МИНОБРНАУКИ РОССИИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«САРАТОВСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ИМЕНИ Н. Г. ЧЕРНЫШЕВСКОГО»
Кафедра дискретной математики и информационных технологий

СОЗДАНИЕ СИСТЕМЫ ОТСЛЕЖИВАНИЯ ЗАДАЧ

АВТОРЕФЕРАТ БАКАЛАВРСКОЙ РАБОТЫ

студента 4 курса 421 группы
направления 09.03.01 — Информатика и вычислительная техника
факультета КНиИТ
Гатилова Дмитрия Сергеевича

Научный руководитель
ассистент _____ М. А. Боринос

Заведующий кафедрой
доцент, к. ф.-м. н. _____ Л. Б. Тяпаев

Саратов 2019

Введение. Современная коммерческая разработка программного обеспечения предполагает слаженную работу целых команд аналитиков, разработчиков, тестировщиков и менеджеров. Процессы разработки крупных проектов сложны и требуют контроля и удобных инструментов распределения задач. Для этого создаются различные системы управления проектами.

Системы отслеживания задач являются частным случаем систем управления проектами и ориентированы на совместную работу разработчиков и тестировщиков. Они позволяют распределять задачи между разработчиками, отправлять отчёты об ошибках и контролировать ход их исправления, расставлять приоритеты между задачами. Это позволяет значительно упростить разработку программного обеспечения.

Системы отслеживания задач актуальны в настоящее время, поскольку ни одна команда разработчиков не обходится без них. Система, удобная в эксплуатации, позволяет сократить время, затрачиваемое на создание продукта.

Целью данной работы является разработка системы отслеживания задач. Были выделены следующие задачи:

- Изучение современной методологии разработки программного обеспечения
- Рассмотрение существующих систем отслеживания задач
- Изучение принципов разработки web-приложений
- Выбор инструментов для реализации системы отслеживания задач и их изучение
- Программная реализация системы отслеживания задач, которая будет обеспечивать возможность работы с задачами и ошибками

В разделе 1 «Методология разработки Agile» рассматривается современная методология разработки Agile, принципы которой используются в большинстве современных IT-компаний.

В разделе 2 «Обзор существующих систем отслеживания задач» рассматриваются системы Jira и Bugzilla, выделяются их особенности, преимущества и недостатки.

В разделе 3 «Определение архитектуры разрабатываемой системы» содержится описание трёхуровневой архитектуры, на основе будет строиться разрабатываемое web-приложение.

В разделе 4 «Выбор и изучение инструментов для разработки» рассматривается фреймворк Spring, принципы, лежащие в его основе, его модули, а также среда разработки IntelliJ IDEA и система сборки Maven.

В разделе 5 «Реализация системы отслеживания задач» описывается создание базы данных MySQL, используемой приложением, функционал системы, структура и создание пакетов, из которых состоит приложение и его запуск.

1 Методология разработки Agile. Методология Agile описывает гибкий подход к разработке программного обеспечения. Она описана в документе Agile Manifesto [1], который включает в себя 4 ценности и 12 принципов. Ценности Agile:

- Люди и взаимодействие важнее процессов и инструментов
- Работающий продукт важнее исчерпывающей документации
- Сотрудничество с заказчиком важнее согласования условий контракта
- Готовность к изменениям важнее следования первоначальному плану

Принципы, значимые с точки зрения требований к разрабатываемой системе отслеживания задач:

- Простота — искусство минимизации лишней работы — крайне необходима
- Работающий продукт следует выпускать как можно чаще, с периодичностью

2 Обзор существующих систем отслеживания задач. Наиболее распространёнными в использовании системами отслеживания задач являются Bugzilla и Jira.

Основное понятие в Bugzilla – баг (от англ. ошибка, дефект). Он обладает следующими атрибутами:

- Статус – текущее состояние бага
- Резолюция – вердикт после рассмотрения бага
- Диаграмма – граф переходов с вершинами-состояниями
- Описание (включает в себя информацию о названии и версии продукта, важности, сроках решения, пользователях, ответственных за исправление бага, платформе и операционной системе)

Преимущества Bugzilla:

- В системе достаточно удобно реализована работа с большим количеством ошибок
- Есть интеграция с некоторыми сторонними сервисами
- Имеется возможность добавления к системе почтового сервера для отправки уведомлений по электронной почте
- Система является бесплатной

Недостатки Bugzilla:

- Относительно сложный процесс установки и настройки системы (для работы дополнительно требуется специальное программное и аппаратное обеспечение)
- Система ориентирована на работу с ошибками, и в ней отсутствуют функции по работе с задачами
- Относительно устаревший и не самый удобный пользовательский интерфейс

Основное понятие в системе Jira – задача. Задача содержит следующие поля:

- Type – тип
- Priority – приоритет
- Components – компоненты, с которыми связана задача
- Status – статус
- Resolution – резолюция
- Asignee – человек, ответственный за выполнение задачи
- Reporter – создатель задачи
- Watchers – люди, отслеживающие данную задачу
- Created – дата и время создания
- Updated – дата и время последнего изменения

- Resolved – дата и время решения
- Description – описание

Преимущества Jira:

- В системе имеется мощный функционал для работы с задачами и статистикой
- Есть интеграция со сторонними сервисами
- Широкие возможности для настройки системы
- Возможность работы с почтовым сервером

Недостатки Jira:

- Для нормальной работы система нуждается в сложной настройке
- Система распространяется на коммерческой основе, её стоимость напрямую зависит от количества сотрудников
- Из-за мощного функционала в интерфейсе иногда нелегко найти то, что необходимо
- В базовом варианте поля в задачах присутствуют в минимальном количестве, а процедура их добавления или удаления требует настройки

В зависимости от ситуации можно использовать ту или иную систему. Если требуется только функционал для работы с ошибками – можно использовать систему Bugzilla. Если требуется более расширенный функционал, лучше использовать систему Jira, если есть такая возможность.

2 Определение архитектуры разрабатываемой системы. В настоящее время для web-приложений чаще всего применяется трёхуровневая архитектура [2]. Она состоит из трёх частей [3]:

1. Уровень представления (presentation layer). На этом уровне реализуется пользовательский интерфейс.

2. Уровень бизнес-логики (business layer) содержит в себе логику обработки данных, полученных от клиента, а также он взаимодействует с базой данных и передаёт результат обработки на уровень представления.
3. Уровень доступа к данным (data access layer) – содержит модели данных и особые классы для взаимодействия с базой данных.

Данная архитектура отличается высокой надёжностью и масштабируемостью. Также приложения, построенные по трёхуровневой архитектуре, проще обслуживать и обновлять, так как при необходимости можно достаточно быстро заменить отдельные компоненты, не затрагивая остальные. Недостатками этой архитектуры считается сложность разработки и сложность развёртывания.

3 Выбор и изучение инструментов для разработки. Фреймворк [4] – программное обеспечение, облегчающее построение архитектуры приложений.

Фреймворк Spring содержит набор модулей для проектирования web-приложений.

Инверсия управления (Inversion of Control, IoC) [5] – один из принципов программирования, который представляет собой набор рекомендаций для реализации слабого связывания компонентов приложения. Принцип Ioc в фреймворке Spring реализован с помощью механизма внедрения зависимостей (Dependency Injection, DI).

Внедрение зависимостей – это механизм, в котором информация о взаимосвязях между объектами задаётся конфигурационными файлами и аннотациями -- специальными формами метаданных в Java.

Spring содержит несколько основных модулей:

- Core Container – базовый функционал, на котором основана работа фреймворка

- Data Access/Integration – функционал для работы с данными
- Web - набор инструментов для проектирования архитектуры web-приложений
- AOP – реализует концепцию аспектно-ориентированного программирования
- Aspects – функционал для реализации парадигм аспектно-ориентированного программирования и интеграция с AspectJ (расширение языка Java)
- Instrumentation – обеспечивает поддержку двух важных элементов фреймворка Spring – classloader (загрузка классов при обращении к ним) и class instrumentation (инструмент для управления исполнением классов Java)
- Messaging – обмен сообщениями в Spring-приложениях
- Test – функционал для тестирования

Для разработки используется среда IntelliJ IDEA 2018.2 Community Edition.

Для сборки приложения используется система сборки Maven, настраиваемая файлом pom.xml, а также плагин Spring Boot, облегчающий сборку Spring-приложений.

4 Реализация системы отслеживания задач. Приложение использует СУБД MySQL в качестве хранилища данных. База данных содержит следующие таблицы:

- Users – пользователи пользователях
- Roles – роли пользователей в приложении
- Contacts – контакты пользователей
- Projects – проекты
- Project_users – таблица, связывающая пользователей и проекты в отношении «многие ко многим»
- Project_roles – роли пользователей на проектах

- Tasks – информация о задачах
- Bugs - информация об ошибках

SQL-запрос создания базы данных представлен в приложении А.

В бакалаврской работе реализовано приложение, которое состоит из следующих пакетов:

- Пакет config – содержит файлы конфигурации приложения
- Пакет controller – содержит классы-контроллеры, которые служат для обработки запросов от клиента и получения результатов
- Пакет dao – содержит классы для доступа к данным
- Пакет entities – содержит классы-сущности, описывающие схему данных
- Пакет services – содержит классы, обеспечивающие сложную логику
- Пакет templates – содержит файлы, обеспечивающие отображение форм ввода и вывода данных в браузере

Полный код разработанного приложения представлен на приложенном диске.

Первое, что видит пользователь при входе в приложение – страница авторизации. Имеются 2 опции – ввести существующие данные или пройти регистрацию. После входа в систему на основной странице расположены ссылки на страницу контактов (содержит список контактов пользователя) и страницу проектов (содержит все проекты, к которым подключен пользователь).

Контакты – это список пользователей, которых при необходимости можно будет быстро подключать к проектам (эта опция доступна администраторам проектов). Контактам можно давать собственные имена на усмотрение пользователей.

На странице проектов можно создать собственный проект или работать с теми, к которым подключен пользователь. После перехода на страницу конкретного проекта, можно проводить следующие действия:

- Добавить пользователя к проекту (для администраторов)
- Удалить пользователя из проекта (для администраторов)
- Добавить задачу
- Добавить ошибку
- Посмотреть список всех задач
- Посмотреть список всех задач со статусом Open
- Посмотреть список всех ошибок
- Посмотреть список всех ошибок со статусом Open

Задача содержит следующий набор полей:

- Название
- Автор
- Пользователь, ответственный за выполнение задачи
- Дата и время создания
- Приоритет (число от 1 до 5, от меньшего к большему)
- Статус (возможные значения – open (над задачей никто не работает), in progress (задача в процессе исполнения), resolved (задача решена, и решение требует проверки) и closed (работа над задачей завершена));
- Описание

Ошибка содержит следующий набор полей:

- Название
- Дата и время создания
- Приоритет (число от 1 до 5, от меньшего к большему)
- Автор

- Пользователь, ответственный за выполнение задачи
- Статус (возможные значения - open (над задачей никто не работает), in progress (задача в процессе исполнения), resolved (задача решена, и решение требует проверки) и closed (работа над задачей завершена))
- Окружение (описание среды, в которой проводилось тестирование)
- Шаги для воспроизведения ошибки (описание шагов, по которым можно воспроизвести ошибку)
- Фактический результат (описание того, что при получается при воспроизведении шагов выше)
- Ожидаемый результат (ожидаемое поведение программы)
- Источник ожиданий
- Воспроизводимость (информация о том, как часто воспроизводится ошибка)
- Описание

Администраторы проекта и авторы задач (ошибок) могут удалять их. Если над задачей (ошибкой) никто не работает, пользователь, нажав кнопку Start working может обозначить, что он начал работать над задачей (ошибкой). Это приведёт к смене статуса с Open на In progress, а в поле автор появится имя и логин этого пользователя. Работающие над задачами (ошибками) по нажатию кнопки End working могут отметить, что они закончили выполнять работу над задачей (ошибкой). Тогда автор задачи (ошибки) или администратор проекта, нажав кнопку Close task (bug), могут подтвердить выполнение работы над задачей (ошибкой). Кнопки, отвечающие за соответствующие действия, появляются лишь в том случае, когда данный пользователь может их совершать.

Заключение. В ходе выполнения данной работы была изучена методология разработки Agile. Были рассмотренны самые популярные системы отслеживания задач – Bugzilla и Jira, выявлены их преимущества и недостатки. Также были рассмотрены основы построения web-приложений. Для разработки был выбран и изучен один из самых лучших и современных инструментов для разработки web-приложений – фреймворк Spring. Разработана система отслеживания задач на его основе.

Одним из самых важных преимуществ разработанной системы является её модульная архитектура, а также выбранные инструменты для реализации. Это преимущество позволит при необходимости без особых сложностей расширять систему новым функционалом и выпуском обновлений, что является достаточно большой проблемой для систем с многолетней историей разработки. Другими преимуществами разработанной системы является интуитивно понятный и лаконичный интерфейс, а также тот факт, что данная система не является коммерческой.

Все задачи данной работы были достигнуты.

Код разработанного приложения приложен на диске.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Agile-манифест разработки программного обеспечения [Электронный ресурс] // URL: https://agilemanifesto.org/iso/ru/manifesto.html (Дата обращения: 20.05.2019). Загл. с экрана. Яз. рус.
2. Трёхуровневая архитектура [Электронный ресурс] // Академик. URL: https://dic.academic.ru/dic.nsf/ruwiki/198141 (Дата обращения: 20.05.2019). Загл. с экрана. Яз. рус.
3. Многоуровневая архитектура [Электронный ресурс] // URL: https://metanit.com/sharp/mvc5/23.5.php (Дата обращения: 20.05.2019). Загл. с экрана. Яз. рус.
4. Фреймворки в веб-разработке [Электронный ресурс] // URL: https://web-creator.ru/articles/about_frameworks (Дата обращения: 20.05.2019). Загл. с экрана. Яз. рус.
5. Что такое Инверсия управления и Внедрение зависимостей (IoC & DI) [Электронный ресурс] // URL: https://shwanoff.ru/ioc-and-di/ (Дата обращения: 20.05.2019). Загл. с экрана. Яз. рус.