost | protected | void | HttpServletRequest request, HttpServletResponse response resp | Удаляет аккаунт пользователя, если введен правильный пароль |
| doGet | protected | void | HttpServletRequest request, HttpServletResponse response | Передает обработку GET запросов обработчику для POST |

**Класс DeleteHandler**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Методы | | | | |
| **Имя** | **Доступ** | **Тип** | **Аргументы** | **Назначение** |
| init | public | void |  | Устанавливает сообщение об ошибке и успехе |
| doPost | public | void | HttpServletRequest request, HttpServletResponse response resp | Удаляет запись пользователя |

**Класс EditNoteHandler**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Методы | | | | |
| **Имя** | **Доступ** | **Тип** | **Аргументы** | **Назначение** |
| init | public | void |  | Устанавливает сообщение об ошибке и успехе |
| doPost | public | void | HttpServletRequest request, HttpServletResponse response resp | Редактирует запись пользователя |

**Класс HttpFilter**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Методы | | | | |
| **Имя** | **Доступ** | **Тип** | **Аргументы** | **Назначение** |
| doFilter | protected | void | ServletRequest  req,  ServletResponse resp,  FilterChain chain | Удаляет из параметров запросов специальные символы |

**Класс LearnedHandler**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Методы | | | | |
| **Имя** | **Доступ** | **Тип** | **Аргументы** | **Назначение** |
| init | public | void |  | Устанавливает сообщение об ошибке и успехе |
| doPost | public | void | HttpServletRequest request, HttpServletResponse response resp | Помечает запись как выученную |

**Класс Login**

|  |  |  |  |
| --- | --- | --- | --- |
| Поля | | | |
| **Имя** | **Доступ** | **Тип** | **Назначение** |
| usersDBManager | private | UsersDBManager | Получение пользователей из базы данных |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Методы | | | | |
| **Имя** | **Доступ** | **Тип** | **Аргументы** | **Назначение** |
| init | public | void |  | Инициализирует соединение с базой данных |
| destroy | public | void |  | Закрывает соединение с базой данных |
| doPost | protected | void | HttpServletRequest request, HttpServletResponse response resp | Делает попытку войти в аккаунт, инициализирует сессию в случае успеха |
| doGet | protected | void | HttpServletRequest request, HttpServletResponse response | Передает обработку GET запросов обработчику для POST |

**Класс Logout**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Методы | | | | |
| **Имя** | **Доступ** | **Тип** | **Аргументы** | **Назначение** |
| doGet | protected | void | HttpServletRequest request, HttpServletResponse response | Выходит из аккаунта, завершает текущую сессию и удаляет файлы cookie |

**Класс NewNoteHandler**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Методы | | | | |
| **Имя** | **Доступ** | **Тип** | **Аргументы** | **Назначение** |
| init | public | void |  | Устанавливает сообщение об ошибке и успехе |
| doPost | public | void | HttpServletRequest request, HttpServletResponse response resp | Создает новую запись и возвращает ее идентификатор |

**Класс Note**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Методы | | | | |
| **Имя** | **Доступ** | **Тип** | **Аргументы** | **Назначение** |
| getId | public | int |  | Геттеры для полей |
| getTitle | public | String |  |
| getText | public | String |  |
| getProgress | public | int |  |
| getAlign | public | String |  |

**Класс NotesDBManager**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Методы | | | | |
| **Имя** | **Доступ** | **Тип** | **Аргументы** | **Назначение** |
| getAllNotes | public | List<Note> | User user | Возвращает все записи пользователя |

**Класс RecoverPassword**

|  |  |  |  |
| --- | --- | --- | --- |
| Поля | | | |
| **Имя** | **Доступ** | **Тип** | **Назначение** |
| usersDBManager | private | UsersDBManager | Получение пользователей из базы данных |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Методы | | | | |
| **Имя** | **Доступ** | **Тип** | **Аргументы** | **Назначение** |
| init | public | void |  | Инициализирует соединение с базой данных |
| destroy | public | void |  | Закрывает соединение с базой данных |
| doPost | protected | void | HttpServletRequest request, HttpServletResponse response resp | Высылает письмо на почту пользователя с его паролем |
| doGet | protected | void | HttpServletRequest request, HttpServletResponse response | Передает обработку GET запросов обработчику для POST |

**Класс Register**

|  |  |  |  |
| --- | --- | --- | --- |
| Поля | | | |
| **Имя** | **Доступ** | **Тип** | **Назначение** |
| usersDBManager | private | UsersDBManager | Получение пользователей из базы данных и регистрация нового пользователя |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Методы | | | | |
| **Имя** | **Доступ** | **Тип** | **Аргументы** | **Назначение** |
| init | public | void |  | Инициализирует соединение с базой данных |
| destroy | public | void |  | Закрывает соединение с базой данных |
| doPost | protected | void | HttpServletRequest request, HttpServletResponse response resp | Регистрирует нового пользователя и инициализирует сессию |
| doGet | protected | void | HttpServletRequest request, HttpServletResponse response | Передает обработку GET запросов обработчику для POST |

**Класс UpdateProgressHandler**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Методы | | | | |
| **Имя** | **Доступ** | **Тип** | **Аргументы** | **Назначение** |
| init | public | void |  | Устанавливает сообщение об ошибке и успехе |
| doPost | public | void | HttpServletRequest request, HttpServletResponse response resp | Обновляет процент изученности в зависимости от результата теста пользователя |

**Класс User**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Методы | | | | |
| **Имя** | **Доступ** | **Тип** | **Аргументы** | **Назначение** |
| getId | public | int |  | Геттеры для полей |
| getName | public | String |  |
| getEmail | public | String |  |
| getPassword | public | String |  |

**Класс User**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Методы | | | | |
| **Имя** | **Доступ** | **Тип** | **Аргументы** | **Назначение** |
| getUser | public | User | int id | Ищет пользователя в базе данных |
| getUser | public | User | String email |
| getUser | public | User | String email,  String password |
| registerUser | public | User | String name,  String email,  String password | Заносит нового пользователя в базу данных |
| deleteUser | public | boolean | int id | Удаляет пользователя и все его записи из базы данных |

**Класс Utils**

|  |  |  |  |
| --- | --- | --- | --- |
| Поля | | | |
| **Имя** | **Доступ** | **Тип** | **Назначение** |
| logger | public | Logger | Разделяемый между сервлетами логгер |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Методы | | | | |
| **Имя** | **Доступ** | **Тип** | **Аргументы** | **Назначение** |
| getCookieValue | public | String | HttpServletRequest request,  String name | Вспомогательные методы |
| addCookie | public | void | HttpServletResponse response,  String name,  String value,  int maxAge |
| removeCookie | public | void | HttpServletRequest request,  String name |