МИНОБРНАУКИ РОССИИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«САРАТОВСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ИМЕНИ Н. Г. ЧЕРНЫШЕВСКОГО»

Кафедра системного анализа и
автоматического управления

РАЗРАБОТКА ПРОГРАММНОГО КОМПЛЕКСА УЧЕТА И ОБРАБОТКИ ЗАЯВОК В СЛУЖБУ ПОДДЕРЖКИ ПОЛЬЗОВАТЕЛЕЙ

АВТОРЕФЕРАТ БАКАЛАВРСКОЙ РАБОТЫ

студента 5 курса 551 группы
направления 09.03.04 — Программная инженерия
факультета КНиИТ
Гаранина Александра Алексеевича

Научный руководитель
к. ф.-м. н., доцент _____ И. Е. Тананко

Заведующий кафедрой
к. ф.-м. н., доцент _____ И. Е. Тананко

Саратов 2023

ВВЕДЕНИЕ

Актуальность темы. В современном быстро меняющемся мире информационных технологий критически важно обеспечить эффективное взаимодействие между пользователями и службой поддержки. Простое и понятное средство связи, которое позволяет пользователям легко и быстро передавать свои заявки, а специалистам по поддержке — удовлетворять их потребности, становится ключевым элементом в обеспечении качественного сервиса. По этой причине многие организации ищут пути оптимизации процесса подачи и обработки заявок. В этом контексте актуальность разработки специализированного программного комплекса для учета и обработки заявок в службу поддержки пользователей становится очевидной.

Объект исследования — процесс обработки заявок в службе поддержки пользователей. Предмет исследования — методы и средства для автоматизации этого процесса.

В данной работе проведен обзор существующих решений и методов обработки заявок, исследованы их особенности и недостатки. Будет сформулированы требования к разрабатываемому комплексу, проведено его теоретическое и практическое проектирование, а также тестирование и анализ результатов.

Ожидается, что результатом работы станет качественный программный комплекс, который будет способствовать улучшению процесса подачи и обработки заявок в службу поддержки, а также повысит удовлетворенность пользователей обслуживанием. В конечном итоге это способствует увеличению эффективности работы всей организации.

Цель бакалаврской работы — разработка программного комплекса для учета и обработки заявок в службу поддержки пользователей, который будет отвечать современным требованиям и стандартам, обладать гибкостью настройки и высокой эффективностью.

Поставленная цель определила **следующие задачи**:

— Изучить и проанализировать существующие методы и модели обработки заявок в службе поддержки.

— Проанализировать существующие программные продукты для учета и обработки заявок в службу поддержки.

— Сформулировать требования к разрабатываемому программному ком-

плексу.

— Разработать архитектуру и структуру программного комплекса.
— Реализовать программный комплекс.
— Провести тестирование и анализ эффективности системы.

Методологические основы разработка программного комплекса учета и обработки заявок в службу поддержки пользователей представлены в работах М. Клепман, Р. Мартин, Э. Гамма, К. Старберг, Л. Криспин, В. Кулямин, К. Вигерс.

Практическая значимость бакалаврской работы. Практическая значимость данной работы заключается в возможности использования разработанного программного комплекса для повышения эффективности работы служб поддержки пользователей, уменьшения времени реакции на обращения пользователей и улучшения качества обслуживания.

Структура и объем работы. Бакалаврская работа состоит из введения, 3 разделов, заключения, списка использованных источников и цифрового носителя в качестве приложения. Общий объем работы — 54 страницы, из них 41 страница — основное содержание, включая 5 рисунков и 1 таблицу, список использованных источников информации — 23 наименования.

КРАТКОЕ СОДЕРЖАНИЕ РАБОТЫ

Первый раздел «Обзор подходов и анализ существующих решений» посвящен определению основных функциональных и технических требований разрабатываемого программного комплекса.

В подразделе 1.1 приведено изучение основных моделей и методов обработки заявок. Первой рассмотренной моделью является модель очереди. Модель очереди – один из самых распространенных подходов к обработке заявок. Следующей рассмотренной моделью является модель распределения. В этой модели заявки распределяются между исполнителями (или ресурсами обработки) на основе различных критериев. Также популярной моделью является модель работы по тикетам. В этой модели каждая заявка представляет собой «тикет», который содержит всю необходимую информацию для обработки заявки (включая описание заявки, информацию о пользователе, историю обработки и т.д.). Последней рассмотренной моделью является модель обратной связи. Эта модель предполагает обратную связь от пользователей по поводу обработки их заявок.

Подраздел 1.2 посвящен анализу существующих систем учета и обработки заявок в службу поддержки. Были рассмотрены различные варианты программного обеспечения, такие как Zendesk, Freshdesk и другие. Исследование включало сравнение функционала этих систем, их масштабируемости и гибкости для потребностей организаций. Были изучены особенности учета заявок, обработки и трекинга их статусов. Это исследование помогло определить необходимый функционал для разрабатываемой системы и выявить потенциальные улучшения.

В подразделе 1.3 приводится идентификация недостатков текущих систем и поиск возможностей для их улучшения, что является важной частью разработки программного комплекса учета и обработки заявок в службу поддержки пользователей. Путем анализа аналогов и существующих решений были определены области, в которых можно достичь значительного улучшения и повысить эффективность работы системы. Среди ключевых направлений для улучшения выделяются следующие: упрощение и оптимизация пользовательского интерфейса, расширение способов подачи заявок через интеграцию с различными коммуникационными каналами, улучшение механизмов отслеживания статуса заявок, обеспечение гибкой и автоматической интеграции с другими системами, а также увеличение гибкости настроек для лучшей адаптации системы под специфические бизнес-процессы и требования организации.

В подразделе 1.4 приводится исследование предпочтений пользователей в использовании различных способов заведения заявок, для чего была проведена специальная исследовательская работа. Целью исследования было определить, какие способы предпочитают использовать пользователи для заведения заявок и какие факторы влияют на их выбор. Одним из ключевых выводов исследования стало то, что большинство пользователей предпочитают использовать для заведения заявок мессенджеры. В частности, 70% респондентов отметили, что они предпочитают использовать Telegram для этой цели. Основными причинами такого выбора стали удобство использования, быстрота обработки заявок и возможность мгновенного получения обратной связи.

Второй раздел «Теоретическое проектирование программного комплекса» посвящен описанию основных функциональных и технических

требований к разработанному программному комплексу обработки заявок в техническую поддержку пользователей, а также построению архитектуры данного комплекса, описаны основные модули и взаимодействие между ними. Дополнительно описаны основные подходы к проектированию.

В подразделе 2.1 описываются общие функциональные требования для программного комплекса учета и обработки заявок в службу поддержки. Основываясь на анализе в предыдущем разделе, комплекс включает в себя модуль обработки заявок, телеграм-бота для приема заявок от пользователей и модуль интеграции с системой Jira для управления заявками. Требования к программному комплексу сформулированы на основе предполагаемых потребностей пользователей и технических особенностей системы:

1. Программный комплекс должен иметь модуль учета и обработки заявок, обеспечивающий функциональность для создания, просмотра и обновления статусов заявок.
2. Программный комплекс должен включать телеграм-бота, обеспечивающего возможность пользователей отправлять заявки и получать уведомления о статусе их заявок через платформу Telegram.
3. Программный комплекс должен включать модуль интеграции с Jira, позволяющий передавать информацию о заявках в систему Jira для дальнейшей обработки и управления проектами.
4. Программный комплекс должен обеспечивать высокую производительность и надежность.

Подраздел 2.2 посвящен описанию общего архитектурного подхода к разработке комплекса. Была проведена декомпозиция архитектуры для данного комплекса. В результате декомпозиции было принято решение разделить систему на три основных модуля: модуль учета и обработки заявок, модуль Telegram-бота и модуль интеграции с Jira. Такое решение позволяет упростить разработку и тестирование системы, а также увеличить ее надежность и гибкость. Для каждого из модулей был разработан свой подход к архитектуре.

Модуль учета и обработки заявок был организован в виде веб-сервиса на базе ASP.NET Core, предоставляющего RESTful API для взаимодействия с остальными модулями. Для работы с базой данных был выбран подход, основанный на использовании ORM (Object-Relational Mapping), что позволило

облегчить работу с данными и увеличить скорость разработки.

Модуль Telegram-бота был спроектирован с использованием библиотеки Telegram.Bot, предоставляющей возможности для работы с Telegram Bot API. В этом модуле был использован принцип конечного автомата для управления потоком диалогов и обработки сообщений.

Модуль интеграции с Jira основывается на использовании Jira REST API. Для упрощения работы с этим API была использована библиотека Atlassian SDK.

Для хранения данных была разработана структура базы данных. Были учтены все основные сущности системы, их связи и зависимости. Структура базы данных была разработана с использованием подхода, основанного на нормализации данных, что позволяет уменьшить дублирование данных и увеличить эффективность работы с данными.

Подраздел 2.3 посвящен формулированию основной структуры разработанного программного комплекса и функциональным требованиям, которые будут использоваться при дальнейшей практической реализации. Общая схема комплекса с указанием модулей и основных потоков данных приведена на рисунке 1.

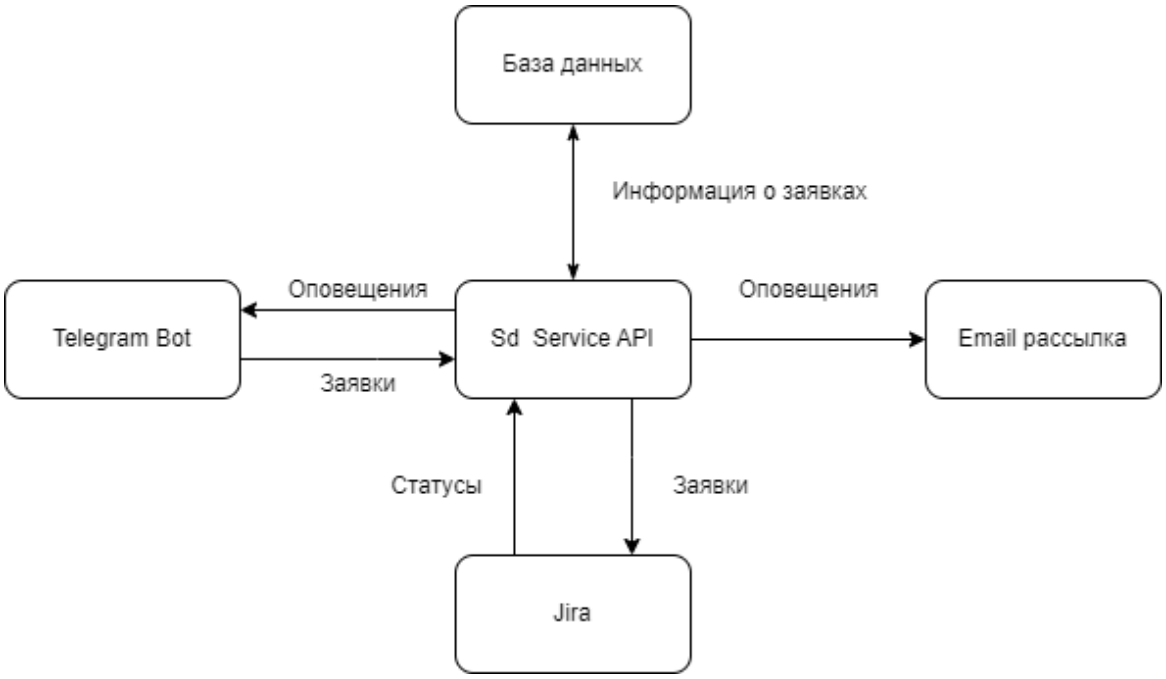

Рисунок 1 – Общая схема комплекса приложений

Также в данном подразделе описаны основные архитектурные и поведенческие паттерны проектирования использованные при разработке, такие

как Dependency Injection, Repository, Chain of Responsibility, Observer, Builder, Singleton. Данные паттерны проектирования были выбраны и применены с целью достижения высокой модульности, гибкости, повторного использования кода и обеспечения расширяемости и поддерживаемости системы. Каждый паттерн решает определенные проблемы проектирования и способствует разделению ответственностей, улучшению архитектуры и упрощению разработки.

Третий раздел «Практическая реализация программного комплекса» посвящен описанию выбора технологической средств разработки, непосредственно самой разработке всех необходимых модулей, а также тестированию программных модулей и комплекса в целом.

В подразделе 3.1 описываются основные технологические средства используемые при разработки, включающие в себя платформу разработки, основной язык программирования, используемый в разработке, а также обоснование и выбор базы данных. Исходя из перечисленных в данном подразделе особенностей рассмотренных технических средств был выбран C# в качестве основного языка программирования, а также база данных PostgreSQL.

В подразделе 3.2 описана разработка основных модулей системы.

Модуль учета и обработки заявок является центральной частью программного комплекса. Этот модуль обрабатывает заявки от пользователей, изменяет их статусы и сохраняет информацию в базе данных.

В данном модуле осуществляется учет и обработка заявок пользователей. Для этого используется Entity Framework Core для взаимодействия с базой данных PostgreSQL.

Первым этапом был этап создания базы данных. Используя PostgreSQL и ORM-технологию Entity Framework Core, была разработана структура базы данных. Она включает в себя таблицы: «Users» для хранения информации о пользователях, «Tickets» для хранения информации о заявках и «TicketStatuses» для отслеживания изменений статусов заявок.

Для разработки REST API модуля обработки заявок был использован фреймворк ASP.NET Core, который предоставляет мощные инструменты для создания веб-приложений и API.

Основными функциями, которые были необходимы в этом модуле, были создание новой заявки, получение статуса заявки и обновление статуса

заявки.

В ходе разработки модуля обработки заявок Swagger использовался для описания REST API модуля, включая методы для создания, получения, обновления и удаления заявок. Swagger позволил нам быстро и точно документировать все эти методы, предоставляя подробную информацию о каждом из них, включая параметры запросов, форматы ответов и коды состояний HTTP.

Следующим разработанным и описанным в данном подразделе модулем является модуль телеграм-бота. Бот реализован с использованием библиотеки Telegram.Bot. Он получает заявки от пользователей и преобразует их в формат, подходящий для модуля учета и обработки заявок. Также он отслеживает статус заявок и отправляет уведомления пользователям через Telegram.

Разработка телеграм бота для системы включала в себя несколько ключевых аспектов: настройку и регистрацию бота, создание команд для взаимодействия с пользователем и интеграцию с внешними системами.

Заключительным разработанным и описанным в подразделе модулем является модуль интеграции с Jira. Для внедрения модуля интеграции с Jira, первым делом было необходимо изучить предлагаемый Jira API. Исследование включало изучение архитектуры API, типов запросов, методов авторизации, структуры данных запросов и ответов, а также обработки ошибок и ограничений. Модуль интеграции с Jira был необходим для создания, обновления и отслеживания заявок прямо из разрабатываемой системы. Для этого были использованы Jira REST API и библиотеку Atlassian.SDK для .NET.

Подраздел 3.3 содержит описание проводимого тестирования в ходе разработки, а также по её окончании. Для тестирования модуля обработки заявок в работе использовали модульные и интеграционные тесты. Эти виды тестирования позволяют обеспечить надежность, функциональность и совместимость модуля с другими компонентами системы. Модульное тестирование выполняется для проверки отдельных компонентов модуля и их правильной работы в изоляции от других компонентов системы.

Интеграционные тесты проводились с использованием симуляторов и моков, чтобы эмулировать работу других компонентов системы и изолировать модуль обработки заявок для более точного тестирования его функцио-

нальности и взаимодействия.

Итоговое модульное и интеграционное тестирование модуля обработки заявок помогло выявить и исправить ошибки, проверить корректность работы функциональности и обеспечить надежность его взаимодействия с другими компонентами системы. Тестирование позволило убедиться в правильной работе модуля и готовности его к использованию в реальной среде службы поддержки пользователей.

ЗАКЛЮЧЕНИЕ

В ходе данной работы был разработан программный комплекс для учета и обработки заявок в службу поддержки пользователей.

В процессе работы было проведено обширное исследование текущего рынка программных продуктов для обработки заявок. Анализ аналогов позволил выявить ряд универсальных требований, а также своеобразные преимущества и недостатки каждой системы. Основываясь на данных исследований, были сформулированы требования к разрабатываемому программному комплексу.

Ключевыми особенностями разработанного программного комплекса стали: модульная архитектура, использование телеграм-бота для упрощения процесса подачи заявок пользователями, интеграция с системой управления проектами Jira, а также учет предпочтений пользователей в процессе обработки заявок.

Для реализации программного комплекса был выбран технологический стек, включающий в себя язык программирования C# и базу данных Postgres. Это обеспечило надежность, гибкость и масштабируемость разработанного решения.

В ходе работы над программным комплексом была разработана и внедрена сложная инфраструктура, включая базу данных для хранения информации о заявках, сервер для обработки заявок, а также телеграм-бота и модуль интеграции с Jira.

Таким образом, в результате данной работы был создан полноценный программный продукт, который способен обеспечить эффективный процесс подачи, обработки и отслеживания заявок в службу поддержки пользователей. Он может стать ценным инструментом для команд поддержки любых

организаций, которые стремятся повысить свою производительность и качество обслуживания клиентов.

Однако, стоит отметить, что разработка программного комплекса – это непрерывный процесс. В дальнейшем планируется добавление новых функций, доработка существующих модулей, улучшение производительности и надежности системы, а также увеличение числа интеграций с другими популярными сервисами.

Основные источники информации:

1. Клепман, М. Высоконагруженные приложения. Программирование, масштабирование, поддержка. / М. Клепман. — СПб.: Питер, 2018. — 640 с.
2. Мартин Р. Чистый код: создание, анализ и рефакторинг. / Р. Мартин. — Библиотека программиста. — СПб.: Питер, 2022. — 464 с.
3. Гамма Э. Паттерны проектирования: Элементы повторно используемого объектно-ориентированного программного обеспечения. / Э. Гамма. — М.: Вильямс, 2001. — 448 с.
4. Старберг К. Принципы, паттерны и практики создания гибкой архитектуры. / К. Старберг. — М.: ДМК Пресс, 2012. — 768 с.
5. Криспин Л. Гибкое тестирование: практическое руководство для тестировщиков ПО и гибких команд. / Л. Криспин. — М.: Вильямс, 2010. — 464 с.
6. Кулямин В. Технологии программирования. Компонентный подход. / В. Кулямин. — СПб.: Просвещение, 2015. — 463 с.
7. Вигерс К. Разработка требований к программному обеспечению. / К. Вигерс. — СПб.: БХВ-Петербург, 2016. — 320 с.