МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение высшего образования

«Новосибирский национальный исследовательский государственный университет» (Новосибирский государственный университет, НГУ)

Структурное подразделение Новосибирского государственного университета

Высший колледж информатики Университета (ВКИ НГУ)

КАФЕДРА ИНФОРМАТИКИ

ОТЧЕТ ПРОИЗВОДСТВЕННОЙ ПРАКТИКИ

ЭТАП: ПМ07 Анализ и оптимизация кода для повышения качества изделий и производительности разработки

ТЕМА ПРАКТИКИ: Оптимизация программного продукта «Книжный бот рекомендации книг»

Директор компании Потапенко В. А.

E2E4GU.RU «\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_2018

Студент 4 курса Дугуржап Х. Ш.

гр. 403с «\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_2018

Новосибирск

2018

СОДЕРЖАНИЕ

[ПЕРЕЧЕНЬ СОКРАЩЕНИЙ И УСЛОВНЫХ ОБОЗНАЧЕНИЙ 3](#_Toc510130958)

[ВВЕДЕНИЕ 4](#_Toc510130959)

[1 ОПИСАНИЕ ПРЕДМЕТНОЙ ОБЛАСТИ 5](#_Toc510130960)

[2 ПОСТАНОВКА ЗАДАЧИ 6](#_Toc510130961)

[3 ФУНКЦИОНАЛЬНЫЕ ТРЕБОВАНИЯ 7](#_Toc510130962)

[Варианты использования 7](#_Toc510130963)

[4 НЕФУНКЦИОНАЛЬНЫЕ ТРЕБОВАНИЯ 14](#_Toc510130964)

[Требования к программному обеспечению 14](#_Toc510130965)

[Требования к аппаратному обеспечению 14](#_Toc510130966)

[Требования к надежности 14](#_Toc510130967)

[Требования к безопасности 15](#_Toc510130968)

[5 ХАРАКТЕРИСТИКА ВЫБРАННЫХ ПРОГРАММНЫХ СРЕД И СРЕДСТВ 16](#_Toc510130969)

[Описание программных сред и средств 16](#_Toc510130970)

[6 СТРУКТУРА ДАННЫХ 19](#_Toc510130971)

[7 ПРОЕКТИРОВАНИЕ ПОЛЬЗОВАТЕЛЬСКОГО ИНТЕРФЕЙСА 26](#_Toc510130972)

[8 СХЕМА ФУНКЦИОНИРОВАНИЯ ПРОГРАММНОГО СРЕДСТВА 27](#_Toc510130973)

[Описание классов 27](#_Toc510130974)

[9 ОПТИМИЗАЦИЯ ПРОГРАММНОГО КОДА ПРИЛОЖЕНИЯ 31](#_Toc510130975)

[ЗАКЛЮЧЕНИЕ 32](#_Toc510130976)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 33](#_Toc510130977)

[ПРИЛОЖЕНИЕ 1. ЛИСТИНГ ПРОГРАММНОГО КОДА 34](#_Toc510130978)

# ПЕРЕЧЕНЬ СОКРАЩЕНИЙ И УСЛОВНЫХ ОБОЗНАЧЕНИЙ

* API (Application Programming Interface, программный интерфейс приложения) – это набор готовых классов, процедур, функций, структур и констант, предоставляемых приложением (библиотекой, сервисом) или операционной системой для использования во внешних программных продуктах. Используется программистами при написании всевозможных приложений.
* SSL (Secure Sockets Layer, уровень защищённых сокетов) – это цифровая подпись сайта, которая нужна для работы протокола защищенной передачи данных в сети. Проще говоря, она обеспечивает шифрованное соединение между пользователем и сайтом. То есть информация, которой они обмениваются, защищена от посторонних — провайдера, оператора, администратора вайфай-сети и прочих.
* ISBN (International Standard Book Number) – международный стандартный книжный номер.
* БД – база данных.
* СУБД – система управления базами данных.
* XSS (Cross-Site Scripting, межсайтовый скриптинг) – тип атаки на веб-системы, заключающийся во внедрении в выдаваемую веб-системой страницу вредоносного кода (который будет выполнен на компьютере пользователя при открытии им этой страницы) и взаимодействии этого кода с веб-сервером злоумышленника.

# ВВЕДЕНИЕ

Темой данной выпускной квалификационной работы является разработка чат-бота «Книжный бот рекомендации книг» для мессенджера Telegram. Рекомендации позволят лучше ориентироваться в популярной литературе, сократить время на поиск книги. На основе этой задачи были сформулированы следующие функциональные требования:

* Бот рекомендует книги пользователям на основе собранных данных о рецензиях различных книг и отправляет уведомления;
* Бот также сохраняет прочитанные и понравившиеся книги;
* Бот также позволяет осуществить поиск книг;

Данный бот рассчитан на аудиторию пользователей, увлекающихся художественной литературой.

Виды проделанных работ:

* исследование предметной области;
* сбор и анализ функциональных и нефункциональных требований;
* изучение и разработка механизма рекомендательной системы;
* выбор инструментов и средств разработки;
* проектирование архитектуры БД;
* проектирование архитектуры чат-бота;
* тестирование и отладка чат-бота.

Используемые программные средства и технологии:

* язык программирования PHP не ранее версии 7.0;
* база данных MySQL;
* сервер с SSL-сертификатом Heroku для хранения чат-бота;
* Node.JS TelegramBot API.

# 1 ОПИСАНИЕ ПРЕДМЕТНОЙ ОБЛАСТИ

Telegram — это кроссплатформенный мессенджер. Мессенджер позволяет обмениваться сообщениями и медиа-файлами разных форматов. Мессенджер позволяет разрабатывать боты для осуществления поиска в интернете или выполнения иных задач. Боты в режиме диалога предоставляют и собирают данные у пользователей, инициируют различные процессы во внешних системах.

В Telegram используется один общий вид ботов, которых от обычных пользователей отличает только наличие приставки «bot» в имени. Сами же боты делятся на несколько направлений:

* Чат-боты — представляют из себя простейший чат, имитирующий общение на заданную пользователем тематику.
* Боты-информаторы — отдельный вид ботов, главная цель которых — информирование пользователя о тех или иных событиях (новости, мероприятия, публикации и т.п.).
* Игровые боты — боты, в которых можно поиграть в различные игры.
* Боты-асcистенты — боты, разработанные различными онлайн-сервисами как дополнение к основной веб-версии.

Четкого деления нет, так как некоторые боты содержат сразу несколько механик и успешно выполняют множество пользовательских задач. С их помощью можно переводить, обучаться, тестировать, искать информацию, играть в игры и даже пользоваться другими сервисами и взаимодействовать с вещами, имеющими выход в глобальную сеть[1].

# 2 ПОСТАНОВКА ЗАДАЧИ

Цель работы — проектирование и разработка Telegram-бота, который дает рекомендации книг на основе собранных данных о рецензиях различных книг с различных сайтов. Бот также включает в себя следующие возможности:

* поиск книг по авторам, названиям и жанрам;
* составление списка прочитанных книг (на их основе также дают рекомендации);
* поиск сайтов для покупки книг.

Рекомендательная система собирает рецензии людей на различные книги с сайтов LiveLib (www.livelib.ru) и ЛитРес (www.litres.ru), и на основе этих рецензий рекомендует книги пользователям бота. Собранные данные включает в себя все рецензии и книги пользователя книжного сайта. Рекомендация книг зависит от предпочтений пользователя, например, если пользователю нравится рекомендуемая книга или читал её, то система будет рекомендовать любимые книги рецензентов, которые оставили положительный отзыв понравившейся или прочитанной книге пользователя. Другими словами, рекомендательная система работает по принципу, что люди, которым нравится одна и та же книга, могут иметь схожие интересы. Исходя из этого, рекомендует книги пользователям бота.

Исходя из всего вышеперечисленного, была поставлена задача: разработать систему и реализовать её интерфейс с помощью telegram-бота. Система предназначена для того, чтобы рекомендовать книги пользователям, а также для осуществления поиска книг по авторам, названиям, жанрам и для сохранения любимых и прочитанных книг в закладки. Название, автор, ISBN, обложка и прочая информация сохраненных книг хранится в БД. Задача была поставлена научным руководителем.

# 3 ФУНКЦИОНАЛЬНЫЕ ТРЕБОВАНИЯ

В боте предусмотрена только одна категория пользователей, которая имеет возможность выполнять следующие функции:

* просмотр рекомендаций;
* получать уведомления о рекомендуемых книгах;
* Просмотр книг, добавленных в прочитанные или понравившиеся;
* поиск книг по авторам, названиям, жанрам;
* просмотр справки.

Кроме бота, нужно создать рекомендательную систему книг. Рекомендательная система собирает рецензии различных книг. Сбор рецензий осуществляется по сайтам: LiveLib ([www.livelib.ru](http://www.livelib.ru)) и ЛитРес ([www.litres.ru](http://www.litres.ru)). После сбора рецензий, система группирует книги по следующим критериям: оставили рецензию книгам или поставили максимальную оценку один и тот же человек. Затем система будет отправлять рекомендации книг пользователям бота. Каким образом будет собирать рецензии и отправлять рекомендации, подробно показаны в вариантах использования №1 и №2.

Ниже показаны все варианты использования ботом и его реакцию на совершенные действия пользователя. В вариантах использования имеются следующие роли, которые выполняют те или иные действия:

* Пользователь — человек, который пользуется ботом;
* Бот-telegram — бот, реализующий интерфейс рекомендательной системы;
* Система — рекомендательная система.

# Варианты использования

Use Case #1.1

Цель: сбор рецензий

1. Система находит книгу M с сайтов LiveLib ([www.livelib.ru](http://www.livelib.ru)) или ЛитРес ([www.litres.ru](http://www.litres.ru)) и все рецензии на эту книгу;

2. Система ищет рецензии пользователей, которые поставили высокий балл этой книге;

3. Система зафиксирует рецензию N-го пользователя, который поставил высокий балл книге M;

4. Далее начинается Use Case #1.2

Use Case #1.2

Цель: сбор рецензий

1. Система собирает рецензию N-го рецензента на книгу M с сайтов LiveLib ([www.livelib.ru](http://www.livelib.ru)) или ЛитРес (www.litres.ru);

2. Система собирает рецензию N+1-го рецензента на книгу M с сайтов LiveLib ([www.livelib.ru](http://www.livelib.ru)) или ЛитРес (www.litres.ru);

3. Система анализирует, какую оценку поставили данные рецензенты на книгу M;

4. Если оба рецензента поставили высокую оценку на эту книгу, то система объединяет этих рецензентов в одну группу;

5. Система находит рецензии на разные книги с высокими оценками N-го и N+1-го пользователя;

6. Система объединяет все книги в группу G, на которой писали рецензии и поставили высокий балл N-ый и N+1-ый рецензент;

7. Система объединяет эти книги и сохраняет название, автор, год издания, ISBN, краткое описание этих книг. Если пользователь бота выберет, что книга M нравится, то система будет рекомендовать книги из группы G.

Use Case #1.3

Цель: сбор рецензий

1. Система собирает рецензию N-го рецензента на книгу M с сайтов LiveLib ([www.livelib.ru](http://www.livelib.ru)) или ЛитРес (www.litres.ru);

2. Система собирает рецензию N+1-го рецензента на книгу M с сайтов LiveLib ([www.livelib.ru](http://www.livelib.ru)) или ЛитРес (www.litres.ru);

3. Система анализирует, какую оценку поставили данные рецензенты на книгу M;

4. Если рецензенты поставили разные оценки на эту книгу, то система ищет другого N+1-го рецензента, который поставил высокую оценку на эту книгу;

5. Система находит другого рецензента и также анализирует оценку этого рецензента;

Use Case #2.1

Цель: получений рекомендаций

1. Пользователь вводит команду "рекомендация";

2. бот показывает рекомендуемую книгу системы и кнопки «нравится», «прочитано» и «дальше»;

3. пользователь выбирает вариант «нравится»;

4. бот добавляет книгу в понравившиеся книги и показывает следующую книгу, похожую на первую книгу и те же кнопки.

Use Case #2.2

Цель: получений рекомендаций

1. Пользователь вводит команду «рекомендация»;

2. бот показывает рекомендуемую книгу и кнопки «нравится», «прочитано» и «дальше»

3. пользователь выбирает вариант «прочитано»;

4. бот добавляет книгу в прочитанные книги и показывает следующую книгу, похожую на первую книгу и те же кнопки.

Use Case #2.3

Цель: получений рекомендаций

1. Пользователь вводит команду «рекомендация»

2. бот показывает рекомендуемую книгу и кнопки «нравится», «прочитано» и «дальше»

3. пользователь выбирает вариант «дальше»

4. бот показывает следующую книгу, не похожую на первую книгу и те же кнопки

Use Case #3.1

Цель: получение уведомлений о рекомендуемых книгах

1. Пользователь вводит команду «уведомления»

2. бот показывает список пунктов, сколько раз в день пользователь хочет получить рекомендации:

\* 1 книга раз в день

\* 2 книги раз в день

\* 1 книга раз в неделю

\* 5 книг раз в неделю (каждое воскресенье)

\* 10 книг раз в месяц (в конце каждого месяца)

\* не получать уведомления;

3. пользователь выбирает один из пунктов, бот будет отправлять уведомления в соответствии с выбором пользователя.

Use Case #3.2

Цель: пользователь только запустил бота и еще не подписался на уведомления

1. Бот не будет отправлять никаких уведомлений о рекомендуемых книгах системы, пользователь может просмотреть рекомендуемые книги, набрав команду «рекомендация» (UseCase #1), а чтобы подписаться на уведомления, вводить команду «уведомления» (UseCase #2.1)

Use Case #3.3

Цель: пользователь подписался на уведомления и хочет отписаться от уведомлений

1. Пользователь вводит команду «уведомления»;

2. бот показывает список пунктов, сколько раз в день пользователь хочет получить рекомендации, среди них есть пункт "не получать уведомления" (UseCase #2.1, этап 2);

3. пользователь выбирает пункт «не получать уведомления» и бот перестает отправлять уведомления, чтобы заново подписаться на уведомления, пользователю нужно вводить команду «уведомления».

Use Case #4.1

Цель: просмотр книг, добавленных в «прочитанные»

1. Пользователь вводит команду «прочитанные»;

2. бот выводит список книг, добавленных в прочитанные;

3. пользователь выбирает одну из книг;

4. бот показывает информацию о книге: название, автор, год издания, ISBN, краткое описание, жанр.

Use Case #4.2

Цель: просмотр книг, добавленных в «понравившиеся»

1. Пользователь вводит команду «понравившиеся»;

2. бот выдает список книг, добавленных в понравившихся;

3. пользователь выбирает одну из книг;

4. бот показывает информацию о книге: название, автор, год издания, ISBN, краткое описание, жанр.

Use Case #4.3

Цель: просмотр книг, добавленных в «понравившиеся» или «прочитанные» (не найдены книги)

1. Пользователь вводит команды «понравившиеся» или «прочитанные»;

2. бот выводит сообщение, что пока нет прочитанных или понравившихся книг.

Use Case #5.1

Цель: поиск книг по названию

1. Пользователь вводит команду «поиск»;

2. бот выдает справку о том, какие команды нужно ввести, чтобы искать книги:

\* поиск по авторам: «автор {{имя автора}}»

\* поиск по жанрам «жанр {{название жанра}}»

\* поиск по названиям «книга {{название книги}}»;

3. пользователь вводит «книга {{название книги}}»;

4. бот выдает список всех книг, где в названии {{название книги}};

5. пользователь выбирает нужную книгу и заходит в страницу книги;

6. на странице книги показана следующая информация – название книги, автор, год издания, ISBN, краткое описание, а также кнопки «прочитано», «нравится», «купить»

Use Case #5.2

Цель: поиск книг по авторам

1. Пользователь вводит команду «автор {{имя автора}}»;

2. бот выдает список всех книг автора {{имя автора}}

3. пользователь выбирает нужную книгу и заходит в страницу книги;

4. на странице книги показана следующая информация: название книги, автор, год издания, ISBN, краткое описание, кнопки «прочитано», «нравится», «купить».

Use Case #5.3

Цель: поиск книг по жанрам

1. Пользователь вводит команду «жанр {{название жанра}}»;

2. бот выдает список всех книг жанра {{жанр}};

3. пользователь выбирает нужную книгу и заходит в страницу книги;

4. на странице книги показана следующая информация — название книги, автор, год издания, ISBN, краткое описание, кнопки "прочитано", «нравится», «купить».

Use Case #5.4

Цель: поиск книг (ошибка при поиске)

1. Пользователь вводит команду «книга {{...}}» или «автор {{...}}» или «жанр {{...}}»;

2. бот выдает, что результат не найден.

Use Case #6.1

Цель: просмотр справки

1. Пользователь вводит команду «справка»;

2. бот показывает всю справку, все команды бота и краткое описание.

# 4 НЕФУНКЦИОНАЛЬНЫЕ ТРЕБОВАНИЯ

# Требования к программному обеспечению

Для обеспечения функционирования программного средства необходимо использовать следующие программные средства:

* базу данных MySQL версии не ниже 5.7;
* интерпретатор PHP версии не ниже 7.0;
* Framework CakePHP версии не ниже 3.5;
* Node.JS Telegram Bot API;
* Amazon API;
* GoodReads API.

# Требования к аппаратному обеспечению

Для обеспечения функционирования бота необходимо использовать:

* количество свободного места на физическом накопителе не меньше 16 ГБ;
* частота процессоров не ниже 1.5 GHz;

Минимальные требования к девайсу клиента:

* установленный мессенджер Telegram.

# Требования к надежности

К боту предъявляются следующие требования в плане надежности:

* система должна проверять вводимые команды пользователя, если команда корректная, то бот должен показать соответствующий вывод команды, иначе сообщение о том, что такая команда не существует.

# Требования к безопасности

Для обеспечения безопасности бота необходимы следующие требования:

* бот должен храниться в сервере с SSL-сертификатом для обеспечения шифрования;
* система должна обеспечивать защиту от XSS и SQL-инъекций.

# 

# 5 ХАРАКТЕРИСТИКА ВЫБРАННЫХ ПРОГРАММНЫХ СРЕД И СРЕДСТВ

В таблице 1 представлены тип и название используемых программных сред и средств:

|  |  |
| --- | --- |
| **Тип ПС** | **Название** |
| Операционная система | Ubuntu 16.04 |
| Среды разработки | NetBeans IDE 8.2,  Atom,  Sublime Text 3 |
| Редактор диаграмм | Dia |
| Инструмент проектирования БД | MySQL Workbench |
| Система контроля версий | Git |
| Языки программирования | PHP, JavaScript |
| СУБД | MySQL |
| Программные платформы (фреймворки) | CakePHP 3.5.12,  Node.js Telegram Bot API |
| Веб-сервис | Bitbucket.org |

Таблица 2 — Authors

# Описание программных сред и средств

Ubuntu — операционная система, основанная на Debian GNU/Linux.

NetBeans IDE 8.2 — свободная интегрированная среда разработки приложений на языках программирования PHP, Python, Java, JavaScript и других. Основные преимущества: кроссплатформенность, свободное ПО, интеграция с git, огромное количество плагинов, подсветка синтаксиса поддерживаемых языков. Основные недостатки: необходимость установки Java Virtual Machine, количество поддерживаемых языков ограничено, низкая производительность.

Atom — бесплатный текстовый редактор с открытым исходным кодом для Linux, macOS, Windows с поддержкой плагинов, написанных на Node.js и встраиваемых под управлением Git [2]. Основные преимущества: кроссплатформенность, интеграция с git, подсветка синтаксиса разных языков, открытый исходный код, свободное ПО, огромное количество плагинов, высокая производительность. Основной недостаток (в то же время еще и преимущество): дополнительная настройка (сложная для новичков) для расширяемости функционала, а также слежение за пользователями.

Sublime Text 3 — кроссплатформенный проприетарный графический текстовый редактор. Основные преимущества: кроссплатформенность, поддержка огромного количества языков, высокая производительность, расширяемость функционала, огромное количество плагинов. Недостатки: проприетарное ПО, установка дополнительных плагинов для расширяемости функционала.

Dia — свободный кроссплатформенный редактор диаграмм. Он может быть использован для создания различных видов диаграмм: блок-схем алгоритмов программ, древовидных схем, статических структур UML, баз данных и других [3]. Основные преимущества: кроссплатформенность, бесплатность.

MySQL Workbench — инструмент для визуального проектирования баз данных, интегрирующий проектирование, моделирование, создание и эксплуатацию БД в единое бесшовное окружение для системы баз данных MySQL [4].

Git — распределенная система контроля версий.

PHP — скриптовый язык программирования общего назначения, применяемый в основном для разработки веб-приложений.

JavaScript — мультипарадигменный язык программирования,

MySQL — свободная реляционная система управления базами данных.

CakePHP — программная платформа (фреймворк) для создания веб-приложений, написанный на языке PHP. Преимущества: открытый исходный код (лицензия MIT), поддержка множества плагинов, поддержка множества СУБД, имеется автогенератор кода bake, консольная интеграция, миграции и другие.

Node.js Telegram Bot API — API для разработки Telegram-бота на программной платформе Node.js. Преимущества: открытый исходный код (лицензия MIT), имеется документация.

Node.js — программная платформа, превращающая JavaScript из узкоспециализированного языка в язык общего назначения. Node.js добавляет возможность JavaScript взаимодействовать с устройствами ввода-вывода через свой API, подключать другие внешние библиотеки, написанные на разных языках, обеспечивая вызовы к ним из JavaScript-кода [5].

Bitbucket.org — веб-сервис для хостинга проектов и их совместной разработки, основанный на системе контроля версий Merkurial и Git.

# 6 СТРУКТУРА ДАННЫХ

В таблице Authors хранится информация об авторах книг. В таблице 2 представлено описание структуры таблицы Authors:

|  |  |  |  |
| --- | --- | --- | --- |
| Название | Описание | Тип | Ключ |
| id | ID автора | Счетчик | Первичный |
| full\_name | Полное имя автора книг | Текстовый | нет |
| biography | Биография автора | Текстовый | нет |

Таблица 2 — Authors

В таблице Books хранится информация о книгах. В таблице 3 представлено описание структуры таблицы Books:

|  |  |  |  |
| --- | --- | --- | --- |
| Название | Описание | Тип | Ключ |
| id | ID книги | Счетчик | Первичный |
| name | Название книги | Текстовый | нет |
| authors | Авторы книги | Текстовый | нет |
| publication\_date | Дата публикации книги | Числовой | нет |
| cover | Обложка книги | Текстовый | нет |
| genres | Жанры книги | Текстовый | нет |
| isbn | ISBN книги | Текстовый | Уникальный |
| rating | Общий рейтинг книги на сайте livelib | Текстовый | нет |
| description | Описание книги | Текстовый | нет |
| link | Ссылка книги на сайте livelib или litres | Текстовый | нет |
| fromsite | С какого сайта был собрана информация о книге (livelib или litres) | Текстовый | нет |
| reviews\_count | Количество рецензий книги | Числовой | нет |
| quote\_id | ID цитаты из книги | Числовой | нет |

Таблица 3 — Books

В таблице Bot\_commands хранится информация о командах, выполняемых ботом. В таблице 4 представлено описание структуры таблицы Bot\_commands:

|  |  |  |  |
| --- | --- | --- | --- |
| Название | Описание | Тип | Ключ |
| id | ID команды | Счетчик | Первичный |
| command | Название команды | Текстовый | нет |
| description | Описание команды | Текстовый | нет |

Таблица 4 — Bot\_commands

В таблице Genres хранится название и ссылки жанров. В таблице 5 представлено описание структуры таблицы Genres:

|  |  |  |  |
| --- | --- | --- | --- |
| **Название** | **Описание** | **Тип** | **Ключ** |
| id | ID жанра | Счетчик | Первичный |
| name | полное название жанра | Текстовый | нет |
| link\_livelib | ссылка жанра в livelib | Текстовый | нет |
| link\_litres | ссылка жанра в litres | Текстовый | нет |

Таблица 5 — Genres

В таблице Liked\_books хранится информация о книгах, понравившихся пользователям бота. В таблице 6 представлено описание структуры таблицы Liked\_books:

|  |  |  |  |
| --- | --- | --- | --- |
| Название | Описание | Тип | Ключ |
| id | ID записи | Счетчик | Первичный |
| user\_id | ID пользователя, который сохранил книгу | Числовой | Внешний |
| book\_id | ID понравившейся (прочитанной) книги | Числовой | Внешний |
| status | Выбор пользователя (добавлено, удалено) | Текстовый | нет |

Таблица 6 — Liked\_books

В таблице Notification\_books хранится информация о книгах, уведомляемых пользователям. В таблице 7 представлено описание структуры таблицы Notification\_books:

|  |  |  |  |
| --- | --- | --- | --- |
| Название | Описание | Тип | Ключ |
| id | ID записи | Счетчик | Первичный |
| user\_id | ID пользователя | Числовой | Внешний |
| book\_id | ID книги | Числовой | Внешний |
| review\_id | ID рецензии | Числовой | Внешний |
| reader\_id | ID рецензента | Числовой | Внешний |
| notification\_id | ID нотификации | Числовой | Внешний |
| user\_choose | Выбор пользователя (нравится, читал, не нравится) | Текстовый | нет |

Таблица 7 — Notification\_books

В таблице Notifications хранится информация о записях, которые нужно отправлять пользователям бота. В таблице 8 представлено описание структуры таблицы Notifications:

|  |  |  |  |
| --- | --- | --- | --- |
| Название | Описание | Тип | Ключ |
| id | ID нотификации | Счетчик | Первичный |
| user\_id | ID пользователя | Числовой | Внешний |
| frequency | Частота отправки уведомления пользователю | Числовой | нет |

Таблица 8 — Notifications

В таблице Quotes хранится информация о цитатах из книг. В таблице 9 представлено описание структуры таблицы Quotes:

|  |  |  |  |
| --- | --- | --- | --- |
| Название | Описание | Тип | Ключ |
| id | ID цитаты | Счетчик | Первичный |
| content | содержание цитаты | Текстовый | нет |
| book\_id | ID книги | Числовой | Внешний |

Таблица 9 — Quotes

В таблице Readers хранится информация о читателях, пишущих рецензии на сайтах. В таблице 10 представлено описание структуры таблицы Readers:

|  |  |  |  |
| --- | --- | --- | --- |
| Название | Описание | Тип | Ключ |
| id | ID читателя | Счетчик | Первичный |
| username | Имя пользователя | Текстовый | нет |
| fromsite | С какого сайта читатель | Текстовый | нет |
| reviews\_count | Количество написанных рецензий читателя | Числовой | нет |
| link | Ссылка на профиль читателя | Текстовый | нет |

Таблица 10 — Readers

В таблице Recommended\_books хранится информация о книгах, которые нужно рекомендовать пользователям бота. В таблице 11 представлено описание структуры таблицы Recommended\_books:

|  |  |  |  |
| --- | --- | --- | --- |
| Название | Описание | Тип | Ключ |
| id | ID рекомендации | Счетчик | Первичный |
| book\_id | ID книги | Числовой | Внешний |
| reader\_id | ID рецензента | Числовой | Внешний |
| review\_id | ID рецензии | Числовой | Внешний |
| user\_id | ID пользователя | Числовой | Внешний |
| user\_choose | Выбор пользователя (нравится, читал, не нравится) | Текстовый | нет |

Таблица 11 — Recommended\_books

В таблице Reviews хранится информация о рецензиях. В таблице 12 представлено описание структуры таблицы Reviews:

|  |  |  |  |
| --- | --- | --- | --- |
| Название | Описание | Тип | Ключ |
| id | ID рецензии | Счетчик | Первичный |
| book\_id | ID книги | Числовой | Внешний |
| reader\_id | ID рецензента | Числовой | Внешний |
| rate | Проставленная оценка рецензента книге | Числовой | нет |

Таблица 12 — Reviews

В таблице Users хранится информация о пользователях ботов. В таблице 13 представлено описание структуры таблицы Users:

|  |  |  |  |
| --- | --- | --- | --- |
| Название | Описание | Тип | Ключ |
| id | ID пользователя | Счетчик | Первичный |
| username | Имя пользователя | Текстовый | нет |
| telegram\_id | ID пользователя в telegram | Числовой | нет |

Таблица 13 — Users

В таблице Collection\_books хранится название, ISBN, ссылки собираемых книг. В таблице 14 представлено описание структуры таблицы Collection\_books:

|  |  |  |  |
| --- | --- | --- | --- |
| Название | Описание | Тип | Ключ |
| id | ID записи | Счетчик | Первичный |
| isbn | ISBN книги | Текстовый | Уникальный |
| link | ссылка книги | Текстовый | нет |
| status | статус сбора (new - если данные о книге не собраны, added - если собраны) | Текстовый | нет |
| name | Название книги | Текстовый | нет |

Таблица 14 — Collection\_books

В таблице Collection\_reviews хранится ISBN, ссылки рецензий собираемых рецензий. В таблице 15 представлено описание структуры таблицы Collection\_reviews:

|  |  |  |  |
| --- | --- | --- | --- |
| Название | Описание | Тип | Ключ |
| id | ID записи | Счетчик | Первичный |
| isbn | ISBN книги | Текстовый | Уникальный |
| link | ссылка на рецензий книги | Текстовый | нет |
| reviews\_count | количество рецензий | Числовой | нет |
| status | статус сбора (если все рецензии книги собраны - collected, иначе - new) | Текстовый | нет |

Таблица 15 — Collection\_reviews

На рисунке 1 представлена концептуальная схема данных программного продукта «Книжный бот рекомендации книг»:

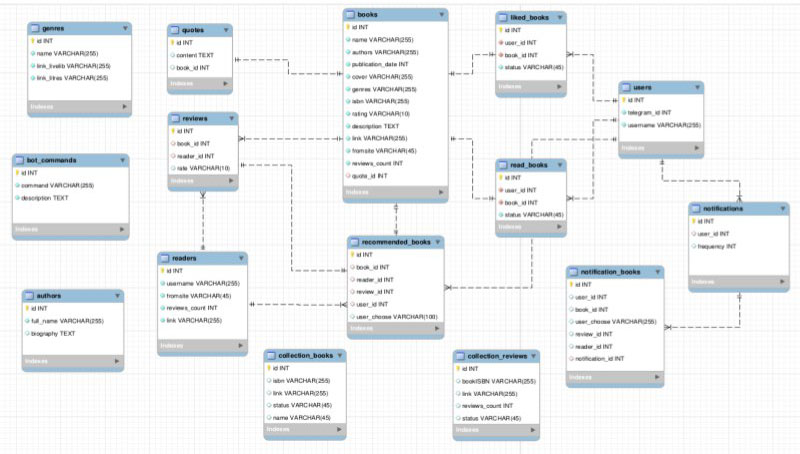


Рисунок 1 — Концептуальная схема данных

# 7 ПРОЕКТИРОВАНИЕ ПОЛЬЗОВАТЕЛЬСКОГО ИНТЕРФЕЙСА

Интерфейс бота на данный момент выглядит как на рисунке 2. Пользователи могут отправить команды, затем бот обрабатывает введенный запрос пользователя и выводит результат на экран мессенджера Telegram:

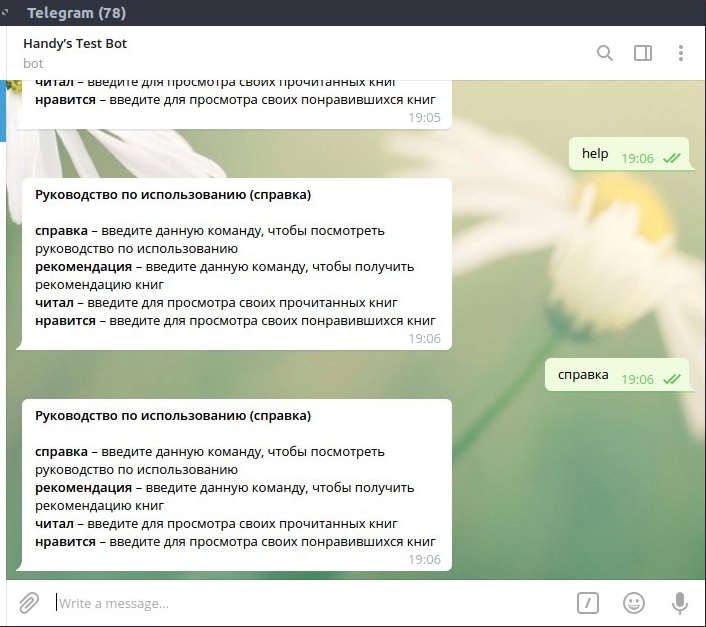


Рисунок 2 — Интерфейс telegram-бота

# 8 СХЕМА ФУНКЦИОНИРОВАНИЯ ПРОГРАММНОГО СРЕДСТВА

Разработка программного продукта делится на 3 части:

* разработка подсистемы для сбора данных о книгах и рецензиях;
* разработка механизма рекомендации книг;
* разработка пользовательского интерфейса в виде Telegram-бота.

На данном этапе выполнена частичная реализация программного продукта, а именно реализована подсистема для сбора данных о книгах и рецензиях, а также модели данных.

# Описание классов

Класс CollectionBooksShell — этот класс реализует сбор данных о книгах из сайтов livelib.ru и litres.ru и автоматизирует данный процесс. У него есть три метода:

* collectionBooksByGenres — метод собирает ссылку, название и isbn всех книг определенного жанра и передает собранные данные классу CollectionBooksTable для дальнейшего занесения полученных данных в таблицу collection\_books в БД.
* addBook — метод собирает подробную информацию о книгах, добавленных в таблицу collection\_books, затем, если собранные данные корректны, передает их классу BooksTable для занесения в таблицу books в БД.
* main — этот метод вызывает методы выше в назначенное время, таким образом запускает процесс сбора данных о книгах.

Класс CollectionReviewsShell — этот класс реализует сбор данных о рецензиях из сайтов livelib.ru и litres.ru и автоматизирует данный процесс. У него есть три метода:

* collectionReviewsByBooks — метод собирает ссылки рецензий определенных книг и передает собранные данные классу CollectionReviewsTable для занесения полученных данных в таблицу collection\_reviews в БД.
* addCollectionReview — метод собирает все данные о рецензиях книг, добавленных в таблицу collection\_reviews, затем, если собранные данные отсутствуют в БД и они корректны, вызывает метод addReview класса ReviewsTable для добавления рецензии в БД.
* main — этот метод вызывает методы выше в назначенное время, таким образом запускает процесс сбора данных о книгах.

Класс CollectionBooksTable — класс для получения, обработки, проверки, занесения и передачи данных таблицы collection\_books. Класс является частью модели collection\_books. На данный момент у него есть три метода:

* addBook — добавление ссылки, ISBN, название новых книг в БД. У него есть входной параметр (массив данных о книге) и выходной параметр (true/false в зависимости от того, добавлена книга в БД или нет);
* checkISBN — метод проверяет наличие книги в БД по ISBN. Возвращает true, если была найдена книга, иначе false.
* changeBookStatus — метод меняет статус книги на added, если были собрана информация о книге и была добавлена в таблицу books.

Класс CollectionReviewsTable — класс для получения, обработки, проверки, занесения и передачи данных таблицы collection\_reviews. Класс является частью модели collection\_books. На данный момент у него есть один метод:

* addBookReviews — добавление ссылок на рецензии, ISBN, название книг в БД. У него есть входной параметр (массив данных о книге) и выходной параметр (true/false в зависимости от того, добавлена информация в БД или нет);

Класс BooksTable — класс для получения, обработки, проверки, занесения и передачи данных таблицы books. На данный момент есть два метода:

* addBook — метод добавляет собранную информацию о книге в БД. У него есть входной параметр (массив данных книг) и выходной параметр (true/false в зависимости от того, добавлена книга в БД или нет);
* getBookByISBN — метод получает данные определенной книги по ISBN. У него есть входной параметр (ISBN книги) и выходной параметр (данные полученной книги из БД).

Класс ReviewsTable — класс для получения, обработки, проверки, занесения и передачи данных таблицы reviews. На данный момент у него есть два метода:

* addReview — метод добавляет собранную информацию о рецензиях в БД. У него есть входной параметр (массив данных о рецензиях) и выходной параметр (true/false в зависимости от того, добавлена рецензия в БД или нет);
* getReviewByBookAndReaderId — метод получает данные рецензии из БД по индивидуальным номера книг и рецензента. У него есть два входных параметра (ID книги и ID рецензента) и выходной параметр (данные полученной рецензии из БД).

Помимо вышеописанных классов и методов существуют другие классы, которые на данном этапе пока не используются. На рисунке 3 показаны диаграммы всех существующих классов.

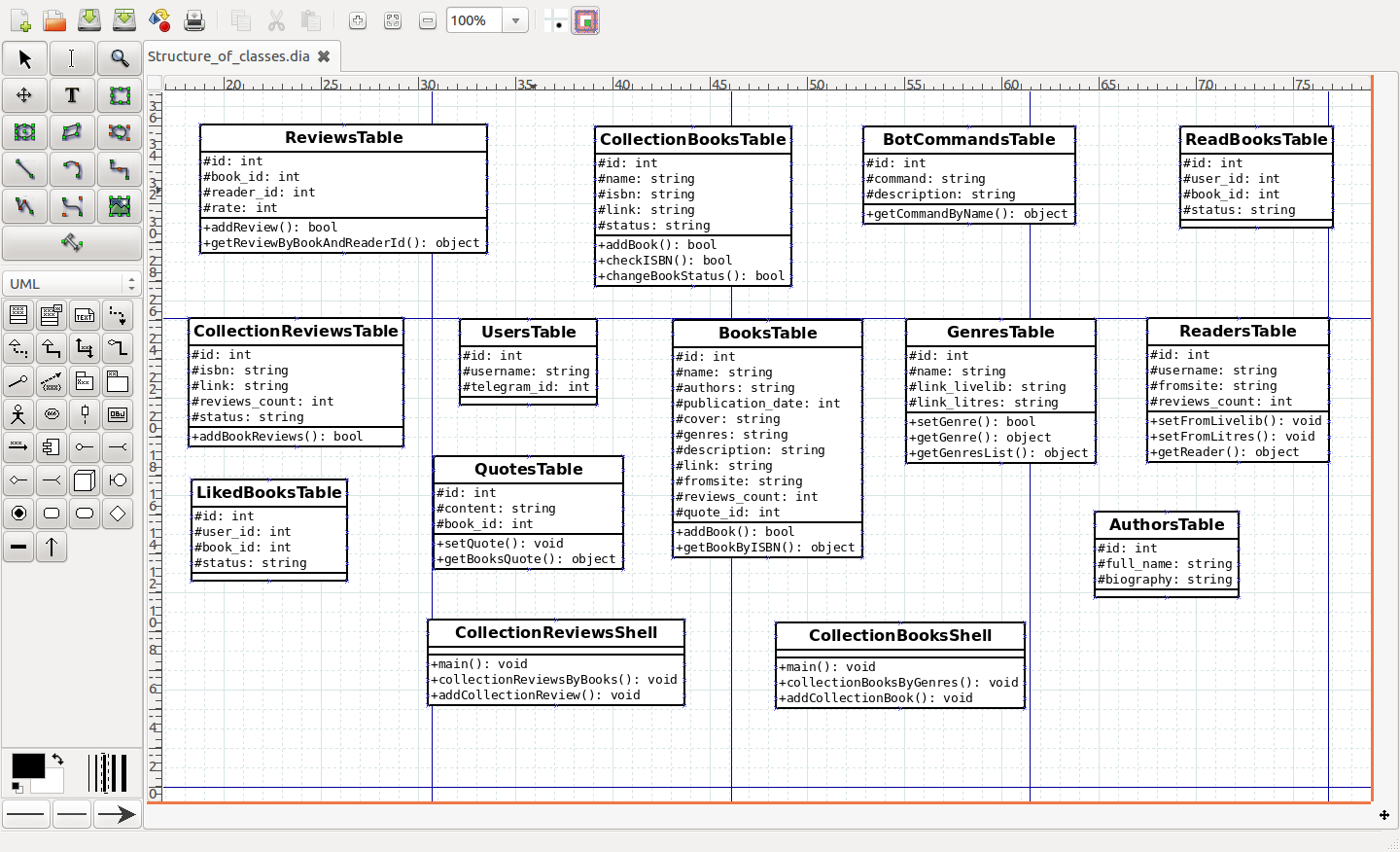


Рисунок 3 — UML-диаграмма классов

# 9 ОПТИМИЗАЦИЯ ПРОГРАММНОГО КОДА ПРИЛОЖЕНИЯ

Были проведены анализ и оптимизация программного кода для повышения производительности и эффективности работы рекомендательной системы бота. Было оптимизировано следующее:

* Механизм рекомендательной системы реализован на платформе Node.JS. Данная платформа позволяет эффективно обрабатывать операции ввода-вывода с большими данными.
* Запросы получения необходимой информации из БД выполнены с помощью ORM Sequelize для повышения производительности.
* Некоторые участки кода, которые снижают эффективность, были полностью переписаны.
* Для увеличения скорости работы СУБД были минимизированы количество операций ввода-вывода.

# ЗАКЛЮЧЕНИЕ

В рамках производственной практики «Участие в анализе и разработке требований к программному продукту» были выполнены следующие пункты:

* Проведено исследование предметной области.
* Сформулированы функциональные и нефункциональные требования к боту.
* Выбраны инструменты и средства проектирования, разработки бота.
* Спроектирована структура базы данных и выбрана архитектура программного продукта.
* Выбраны инструменты и средства для проектирования и разработки программного продукта.
* Выполнена оптимизация программного кода для повышения качества и производительности программного продукта.

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Боты в Telegram и как они работают [Электронный ресурс] – URL: <https://ru.epicstars.com/botyi-telegram> - (Дата обращ. 05.02.2018)
2. Atom: новый редактор от Гитхаба [Электронный ресурс] – URL: <https://habrahabr.ru/post/214079/> - (Дата обращ. 08.02.2018)
3. Dia [Электронный ресурс] – URL: <https://wiki.gnome.org/Apps/Dia>
4. MySQL Workbench [Электронный ресурс] – URL: <https://www.mysql.com/products/workbench/> - (Дата обращ. 06.02.2018)
5. Node.js – Руководство для начинающих [Электронный ресурс] – URL: <https://nodeguide.ru/doc/felix/beginner/> (Дата обращ. 14.02.2018)

# ПРИЛОЖЕНИЕ 1. ЛИСТИНГ ПРОГРАММНОГО КОДА

Класс CollectionBooksShell – этот класс выполняет автоматизацию сбора данных о книгах

class CollectionBooksShell extends Shell {

/\*\*

\* main() method.

\*

\* @return bool|int|null Success or error code.

\*/

public function main() {

$this->collectionBooksByGenres();

//$this->addCollectionBook();

}

public function collectionBooksByGenres() {

$id = 14;

$livelib = LIVELIB;

$genre = $this->Genres->find('all')->where(['id' => $id])->first();

if (empty($genre)) {

$this->out('Запись не найдена');

} else {

$pageOfGenre = $genre['link\_livelib'];

$dom = new \DOMDocument("1.0", "UTF-8");

$internalErrors = libxml\_use\_internal\_errors(true);

$dom->loadHTMLFile($pageOfGenre);

$finder = new \DOMXPath($dom);

$bookName = "block-book-title";

$bookISBN = "book-details-info";

$query1 = sprintf("//a[contains(@class, '%s')]", $bookName);

$query2 = sprintf("//\*[contains(@class, '%s')]", $bookISBN);

$books = $finder->query($query1);

$infos = $finder->query($query2);

for ($id = 0; $id < $books->length; $id++) {

$isbn = $infos[$id]->nodeValue;

$afterisbn = stristr($isbn, 'ISBN: ');

if (!empty($afterisbn)) {

$keywords = preg\_split("/[\s,]+/", $afterisbn);

$isbn = $keywords[1];

$this->out($books[$id]->nodeValue . PHP\_EOL);

$this->out($livelib . $books[$id]->getAttribute('href') . PHP\_EOL);

$this->out($isbn . PHP\_EOL);

$checkisbn = $this->CollectionBooks->checkISBN($isbn);

if (!empty($checkisbn)) {

$this->out("Книга уже добавлена в таблицу");

} else {

$book = $this->CollectionBooks->newEntity();

$book->name = $books[$id]->nodeValue;

$book->isbn = $isbn;

$book->link = $livelib . $books[$id]->getAttribute('href');

$book->status = 'new';

$status = $this->CollectionBooks->addBook($book);

$this->out($status);

}

} else {

continue;

}

}

libxml\_use\_internal\_errors($internalErrors);

}

}

public function addCollectionBook() {

$id = 150;

$addBook = array();

$book = $this->CollectionBooks->find('all')->where(['id' => $id])->first();

if (empty($book)) {

$this->out('Запись не найдена');

} else {

if ($book['status'] == 'new') {

$pageOfBook = $book['link'];

$dom = new \DOMDocument("1.0", "UTF-8");

$internalErrors = libxml\_use\_internal\_errors(true);

$dom->loadHTMLFile($pageOfBook);

$finder = new \DOMXPath($dom);

$reviewCount = "menu-tabs-reviews";

$bookTitle = "name";

$bookAuthor = "book-author";

$bookCover = "main-image-book";

$isbn = "isbn";

$genres = "label-genre";

$description = "description";

$rating = "ratingValue";

$name = $finder->query(sprintf("//\*[contains(@itemprop, '%s')]", $bookTitle));

$count = $finder->query(sprintf("//a[contains(@class, '%s')]", $reviewCount));

$author = $finder->query(sprintf("//a[contains(@id, '%s')]", $bookAuthor));

$cover = $finder->query(sprintf("//img[contains(@id, '%s')]", $bookCover));

$isbn = $finder->query(sprintf("//span[contains(@itemprop, '%s')]", $isbn));

$genres = $finder->query(sprintf("//a[contains(@class, '%s')]", $genres));

$description = $finder->query(sprintf("//p[contains(@itemprop, '%s')]", $description));

$rating = $finder->query(sprintf("//span[contains(@itemprop, '%s')]", $rating));

$reviewCount = $count->item(0)->nodeValue;

$count = stristr($reviewCount, ' ');

$count = $count;

$count = trim($count);

$cover = $cover->item(0)->getAttribute('src');

/\* Собранные данные храним в массив, чтобы в дальнейшем сохранить в базу \*/

$addBook['name'] = $name->item(0)->nodeValue;

$addBook['authors'] = $author->item(0)->nodeValue;

$addBook['cover'] = $cover;

$addBook['genres'] = '';

$addBook['reviews\_count'] = $count;

$addBook['isbn'] = $isbn->item(0)->nodeValue;

$addBook['description'] = $description->item(0)->nodeValue;

$addBook['rating'] = $rating->item(0)->nodeValue;

if (empty($addBook['reviews\_count']))

$addBook['reviews\_count'] = 0;

$addBook['link'] = $book['link'];

foreach ($genres as $genre) {

$addBook['genres'] .= $genre->nodeValue . ", ";

}

$addBook['genres'] = trim($addBook['genres'], " ,");

$addBook['description'] = trim($addBook['description']);

$addBook['fromsite'] = 'livelib';

/\* Конец создания массива \*/

if ($addBook['reviews\_count'] > 1) {

$checkisbn = $this->Books->getBookByISBN($addBook['isbn']);

if (!empty($checkisbn)) {

$this->out("Книга уже добавлена в БД");

} else {

$added\_status = $this->Books->addBook($addBook);

if ($added\_status) {

$added = $this->CollectionBooks->changeBookStatus($book);

$this->out("Книга добавлена");

} else {

$this->out("Ошибка добавления");

}

}

} else {

$this->out("Книге оставлены очень мало рецензий");

}

libxml\_use\_internal\_errors($internalErrors);

} else {

$this->out("Книга уже добавлена в БД");

}

}

}

}