

СОДЕРЖАНИЕ

ВВЕДЕНИЕ	3
1 ОПИСАНИЕ ПРЕДМЕТНОЙ ОБЛАСТИ.....	4
2 КРАТКАЯ ХАРАКТЕРИСТИКА СРЕДСТВ РАЗРАБОТКИ.....	5
2.1 Язык программирования и средства создания пользовательского интерфейса	5
2.2 Средства получения исполняемого кода, средства отладчики исполняемого кода и средства тестирования программ.....	5
2.3 Средства управления базами данных	6
2.4 Средства запуска готового проекта	7
2.5 Средства проектирования приложения	7
3 ПРОЕКТИРОВАНИЕ ПРОГРАММНОГО СРЕДСТВА.....	9
3.1 Функциональные требования	9
3.2 IDEF0-диаграммы и ее описание	10
3.3 Диаграмма «сущность-связь».....	10
3.4 Модель базы данных	11
3.5 Диаграмма вариантов использования и ее описание	11
3.6 Диаграммы последовательностей	13
3.7 Кооперативные диаграммы.....	14
3.8 Диаграммы активности	14
3.9 Диаграмма классов	19
3.10 Диаграмма размещения.....	19
4 ОПИСАНИЕ ПРОГРАММНОГО КОДА	21
5 ТЕСТИРОВАНИЕ ПРОГРАММНОГО СРЕДСТВА	23
6 РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ	29
ЗАКЛЮЧЕНИЕ.....	30
СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ	31
ПРИЛОЖЕНИЕ А.....	32
ПРИЛОЖЕНИЕ Б	34

ВВЕДЕНИЕ

Согласно заданию, необходимо разработать программное средство для просмотра сведений о сотрудниках предприятия с возможностью выдачи заданий и контроля их выполнения сотрудниками предприятия.

Целью проекта является проектирование и разработка автоматизированной системы учета выданных заданий. Программное средство позволяет: хранить и просматривать личные данные сотрудников, совершать сортировку; удаленно выдавать задания; контролировать работу сотрудников.

Поставленная задача актуальна, поскольку на многих предприятиях работают несколько сотен сотрудников. Учитывать вручную и контролировать процесс работы предприятия очень проблематично. Поэтому решение данной задачи будет очень востребовано на крупных предприятиях с большим количеством сотрудников. Причем, если эти же сотрудники работают удаленно, то востребованность данного проекта многократно возрастает.

Перечень средств разработки:

- Средства проектирования приложения: IBM Rational Rose Enterprise Edition, Microsoft Visio 2010;
- Язык программирования: Java8;
- Средства создания пользовательского интерфейса: Swing;
- Средства получения исполняемого кода: IntelliJ IDEA 2017.3.5 x64;
- Средства управления базами данных: MySQL;
- Средства отладчики исполняемого кода: IntelliJ IDEA 2017.3.5 x64;
- Средства запуска готового проекта: JVM;
- Средства тестирования программ: IntelliJ IDEA 2017.3.5 x64.

1 ОПИСАНИЕ ПРЕДМЕТНОЙ ОБЛАСТИ

В данное время очень много предприятий с большим количеством сотрудников, при этом руководителей меньше. И в процессе контроля выполнения заданий, очень большая нагрузка ложится на руководителей предприятия, что приводит к ошибкам. Для того что бы избежать всех неудобств и будет разработана программное средство для большого количества сотрудников и контроля выполнения заданий.

Результат проекта – технологическое программное средство для автоматизированного управления функциями предприятия.

Уровень подготовленности и обеспеченности ИТ целевой аудитории: целевая аудитория должна уметь обращаться с компьютером в целом. На предприятии необходимо наличие сервера, локальной сети, подключение к сети интернет.

Обоснование выбора средств разработки и функционирования проекта:

- Приложение должно обладать пользовательским графическим интерфейсом. Для создания графического интерфейса используется Swing библиотека.
- Поскольку программное средство предназначено для работы нескольких сотен людей, то необходимо обеспечить безопасное и стабильное хранения данных. По этой причине используется СУБД.
- Проект компилируемый с точки зрения разработки. Для того, чтобы запустить программу, ее необходимо скомпилировать. Интерпретируемый с точки зрения процессора, т.к. скомпилированный код (байт-код) не может непосредственно выполняться процессором. Нужен специальный интерпретатор (JRE, Java Runtime Environment), который будет преобразовывать байт-код в коды процессора. У клиента (сотрудника предприятия) должны быть установлены JVM, СУБД MySQL и разработанное программное средство.

2 КРАТКАЯ ХАРАКТЕРИСТИКА СРЕДСТВ РАЗРАБОТКИ

2.1 Язык программирования и средства создания пользовательского интерфейса

Java – сильно типизированный объектно-ориентированный язык программирования, позволяющий создавать программы, которые могут исполняться на любой платформе без каких-либо доработок (кроссплатформенность) с добавлением garbage collector'a - автоматического сборщика "мусора" (механизм освобождения памяти). Также известно, что Java ориентирована на Internet, и самое распространенное ее применение - небольшие программы, апплеты, которые запускаются в браузере и являются частью HTML-страниц. С помощью Java web-страницу можно наполнить не только обычным текстом, но и динамическими элементами. С его помощью можно создать сложный пользовательский интерфейс.

Java-платформа обладает следующими преимуществами:

- Переносимость, или кроссплатформенность;
- Объектная ориентированность, создана эффективная объектная модель;
- Встроенная и прозрачная модель безопасности;
- Ориентация на Internet-задачи, сетевые распределенные приложения;
- Динамичность, легкость развития и добавления новых возможностей;

Swing – библиотека для создания графического интерфейса для программ на языке Java. Он содержит ряд графических компонентов (англ. Swing widgets), таких как кнопки, поля ввода, таблицы и т. д. Swing относится к библиотеке классов JFC, которая представляет собой набор библиотек для разработки графических оболочек.

2.2 Средства получения исполняемого кода, средства отладки исполняемого кода и средства тестирования программ

IntelliJ IDEA — интегрированная среда разработки программного обеспечения для многих языков программирования, в частности Java, JavaScript, Python, разработанная компанией JetBrains. Обладает широким набором интегрированных инструментов для рефакторинга, которые позволяют программистам быстро реорганизовывать исходные тексты программ.

IntelliJ IDEA предоставляет интегрированный инструментарий для разработки графического пользовательского интерфейса. Среди прочих возможностей, среда хорошо совместима со многими популярными свободными инструментами разработчиков, такими как CVS, Subversion, Apache Ant, Maven и JUnit.

Среда доступна в двух редакциях: Community Edition и Ultimate Edition. Community Edition является полностью свободной версией, доступной под лицензией Apache 2.0, в ней реализована полная поддержка Java SE, Groovy, Scala, а также интеграция с наиболее популярными системами управления версиями. В редакции Ultimate Edition, доступной под коммерческой лицензией, реализована поддержка Java EE, UML-диаграмм, подсчёт покрытия кода, а также поддержка других систем управления версиями, языков и фреймворков.

Также IntelliJ IDEA обладает встроенным отладчиком исполняемого кода и средствами тестирования программ.

2.3 Средства управления базами данных

MySQL – свободная реляционная система управления базами данных. MySQL является решением для малых и средних приложений. Обычно MySQL используется в качестве сервера, к которому обращаются локальные или удалённые клиенты, однако в дистрибутив входит библиотека внутреннего сервера, позволяющая включать MySQL в автономные программы.

Гибкость СУБД MySQL обеспечивается поддержкой большого количества типов таблиц: пользователи могут выбрать как таблицы типа MyISAM, поддерживающие полнотекстовый поиск, так и таблицы InnoDB, поддерживающие транзакции на уровне отдельных записей. Более того, СУБД MySQL поставляется со специальным типом таблиц EXAMPLE, демонстрирующим принципы создания новых типов таблиц. Благодаря открытой архитектуре и GPL-лицензированию, в СУБД MySQL постоянно появляются новые типы таблиц.

MySQL имеет API для языков Delphi, C, C++, Эйфель, Java, Лисп, Perl, PHP, Python, Ruby, Smalltalk, Компонентный Паскаль и Tcl, библиотеки для языков платформы .NET, а также обеспечивает поддержку для ODBC посредством ODBC-драйвера MyODBC.

MyODBC представляет собой драйвер ODBC (2.50) уровня 0 (с некоторыми возможностями уровней 1 и 2) для подсоединения совместимого с ODBC приложения к MySQL. MyODBC работает на всех системах Microsoft Windows и на большинстве платформ Unix.

2.4 Средства запуска готового проекта

Java Virtual Machine (сокращенно Java VM, JVM) — виртуальная машина Java — основная часть исполняющей системы Java, так называемой Java Runtime Environment (JRE). Виртуальная машина Java исполняет байт-код Java, предварительно созданный из исходного текста Java-программы компилятором Java (javac). JVM может также использоваться для выполнения программ, написанных на других языках программирования. Например, исходный код на языке Ada может быть откомпилирован в байт-код Java, который затем может выполняться с помощью JVM.

JVM является ключевым компонентом платформы Java. Так как виртуальные машины Java доступны для многих аппаратных и программных платформ, Java может рассматриваться и как связующее программное обеспечение, и как самостоятельная платформа. Использование одного байт-кода для многих платформ позволяет описать Java как «скомпилировано однажды, запускается везде» (compile once, run anywhere).

Виртуальные машины Java обычно содержат Интерпретатор байт-кода, однако, для повышения производительности во многих машинах также применяется JIT-компиляция часто исполняемых фрагментов байт-кода в машинный код.

2.5 Средства проектирования приложения

CASE-средство IBM Rational Rose со времени своего появления претерпело серьезную эволюцию, и в настоящее время представляет собой современный интегрированный инструментарий для проектирования архитектуры, анализа, моделирования и разработки программных систем. Именно в IBM Rational Rose язык UML стал базовой технологией визуализации и разработки программных систем, что определило популярность и стратегическую перспективность этого инструментария.

Наиболее характерные функциональные особенности этой программы заключаются в следующем:

- интеграция с MS Visual Studio 6, которая включает поддержку на уровне прямой и обратной генерации кодов и диаграмм Visual Basic и Visual C++ с использованием ATL (Microsoft Active Template Library), Web-Классов, DHTML и протоколов доступа к различным базам данных;
- непосредственная работа (инжиниринг и реинжиниринг) с исполняемыми модулями и библиотеками форматов EXE, DLL, TLB, OCX.

- поддержка технологий MTS (Microsoft Transaction Server) и ADO (ActiveX Data Objects) на уровне шаблонов и исходного кода, а также элементов технологии Microsoft - COM+ (DCOM);
- полная поддержка компонентов CORBA и J2EE, включая реализацию технологии компонентной разработки приложений CBD (Component-Based Development), языка определения интерфейса IDL (Interface Definition Language) и языка определения данных DDL (Data Definition Language);
- полная поддержка среды разработки Java-приложений, включая прямую и обратную генерацию классов Java формата JAR, а также работу с файлами формата CAB и ZIP.

Microsoft Visio — векторный графический редактор, редактор диаграмм и блок-схем для Windows. Основным назначением Visio является визуализация данных или, проще говоря, изготовление практически любых схем, иллюстраций и диаграмм, необходимых для оформления деловой документации.

Используя эту программу, вы можете отобразить ход выполнения проекта, подготовить план здания или, например, макет веб-страницы. Содержание иллюстраций зависит только от вашего воображения. При этом вы сможете гибко настраивать внешний вид элементов, что позволит не только использовать функциональную окраску частей иллюстрации, но и получить опрятный и привлекательный документ.

3 ПРОЕКТИРОВАНИЕ ПРОГРАММНОГО СРЕДСТВА

3.1 Функциональные требования

В результате должно быть разработано технологическое программное средство для автоматизированного управления функциями предприятия, позволяющий: хранить и просматривать личные данные сотрудников, совершать сортировку; удаленно выдавать задания; контролировать работу сотрудников.

Программное средство включает в себя роли с различными правами доступа: руководитель (полный доступ ко всему функционалу программного средства) и сотрудника (ограниченный доступ к функционалу программного средства).

Роль с правами руководителя должна обеспечивать выполнение следующих функциональных возможностей:

1. Управление учетными записями пользователей:

- просмотр всех учетных записей пользователей;
- сортировка учетных записей пользователей;
- создание новой учетной записи пользователя;
- редактирование учетной записи пользователя;
- удаление учетной записи пользователя;

2. Работа с выданными заданиями сотруднику:

- просмотр выданных заданий учетной записи пользователя;
- сортировка выданных заданий учетной записи пользователя;
- создание и выдача задания учетной записи пользователя;
- редактирование выданного задания учетной записи пользователя;
- удаление выданного задания учетной записи пользователя;

Роль с правами сотрудника должна обеспечивать выполнение следующих функциональных возможностей:

- просмотр своих заданий;
- сортировка своих заданий;
- отчет о выполнении своего задания.

Предусмотреть авторизацию пользователя перед началом работы. После прохождения авторизации в зависимости от роли пользователя выводиться:

- Руководитель: группа кнопок для работы с таблицей учетных записей пользователей и таблицей выданных заданий, при нажатии на которые

пользователь переходит к выполнению нужной операции, таблица учетных записей пользователей, таблица выданных заданий;

- Сотрудник: группа кнопок для работы с таблицей выданных заданий, при нажатии на которые пользователь переходит к выполнению нужной операции, таблица выданных заданий.

3.2 IDEF0-диаграммы и ее описание

Цель модели: определить основные этапы процесса выдачи и выполнения задания, их влияние друг на друга и на результаты выполнения задания с целью продуктивной работы предприятия.

Претенденты на точку зрения: руководитель, сотрудник. С учетом цели модели предпочтение следует отдать точке зрения руководителя, так как она наиболее полно охватывает все этапы задания и только с этой точки зрения можно показать взаимосвязи между отдельными этапами и обязанности сотрудников.

Субъектом моделирования является сама система учета выданных заданий.

IDEF0-диаграммы представлены на рисунках Б.1 и Б.2 в приложении Б.

3.3 Диаграмма «сущность-связь»

Для проектирования базы данных воспользуемся методом моделирования "сущность-связь" (ER modeling), который дает абстрактную модель предметной области, используя следующие основные понятия: сущности (entities), взаимосвязи (relationships) между сущностями и атрибуты (attributes) для представления свойств сущностей и взаимосвязей.

Выделим сущности: сотрудник (идентификатор, фамилия, имя, отчество, логин, пароль, дата рождения, город, страна), задание (идентификатор, идентификатор сотрудника, наименование, текст задания, дата выдачи задания, дата завершения выполнения задания, дата выполнения, флаг о выполнении задания, флаг о проверке выполненного задания).

Сотрудник способен выполнять несколько заданий, из этого следует, что сущности имеют связь один ко многим. Диаграмма «сущность-связь» представлена на рисунке 1.



Рисунок 1 – Диаграмма «сущность-связь»

3.4 Модель базы данных

В разделе мы выяснили, что связь между сущностями является «один-много». Теперь выделим атрибуты сущностей. Информация о сотруднике: фамилия, имя, отчество, логин, пароль, дата рождения, город, страна. Информация о задании: кто выполняет, наименование, текст задания, дата выдачи задания, дата завершения выполнения задания, дата выполнения, выполненное/невыполненное, проверенное/непроверенное. Модель базы данных представлена на рисунке 2.

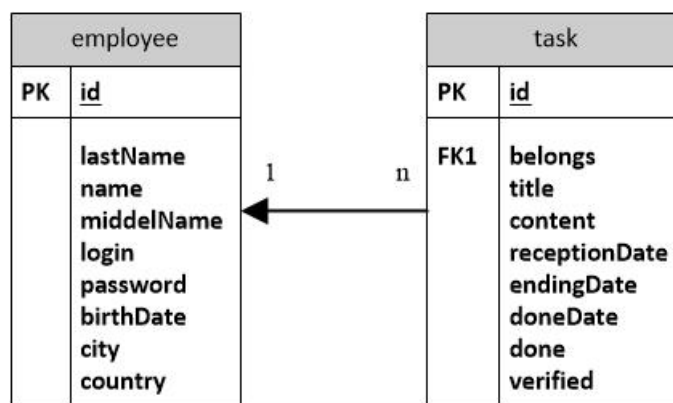


Рисунок 2 – Модель базы данных

3.5 Диаграмма вариантов использования и ее описание

Диаграмма вариантов использования в данном курсовом проекте предоставлена на рисунке 3. На диаграмме, действующие субъекты предоставляются человечками, варианты использования – эллипсами. Прямоугольная рамка окружает все варианты использования, оставляя за своими пределами действующих субъектов, формируя границу системы. Внутри границы системы находится программное обеспечение, которое необходимо создать.

Начало программы: в браузере отображается приветственное окно с формой для ввода логина и пароля.

Актер «Руководитель» имеет 2 варианта использования программного средства:

- Управление учетными записями пользователей;
- Работа с выданными заданиями.

Управление учетными записями пользователей служит для просмотра, сортировки, поиска, создания, редактирования, удаления данных учетных записей пользователей.

Вариант использования «Управление учетными записями пользователей» имеет 5 вариантов использования программного средства:

- Просмотр всех учетных записей пользователей;
- Сортировка всех учетных записей пользователей;
- Создание учетной записи пользователя;
- Редактирование учетной записи пользователя;
- Удаление учетной записи пользователя.

Работа с выданными заданиями служит для просмотра , сортировки, поиска, создания и выдачи, редактирования, удаления данных о заданиях. Вариант использования «Работа с выданными заданиями» имеет 5 вариантов использования программного средства:

- Просмотр выданных заданий учетной записи пользователя;
- Сортировка выданных заданий учетной записи пользователя;
- Создание и выдача задания учетной записи пользователя;
- Редактирование выданного задания учетной записи пользователя;
- Удаление выданного задания учетной записи пользователя.

Актер «Сотрудник» имеет 3 варианта использования программного средства:

- Просмотр своих заданий;
- Сортировка своих заданий;
- Отчет о выполнении своего заданий.

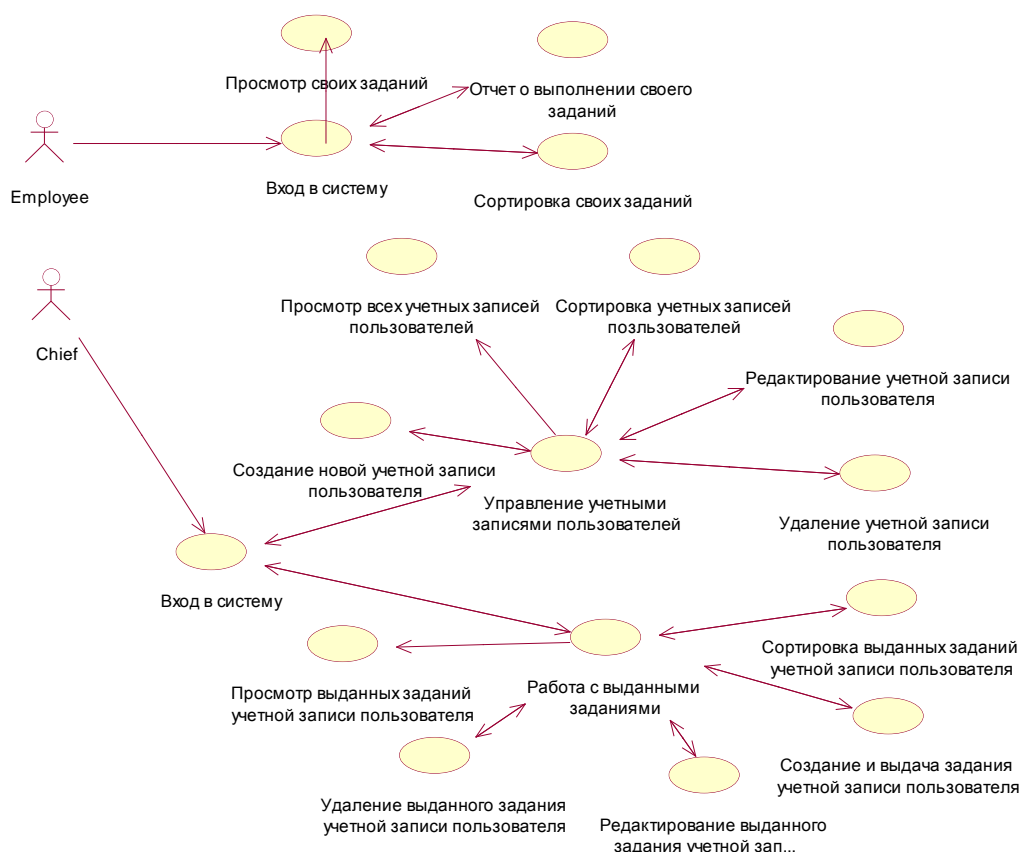


Рисунок 3 – Диаграмма вариантов использования программного средства

3.6 Диаграммы последовательностей

Взаимодействие объектов в системе происходит посредством приема и передачи сообщений объектами-клиентами и обработки этих сообщений объектами-серверами. При этом в разных ситуациях одни и те же объекты могут выступать и в качестве клиентов, и в качестве серверов.

Данный тип диаграмм позволяет отразить последовательность передачи сообщений между объектами. При этом, этот тип диаграммы не акцентирует внимание на конкретном взаимодействии, главный акцент уделяется последовательности приема/передачи сообщений.

Диаграммы последовательностей представлены на листах 1, 3, 5, 7, 9, 11, 13, 15, 17, 19, 21, 23, 25, 27.

3.7 Кооперативные диаграммы

Если диаграмма последовательности действий служит для визуализации временных аспектов взаимодействия, то диаграмма кооперации предназначена для спецификации структурных аспектов взаимодействия. Главная особенность диаграммы кооперации заключается в возможности графически представить не только последовательность взаимодействия, но и все структурные отношения между объектами, участвующими в этом взаимодействии.

Кооперативные диаграммы представлены на листах 2, 4, 6, 8, 10, 12, 14, 16, 18, 20, 22, 24, 26, 28.

3.8 Диаграммы активности

Смоделируем все варианты использования. Для этого представим их в виде диаграмм активностей. Диаграммы активностей представлены на рисунках 4 - 9.

Поскольку действия остальных вариантов использования схожи, то с целью уменьшения количества повторений, остальные диаграммы не будут представлены в этом курсовом проекте.

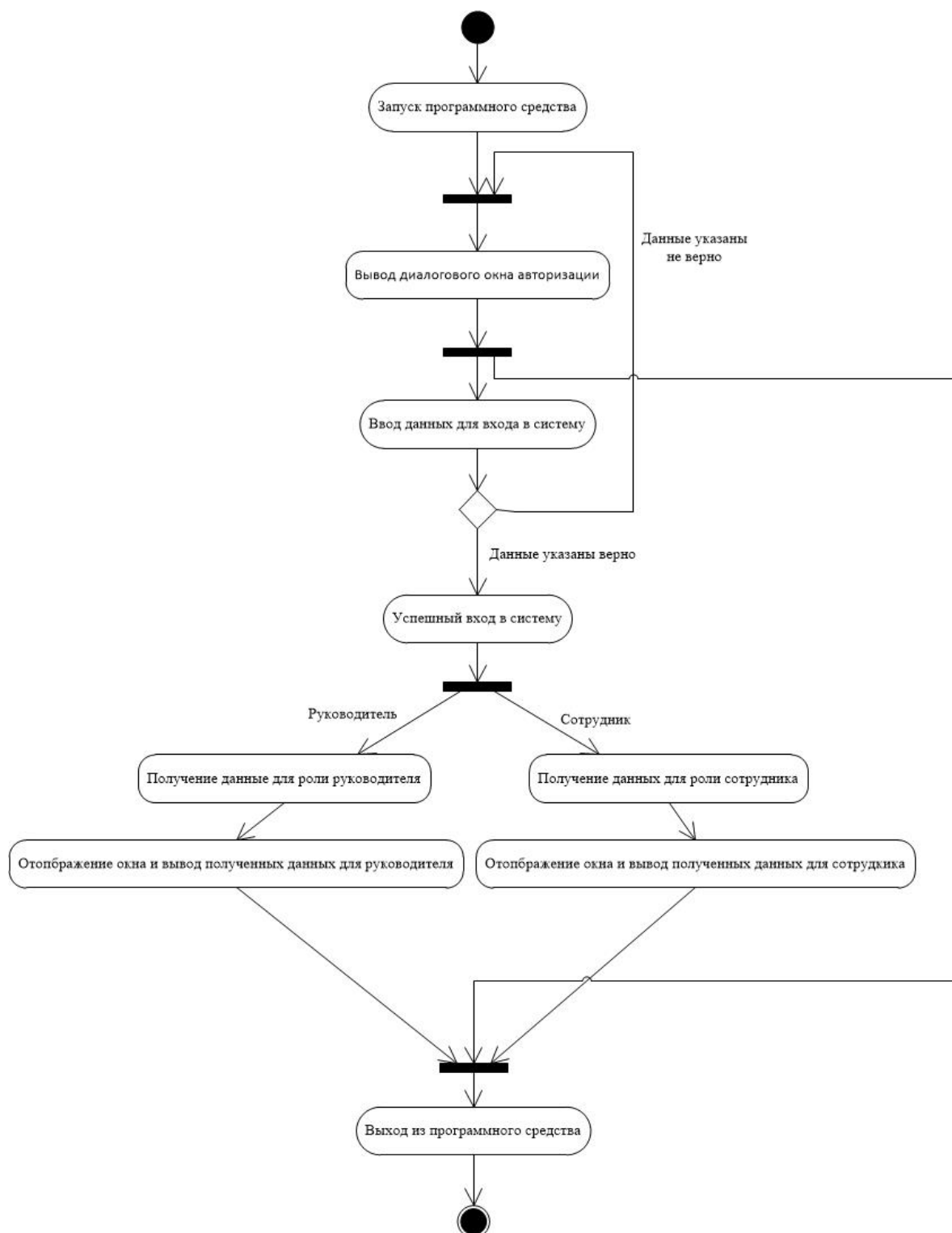


Рисунок 4 – Диаграмма запуска программного средства и завершения его работы



Рисунок 5 – Диаграмма добавления данных о новом сотруднике



Рисунок 6 – Диаграмма обновления записей в таблице сотрудников



Рисунок 7 – Диаграмма удаления записи о сотруднике

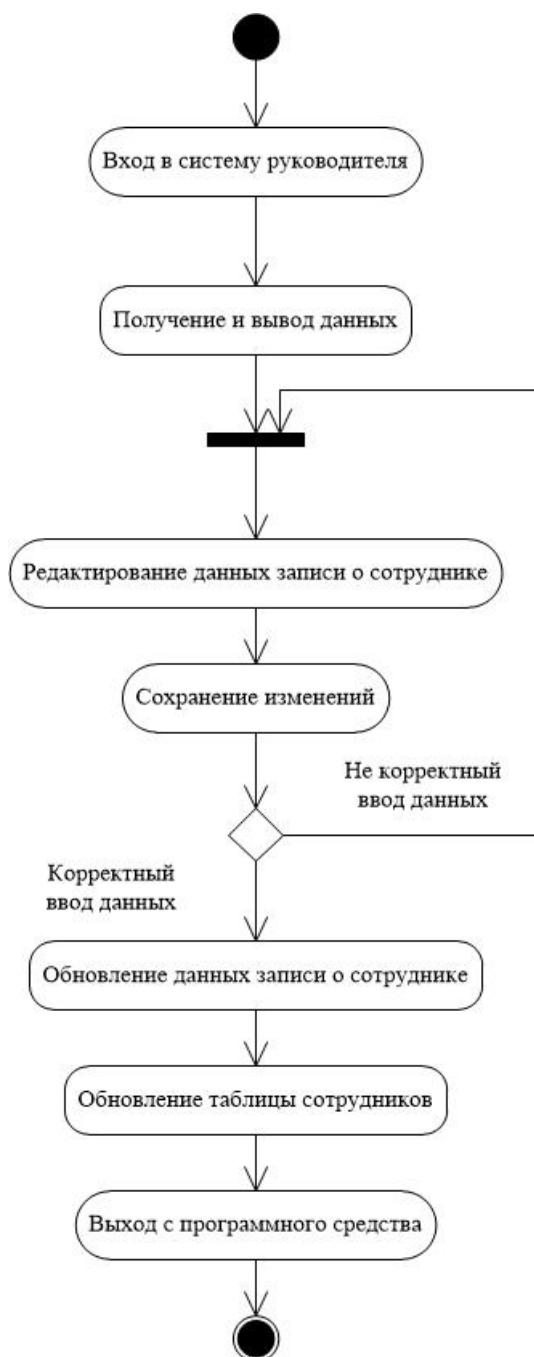


Рисунок 8 – Диаграмма редактирования данных записи о сотруднике



Рисунок 9 – Диаграмма сортировки записей в таблице сотрудников

3.9 Диаграмма классов

Диаграмма классов – диаграмма, демонстрирующая классы системы, их атрибуты, методы и взаимосвязи между ними.

Диаграммы классов используются при моделировании ПС наиболее часто. Они являются одной из форм статического описания системы с точки зрения ее проектирования, показывая ее структуру. Диаграмма классов не отображает динамическое поведение объектов изображенных на ней классов. На диаграммах классов показываются классы, интерфейсы и отношения между ними.

Диаграмма классов представлена на листе 29.

3.10 Диаграмма размещения

Диаграмма размещения отражает физические взаимосвязи между программными и аппаратными компонентами системы. Она является хорошим средством для того, чтобы показать маршруты перемещения объектов и компонентов в распределенной системе.

Программное средство устанавливается на офисный компьютер, который имеет доступ к серверу базы данных MySQL. Диаграмма размещения представлена на рисунке 10.

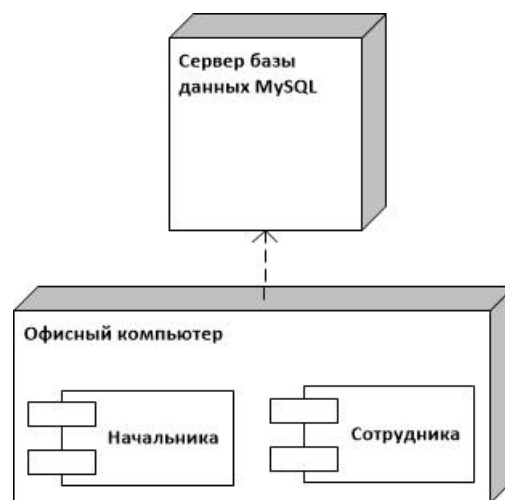


Рисунок 10 – Диаграмма размещения

4 ОПИСАНИЕ ПРОГРАММНОГО КОДА

В данном курсовом проекте использовались следующие паттерны проектирования программных средств: FactoryMethod, Singleton, Command. Которые позволили спроектировать и разработать проект с малой связностью и сильным взаимодействием объектов программного средства.

В программе реализовано 41 классов (classes), определены 3 интерфейса (interfaces), 1 объект перечисления (enumerations). Диаграмма классов представлена в листе 29.

Класс CommandConstant – предназначен, для хранения литеральных названий команд для их вызова и названия ключей для классов ContentCommand и ResultCommand.

Классы ContentCommand и ResultCommand – предназначены для получения командами атрибутов и возвращения результатов выполнения команд соответственно.

Класс CommandException – предназначен для генерации, обработки, хранения, отображения информации о выловленных исключениях при выполнении команд.

Класс Invoker (получатель команды) – предназначен для получения команд для их последующего вызова.

Класс ChiefReciver – предназначен для выполнения команд.

Интерфейс Command – предназначен для определения структуры метода, который будет вызываться для выполнения команды. В процессе использования программного средства метод интерфейса Command заменяется на переданный перечислением CommandType командной ссылкой на метод класса ChiefReciver.

Перечисление CommandType – предназначен для хранения, поиска и возвращения ссылки на методы класса ChiefReciver.

Класс CommandFactory – позволяет получить команду по его названию, который предустановлен в классе CommandConstant. Причем, реализуя паттерн Singleton, он создает экземпляр команды только один раз за всю работу проекта и позволяет пользоваться экземпляром всем, кому это потребуется.

Класс ConnectionFactory – позволяет получить соединение с базой данных MySQL. Причем, реализуя паттерн Singleton, он создает экземпляр соединения только один раз за всю работу проекта и позволяет пользоваться экземпляром всем, кому это потребуется.

Класс DaoConstant – предназначен для хранения и получения предустановленных настроек для соединения с базой данных MySQL, имен таблиц и полей и запросов, для получения, обновления, создания, удаления данных.

Класс DaoException – предназначен для генерации, обработки, хранения, отображения информации о выловленных исключениях при выполнении запросов к базе данных MySQL.

Интерфейс EmployeeDao – определяет основные методы для работы с базой данных учетных записей сотрудников.

Класс EmployeeDaoImpl – предназначен для работы с базой данных учетных записей сотрудников.

Интерфейс TaskDao – определяет основные методы для работы с базой данных выданных заданий.

Класс TaskDaoImpl – предназначен для работы с базой данных выданных заданий.

Класс Employee – предназначен для хранения, получения данных об учетных записях сотрудников.

Класс Task – предназначен для хранения, получения данных о выданных заданиях.

Классы AuthorizationForm, CreateEmployeeDialog, CreateTaskDialog, MainFrame, ToolBarPanel, ToolBarPanelEmployee, TwoTablePanel, DialogClosing – предназначены для создания графического интерфейса программного средства. Использую библиотеку Swing.

Классы CreateEmployeeTableListener, CreateTaskTableListener, ShowCreateEmployeeDialog, ShowCreateTaskDialog, DeleteEmployeeTableListener, DeleteTaskTableListener, LoginListener, RefreshEmployeeTableListener, RefreshTaskByBelongsTableListener, RefreshTaskTableListener, SaveEmployeeTableListener, SaveTaskByBelongsTableListener, SaveTaskTableListener, SortEmployeeTableListener, SortTaskTableListener, MainFrameAction – предназначены для выполнения операций, которые пользователь совершает в процессе пользования программного средства. Использую библиотеку Swing.

Класс GuiConsyant – предназначен для хранения и получения названия полей таблиц, использующиеся для отображения информации в графическом интерфейсе пользователя.

Класс GuiException – предназначен для генерации, обработки, хранения, отображения информации о выловленных исключениях при выполнении пользователем операций в графическом интерфейсе.

Класс Runner – для запуска программного средства на компьютере пользователя.

5 ТЕСТИРОВАНИЕ ПРОГРАММНОГО СРЕДСТВА

Тестирование ПО — это проверка соответствия между реальным и ожидаемым поведением программы, осуществляемое на конечном наборе тестов, выбранном определенным образом. Проведем функциональное тестирование программы. Для этого выполним 5 произвольных сценариев и проверим корректность работы программы.

Тест-кейс 1. Запуск и выход из программного средства, авторизация руководителя

После запуска программного средства отображается диалоговое окно авторизации (рис. 11).

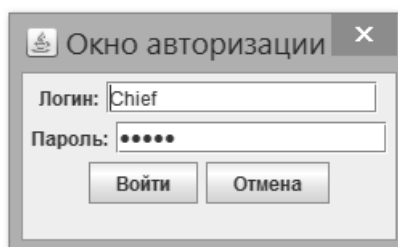


Рисунок 11 – Диалоговое окно авторизации

В поле логина вводим «Chief», в поле пароля вводим «admin», нажимаем кнопку войти и крестик на диалоговом окне. После мы должны увидеть главное окно руководителя (рис. 12). Слева находится панель инструментов, по центру 2 таблицы (верхняя таблица отображает информацию обо всех сотрудниках, нижняя – выданные задания).

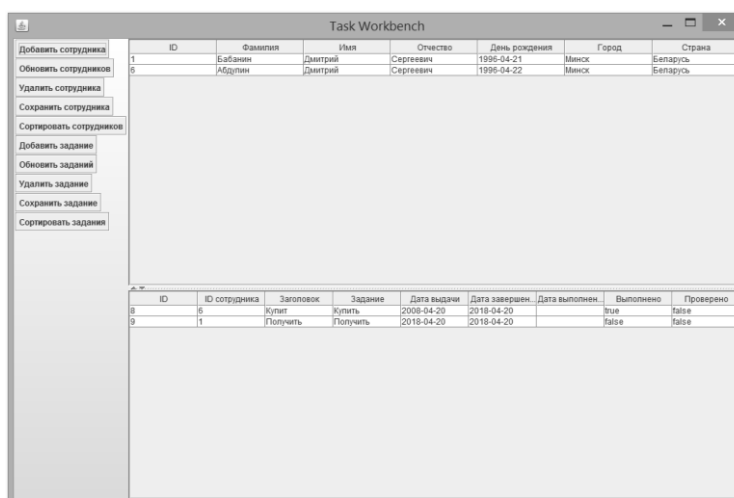


Рисунок 12 – Главное окно руководителя

Чтобы выйти из программы необходимо на главном окне нажать крестик. Всплывет диалоговое окно подтверждения выхода из программного средства (рис. 13).

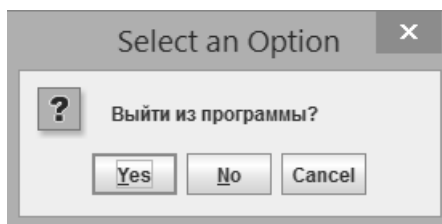


Рисунок 13 – Диалоговое окно подтверждения выхода

Нажимаем кнопку Yes и программа завершает свою работу.

Тест-кейс 2. Запуск и выход из программного средства, авторизация сотрудника

Запуск и выход из программы производиться также, как в тест-кейсе 1, так что по второму разу это рассматривать не будем.

В окне авторизации вводи логин и пароль существующей учетной записи сотрудника. Например выберем учетную запись Бабанина Дмитрия. Вводим в поле логина «1234», в поле пароля «1234», нажимаем войти. Отображается главное окно сотрудника Дмитрия Бабанин с его таблицей заданий (рис 14).

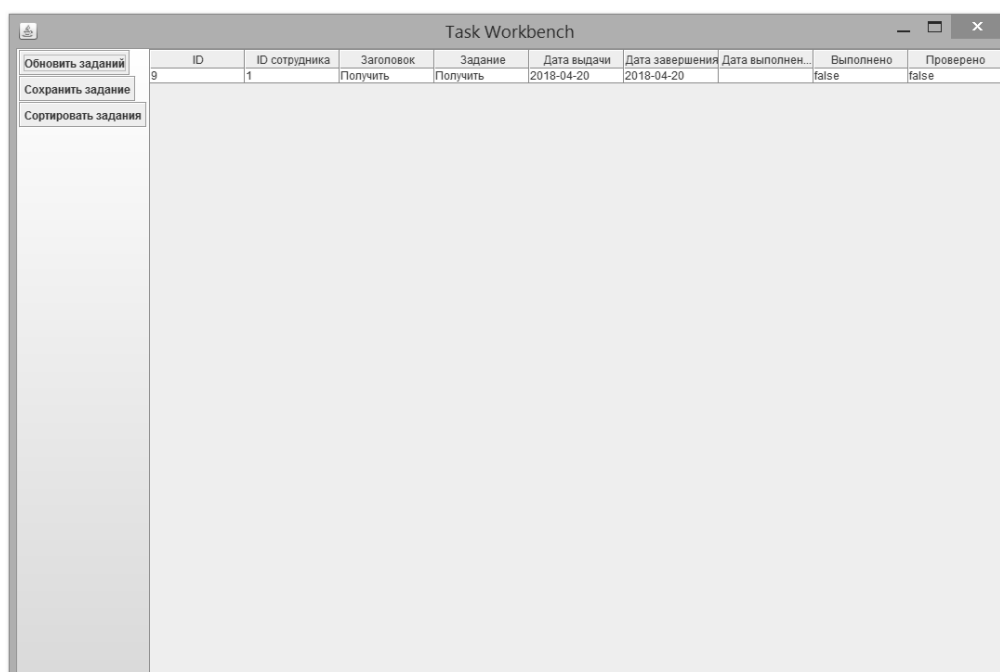
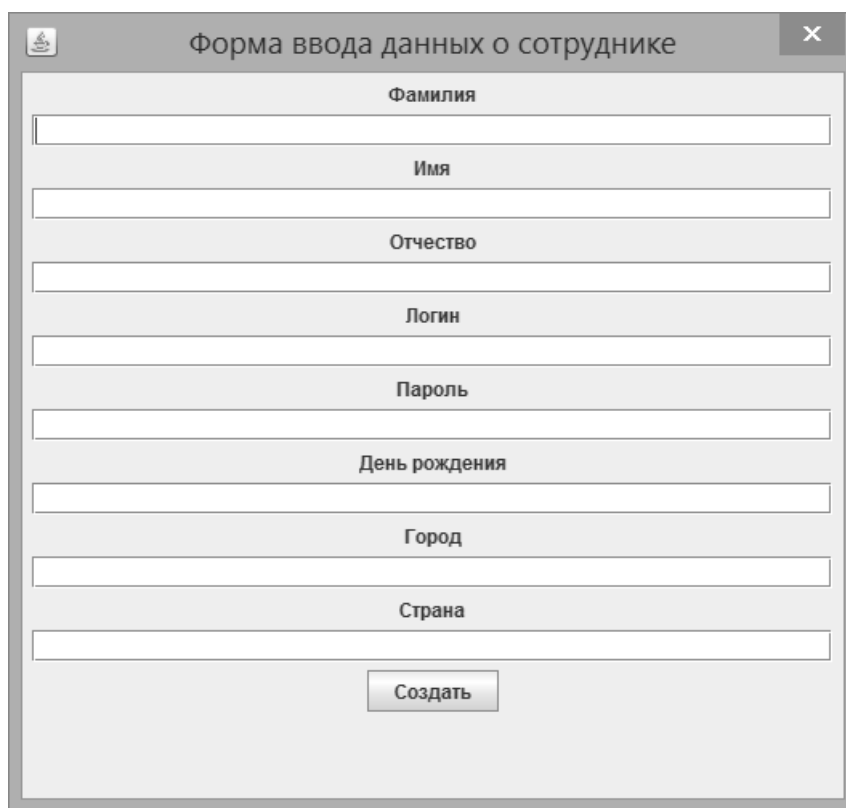


Рисунок 14 – Главное окно сотрудника

Тест-кейс 3. Добавление новой учетной записи сотрудника

В главном окне руководителя на панели инструментов нажимаем кнопку Добавить сотрудника. Всплывает диалоговое окно Форма ввода данных о сотруднике (рис 15).



The image shows a dialog box titled "Форма ввода данных о сотруднике" (Form for entering employee data). It has a standard Windows-style title bar with a close button (X) in the top right corner. The form contains several input fields, each with a label above it: "Фамилия" (Surname), "Имя" (Name), "Отчество" (Patronymic), "Логин" (Login), "Пароль" (Password), "День рождения" (Date of birth), "Город" (City), and "Страна" (Country). Below these fields is a button labeled "Создать" (Create). The form is designed with a light gray background and white input fields.

Рисунок 15 – Форма ввода данных о сотруднике

В поле фамилии вводим «Казаченко», в поле имени «Максим», в поле отчества «Дмитриевич», логин «4321», пароль «4321», дата рождения «1996-08-26», город «Минск», страна «Беларусь». Нажимаем кнопку Создать и крестик на диалоговом окне. В таблице сотрудников появилась новая запись (рис. 16).

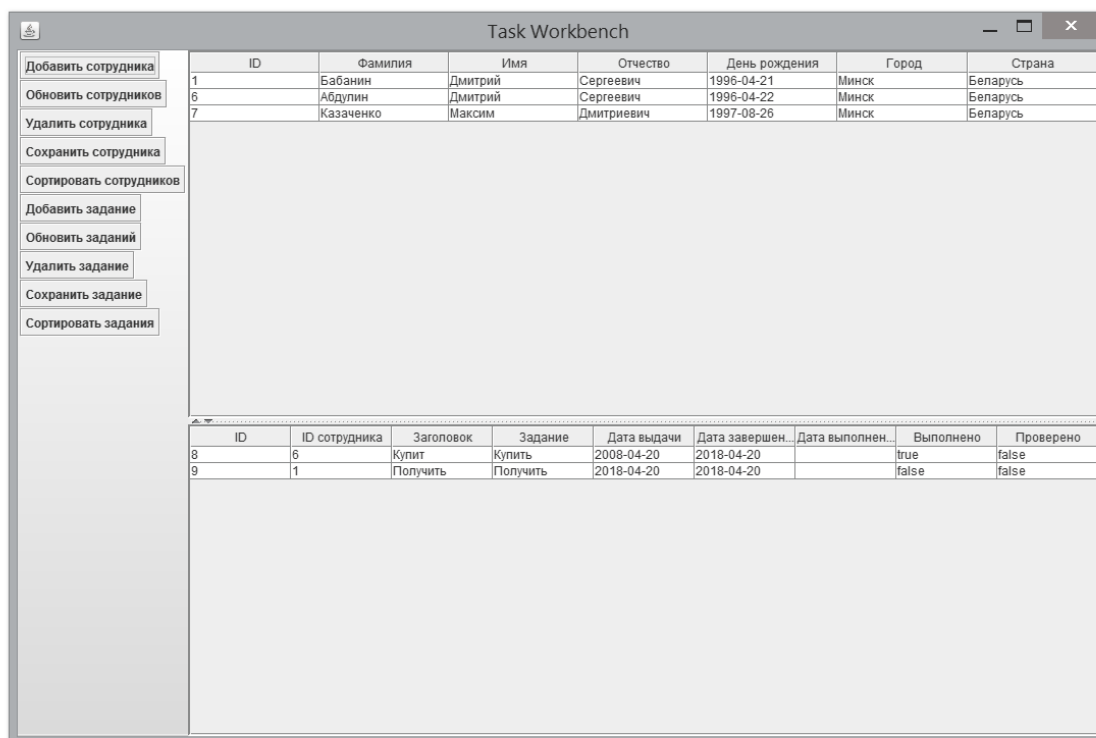


Рисунок 16 – Главное окно руководителя с новой записью в таблице сотрудников

Тест-кейс 4. Удаление учетной записи сотрудника и его заданий

В главном окне руководителя выделяем учетную запись сотрудника с идентификатором 6 одним щелчком левой кнопки мыши. Нажимаем на кнопку Удалить сотрудника. Таблицы сотрудников и выданных заданий обновляется (рис. 17). Теперь в таблице сотрудников всего 2 записи, а в таблице выданных заданий одна, так как вторая запись принадлежала удаленному сотруднику.

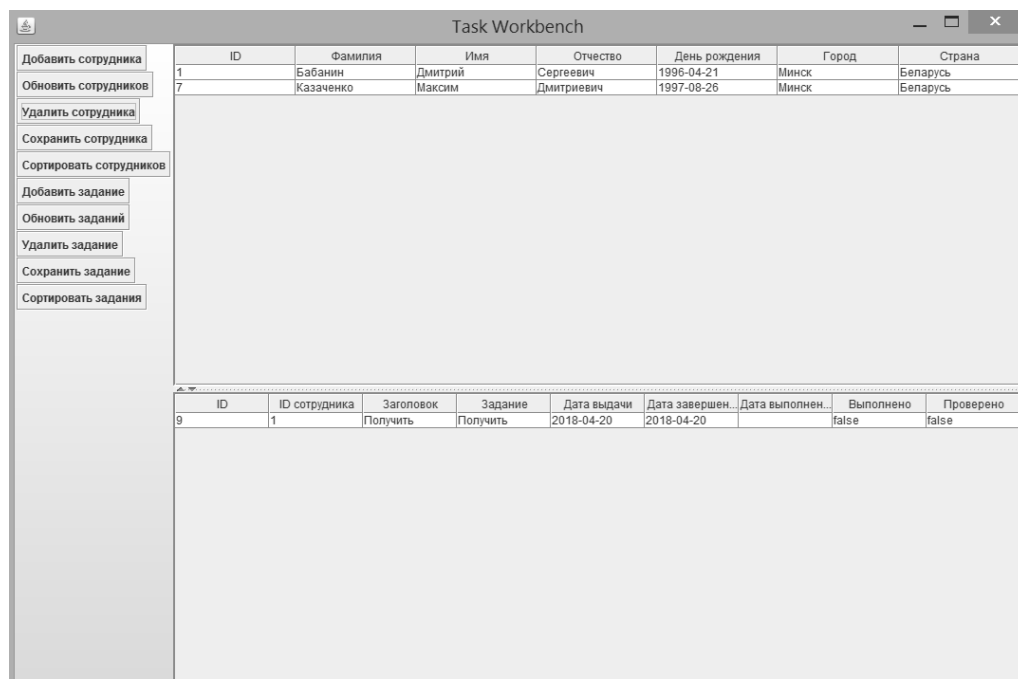


Рисунок 17 – Обновленные таблицы сотрудников и выданных заданий с удаленными записями

Тест-кейс 5. Сортировка и обновление таблицы учетных записей сотрудников

Перед тем, как выполнять этот тест-кейс необходимо добавить пару учетных записей сотрудников. На рисунке 18 показана не отсортированная таблица сотрудников с добавленными записями.

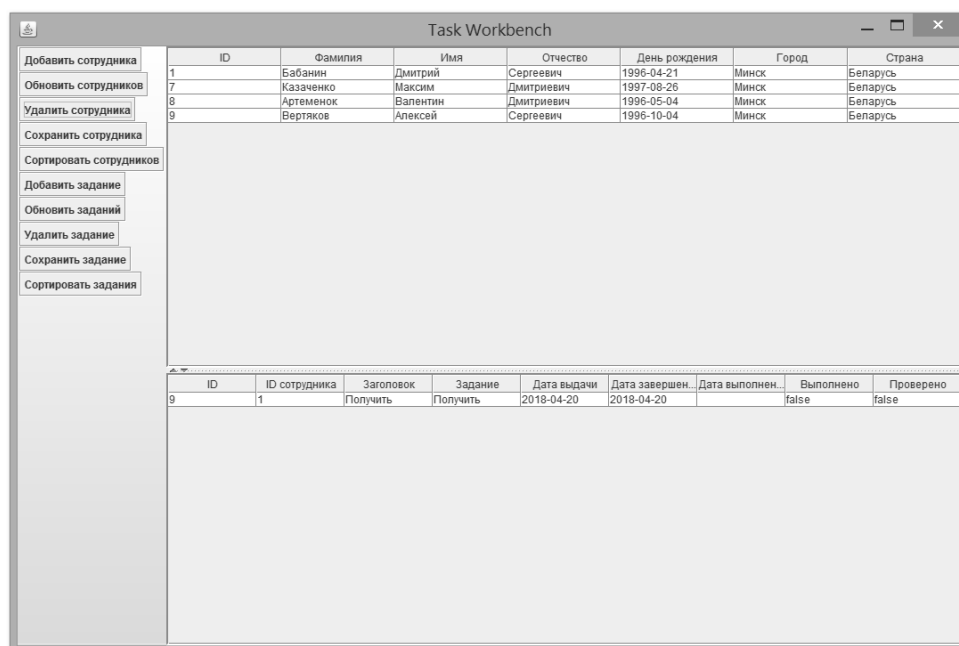


Рисунок 18 – Таблица сотрудников с новыми учетными записями

Нажимаем кнопку Сортировать сотрудников и ведем, что записи от сортировались по фамилиям учетных записей сотрудников (рис. 20).

ID	Фамилия	Имя	Отчество	День рождения	Город	Страна
8	Артеменок	Валентин	Дмитриевич	1996-05-04	Минск	Беларусь
1	Бабанин	Дмитрий	Сергеевич	1996-04-21	Минск	Беларусь
9	Вертяков	Алексей	Сергеевич	1996-10-04	Минск	Беларусь
7	Казаченко	Максим	Дмитриевич	1997-08-26	Минск	Беларусь

ID	ID сотрудника	Заголовок	Задание	Дата выдачи	Дата завершен...	Дата выполнен...	Выполнено	Проверено
9	1	Получить	Получить	2018-04-20	2018-04-20		false	false

Рисунок 20 – Таблица с отсортированной таблицей учетных записей сотрудников

6 РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ

Некоторый функционал приложения уже был рассмотрен в разделе «Тестирование программного средства».

Осталось рассмотреть функции редактирования и сохранения информации об учетной записи сотрудников у руководителя и сотрудника. Так как работа с выданными заданиями схожа с работой учетных записей пользователей, то эти функции рассматриваться не будут.

Прежде всего в таблице сотрудников можно изменять все поля кроме идентификатора. При его изменении и попытке сохранить программа просто проигнорирует операцию. Тоже касается и записей выданных заданий.

Для изменения поля записи необходимо дважды щелкнуть левой кнопкой мыши на соответствующую ячейку. Ячейка выделится рамочкой и появится редактирующий курсор (рис 21).

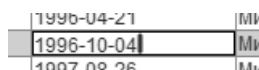


Рисунок 21 – Выделение поля Дата рождения для его изменения

При некорректном вводе даты и попытке сохранить программа проигнорирует только изменение даты, все остальное будет успешно изменено. Тоже касается и записей выданных заданий.

Для завершения редактирования необходимо нажать Enter, выделить измененную запись курсором мыши и нажать кнопку Сохранить сотрудника.

В главном окне сотрудника можно изменять только одно поле записи выданных заданий – это Выполнено, для уведомления руководителя о выполненном задании. Все остальные изменения полей игнорируются.

ЗАКЛЮЧЕНИЕ

В результате выполнения курсовой работы было спроектировано и разработано программное средство учета выданных заданий. Существующий функционал позволяет просматривать, добавлять, удалять, изменять, сортировать записи учетных записей сотрудников и выданных заданий.

Система может использоваться в компаниях с большим количеством сотрудников. Это позволит сэкономить время и силы руководителей предприятия, контролировать выполнение работ удаленно.

В дальнейшем, в программу можно внедрить такие возможности, как:

- Поиск нужных записей;
- Составление и просмотр статистики выполнения работ каждого сотрудника;
- Формирование отчетов о выполнении и проверки на качество выполнения работ;
- Внедрение системы бальной выплаты зарплат сотрудникам;
- Разработать Web-приложение и программное средство на платформе Android.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

- 1 Бахтизин, В. В. Технология разработки программного обеспечения / В. В. Бахтизин, Л. А. Глухова. – М. : Минск БГУИР 2010. – 267 с.
- 2 UML and Rational Rose : Exercises – Wendy Boggs, Michael Boggs
- 3 Г.Шилдт «Java 8. Руководство для начинающих»(6-е издание) (2015)
- 4 Блох Д. «Эффективное программирование.» (2014)
- 5 Кей Хорстманн, Гари Корнелл «Java. Библиотека профессионала. Том 1».10-е издание (2016)
- 6 «ГОСТ 34.602-89»

ПРИЛОЖЕНИЕ А

Техническое задание

Введении

Разрабатываемое ПО будет применяться для автоматизации процесса учета выданных заданий на предприятии, и устанавливаться на офисный компьютер руководителя и сотрудника предприятия.

Основания для разработки

Поскольку на многих предприятиях работают несколько сотен сотрудников. Учитывать вручную и контролировать процесс работы предприятия очень проблематично. Поэтому решение данной задачи будет очень востребовано на крупных предприятиях с большим количеством сотрудников. Причем, если эти же сотрудники работают удаленно, то востребованность данного проекта многократно возрастает.

Назначение разработки

Разрабатываемый программный продукт будет применяться для ускорения и автоматизации учета выданных заданий. Руководитель сможет с помощью программного средства выдать каждому сотруднику задания, контролировать работу предприятия, а также просмотреть все в лаконичном виде.

Требования к программе или программному изделию

1. Требования к функциональным характеристикам:
 - Просмотреть учетные записи сотрудников
 - Добавить учетную запись сотрудника
 - Удалить сотрудника с его заданиями
 - Изменить данные о сотруднике
 - Сортировать сотрудников по фамилиям
 - Просмотреть выданные задания
 - Добавить и выдать задание
 - Удалить выданное задание
 - Изменить выданное задание
 - Сортировать выданные задания по идентификатору

2. Требования к надежности

Входные параметры должны быть отфильтрованы на предмет ввода некорректных данных. Программное средство должно быть защищено от нежелательного вмешательства.

3. Условия эксплуатации

Условия эксплуатации, при которых должны обеспечиваться заданные характеристики, должны удовлетворять требованиям, предъявляемым к техническим средствам в части условий их эксплуатации.

4. Требования к составу и параметрам технических средств

ПО должно работать корректно на устройствах с операционной системой Windows 8.

5. Требования к информационной и программной совместимости

ПО не должно предъявлять дополнительных требований для обеспечения совместимости.

6. Требования к транспортированию и хранению

Готовое ПО должно быть в формате jar и загружено на <http://github.com/> в приватный репозиторий. Ключ от репозитория отправлен на мобильный телефон +375291234567 в виде СМС сообщения.

7. Требования к программной документации

При разработке ПО необходимо руководствоваться ГОСТ 34 201-89 «Виды, комплектность и обозначения документов при создании автоматизированных систем».

Технико-экономические показатели

Разработка ПО носит некоммерческий характер

Стадии и этапы разработки

1. Формирование проектной документации
2. Согласование требований
3. Разработка ПО
4. Тестирование ПО
5. Приемка ПО

Порядок контроля и приемки

В ходе приемки готового программного обеспечения, будет проверена работоспособность при помощи 5 тест-кейсов. В случае успешного решения всех тест-кейсов, ПО будет считаться пригодным к эксплуатации. В ином случае, в акте приема-передачи делается запись с замечаниями к ПО, которые необходимо устранить, а также сроки устранения.

Принятые сокращения

ПО – программное обеспечение