МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ ДОНЕЦКОЙ НАРОДНОЙ РЕСПУБЛИКИ

ГОУВПО «ДОНЕЦКИЙ НАЦИОНАЛЬНЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

ФАКУЛЬТЕТ Компьютерных наук и технологий
КАФЕДРА Программной инженерии
НАПРАВЛЕНИЕ 09.03.04 Программная инженерия
ПРОФИЛЬ Инженерия программного обеспечения

Оценка проекта: _____
Члены комиссии:

_____ _____
Подпись расшифровка подписи

_____ _____
Подпись расшифровка подписи

_____ _____
Подпись расшифровка подписи
«___» _____ 2020 г.

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА
к курсовому проекту по дисциплине
«Архитектура и проектирование программного обеспечения»

Руководитель
_____
должность, звание подпись расшифровка подписи
Нормоконтролер
_____
должность, звание подпись расшифровка подписи
Студент
_____
номер группы подпись расшифровка подписи

Донецк – 2020

РЕФЕРАТ

Тема курсового проекта: разработка информационной системы «Кадровое агентство выпускников вуза».

В состав проекта входят: пояснительная записка 63 страницы, 17 рисунков, 14 таблиц, 4 библиографических названия, 4 приложения.

В курсовом проекте рассмотрен процесс проектирования программного обеспечения для системы автоматизации информационной системы кадрового агентства выпускников вуза.

Проведен анализ предметной области и сформированы требования к разрабатываемой программной системе. Исходя из требований (атрибутов качества), выбрана архитектура системы с применением системных паттернов. Сделан обзор методологий разработки программного обеспечения, языков описания архитектур программных систем и применяемых видов. Выполнены этапы проектирования системы, включая разработку в CASE-среде различных моделей на языке UML.

С учетом типа приложения выбраны инструментальные средства разработки программного обеспечения: C#, Windows Forms, PostgreSQL. При разложении системы на объекты использованы паттерны проектирования. Создано программное обеспечение, обладающее следующими функциональными характеристиками: механизм подписки/отписки на обновления БД, информирование всех подписанных объектов об изменениях в БД, предоставление понятного интерфейса для взаимодействия с БД.

Область применения системы: поиск работников для работодателей и вакансий для выпускников.

ПРОЕКТИРОВАНИЯ ИНФОРМАЦИОННОЙ СИСТЕМЫ «КАДРОВОЕ АГЕНТСТВО ВЫПУСКНИКОВ ВУЗА», ПРОГРАММНАЯ СИСТЕМА, АРХИТЕКТУРА, МОДЕЛИ UML, ПАТТЕРНЫ ПРОЕКТИРОВАНИЯ, ТЕХНОЛОГИЯ РАЗРАБОТКИ ПРОГРАММ

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Проектирование архитектуры программного обеспечения – это один из важнейших этапов создания ПО. В зависимости от выбранной архитектуры и качества ее проектирования проекты могут сильно отличаться по качеству, сложности, связности внутренних частей системы, сложности масштабируемости, сложности поддержки. ПО, которое спроектировано качественно, легко реализуется и поддерживается, также все модули системы являются независимости и имеют возможность расширения функционала без переделывания уже существующих структур и модулей.

В первом разделе проводится анализ требований проектируемой системы, а также производится оценка качества результатов анализа требований.

Во втором разделе выполняется проектирование системы. Выбирается и обосновывается выбор архитектуры ПО, стратегии разработки, инструментальные средства разработки, выявляются основные сценарии использования, выполняется описание особенностей системы при помощи языка UML. Проектируется пользовательский интерфейс системы, производится анализ качества проектирования и оценка дизайна проектируемой системы.

В третьем разделе описываются паттерны проектирования, примененные в данной системе, а также выполняется реализации части программы, которая связана с выбранными паттернами.

Целью курсового проекта является проектирование ПО для информационной системы «Кадровое агентство выпускников вуза», а также выборочная реализация системы при помощи выбранных инструментов.

1 АНАЛИЗ ОБЪЕКТА АВТОМАТИЗАЦИИ

1.1 Назначение системы и задачи автоматизации

Информационная система «Кадровое агентство выпускников вуза» предназначена для трудоустройства выпускников, а также для помощи работодателям в поиске потенциальных работников.

Система имеет два вида пользователей – работодателя и выпускника.

Система должна предоставлять работодателю возможность выполнять следующие задачи:

– Создать заявку;

– Удалить заявку;

– Просмотреть список выпускников;

– Связаться с выпускником;

– Найти выпускника по различным параметрам.

Система должна предоставлять выпускнику возможность выполнять следующие задачи:

– Удалить заявку;

– Просмотреть вакансии;

– Просмотреть предложения от работодателей.

Источником данных для заполнения информации о выпускниках являются списки выпускников, предоставляемые вузами, в которых указывается вся необходимая информация.

Источником данных для изменения информации о выпускниках, вакансиях, работодателях являются действия, производимые выпускниками или работодателями.

Информацией, подаваемой на выход программы для работодателей является:

– Список выпускников:

– Резюме выбранных выпускников.

В списке выпускников указывается год окончания вуза, средний балл, название вуза, название специальности.

Резюме может быть просмотрено в том случае, если выпускник предоставил его.

Информацией, подаваемой на выход программы для выпускников является:

– Список вакансий;

– Предложения работодателей.

В списке вакансий указывается специальность, оклад, место работы, работодатель, специальные требования (если есть).

Предложения работодателей - это список приглашений на работу, которые работодатели уже отправили выпускнику.

Целью курсового проекта является проектирование и разработка системы для автоматизации кадрового агентства выпускников вуза, основной задачей которого является сбор информации о выпускниках и вакансиях, а также предоставление возможности взаимодействия работодателей и выпускников.

Для взаимодействия с пользователем необходимо реализовать интуитивно понятный интерфейс, который позволит без затруднений со стороны пользователя выполнять необходимые действия. Также необходимо реализовать отдельный интерфейс для пользователя-работодателя и пользователя-выпускника.

1.2 Формирование и анализ требований

1.2.1 Требования заказчика

1. Система должна собирать информацию о всех выпускниках вуза.

2. Система должна группировать выпускников по специальностям, которые они окончили.

3. Система должна иметь возможность поиска выпускников по специальности.

4. Система должна иметь возможность сортировки выпускников по среднему баллу диплома.

5. Система должна иметь возможность сортировки выпускников по специальности.

6. Система должна иметь возможность предоставлять полную информацию о каждом конкретном выпускнике.

7. Система должна предоставлять работодателю возможность связаться с выпускником.

8. Система должна предоставлять возможность работодателю оставить заявку на выпускника определенной специальности.

9. Система должна предоставлять возможность удалять выпускника из базы по его желанию.

10. Система должна предоставлять возможность сортировки выпускников по году выпуска.

11. Система должна предоставлять работодателю возможность изменять уже существующую заявку.

12. Система должна предоставлять работодателю возможность удалять свою заявку.

13. Система должна сохранять информацию о выпускниках в течение 2 лет после выпуска.

14. Система должна предоставлять выбор вакансий выпускнику, если он подходит на несколько из них.

15. Система должна оповещать работодателя о выпускнике, желающем работать на него.

1.2.2 Требования разработчика

16. Система должна предоставлять пользователю интуитивно понятный интерфейс.

17. Система должна иметь справку для пользователя.

18. Система должна обрабатывать исключительные ситуации.

19. Система должна быть отказоустойчивой.

20. Система должна предоставлять пользователю данные в понятной форме.

21. Система должна иметь возможность импорта выпускников из файла.

22. Система должна защищать аккаунты пользователей от несанкционированного входа.

23. В случае аварийного завершения программы система не должна терять данные, которые имелись до начала пользовательской сессии.

24. Для корректной работы с системой необходимо:

– 512 Мб. оперативной памяти;
– операционная система семейства Windows;
– 512 или более Мб. вторичной памяти;
– периферийные устройства пользователя;
– тактовую частоту процессора 1.2 ГГц.

1.3 Задачи архитектуры программного обеспечения

1. Уменьшение времени на поиск выпускников по конкретному параметру.

2. Уменьшение времени на осуществление взаимодействия между работодателями и выпускниками.

3. Автоматизация заполнения списка выпускников.

4. Уменьшение количества работников, требующихся для поддержания системы.

5. Уменьшение затрет за счет отсутствия необходимости аренды офиса.

6. Уменьшение рисков, связанных с функциями системы.

7. Уменьшение стоимости поддержки системы автоматизации.

8. Повышение прозрачности работы программы за счет предоставления пользователям всей необходимой информации.

1.4 Оценка качества результатов этапа анализа требований

Факторы и метрики, определяющие качество на различных этапах ЖЦ, поддающиеся анализу для группы 5016 – сервисные программы.

Таблица 1.1 – Факторы, критерии и метрики на различных этапах ЖЦ

№	Факторы	Метрики
1	Надёжность	1. Средства восстановления при ошибках на входе
2		2. Средства восстановления при сбоях оборудования
3	Сопровождаемость	1. Простота архитектуры проекта
4	Удобство применения	8. Эксплуатация
5		9. Управление меню
6		10. Функция поддержки справочной системы (помощи HELP)
7		11. Управление данными
8		12. Рабочие процедуры (JOBS)
9	Эффективность	1. Функции автоматизации
10		2. Затраты времени
11		3. Использование вычислительных ресурсов
12	Универсальность	7. Зависимость от используемого комплекта технических средств
13		8. Зависимость от базового программного обеспечения
14		9. Изоляция немобильности
15	Корректность	5. Требования, предъявляемые к единообразию интерфейсов между модулями и пользователями
17		6. Требования, предъявляемые к единообразию кодирования, символики и определения общих переменных
18		8. Требования, предъявляемые к соответствию ПС стандартам программирования
19		10. Требования, предъявляемые к полноте тестирования

Таблица 1.2 – Оценочные элементы фактора «надёжность»

Код элемента	Наименование	Оценка
H0101	Наличие требований к программе по устойчивости функционирования при наличии ошибок во входных данных	1
H0102	Возможность обработки ошибочных ситуаций	1
H0103	Полнота обработки ошибочных ситуаций	1
H0104	Наличие тестов для проверки допустимых значений входных данных	0,5
H0105	Наличие системы контроля полноты входных данных	1
H0106	Наличие средств контроля корректности входных данных	1
H0107	Наличие средств контроля непротиворечивости входных данных	1
H0108	Наличие проверки параметров и адресов по диапазону их значений	1
H0109	Наличие обработки граничных результатов	1
H0110	Наличие обработки неопределённостей	1
H0201	Наличие требований к программе по восстановлению процесса выполнения в случае сбоя ОС, процессора, внешних устройств	0
H0202	Наличие требований к программе по восстановлению результатов при отказах процессора, ОС	0
H0203	Наличие средств восстановления процесса в случае сбоев оборудования	1
H0204	Наличие возможности разделения по времени выполнения отдельных функций программ	0
H0205	Наличие возможности повторного старта с точки останова	0

Таблица 1.3 – Оценочные элементы фактора «сопровождаемость»

Код элемента	Наименование	Оценка
С0101	Наличие модульной схемы программы	1
С0102	Оценка программы по числу уникальных модулей	0,5

Таблица 1.4 – Оценочные элементы фактора «удобство применения»

Код элемента	Наименование	Оценка
У0801	Уровень языка общения пользователя с программой	1
У0802	Лёгкость и быстрота загрузки и запуска программы	1
У0803	Лёгкость и быстрота завершения работы программы	1
У0804	Возможность распечатки содержимого программы	0
У0805	Возможность приостанова и повторного запуска работы без потерь информации	1
У0901	Соответствие меню требованиям пользователя	1
У0902	Возможность прямого перехода вверх и вниз по многоуровневому меню	0
У1001	Возможность управления подробностью получаемых входных данных	1
У1002	Достаточность полученной информации для продолжения работы	1
У1101	Обеспечение удобства ввода данных	1
У1102	Лёгкость восприятия	1
У1201	Обеспечение программой выполнения предусмотренных рабочих процедур	1
У1202	Достаточность информации, выдаваемой программой для составления дополнительных процедур	1

Таблица 1.5 – Оценочные элементы фактора «эффективность»

Код элемента	Наименование	Оценка
Э0101	Проблемно-ориентированные функции	0,5
Э0102	Машинно-ориентированные функции	0,5
Э0103	Функции ведения и управления	1
Э0104	Функции ввода/вывода	1
Э0105	Функции защиты и проверки данных	1
Э0106	Функции защиты от несанкционированного доступа	1
Э0107	Функции контроля доступа	0
Э0108	Функции защиты от внесения изменений	0,5
Э0109	Наличие соответствующих границ функциональных областей	1
Э0110	Число знаков после запятой в результатах вычислений	0
Э0201	Время выполнения программ	1
Э0202	Время реакции и ответов	1
Э0203	Время подготовки	1
Э0205	Затраты времени на защиту данных	0,5
Э0206	Время компиляции	0,5
Э0301	Требуемый объём внутренней памяти	1
Э0302	Требуемый объём внешней памяти	1
Э0303	Требуемые периферийные устройства	0,5
Э0304	Требуемое базовое программное обеспечение	0,5

Таблица 1.6 – Оценочные элементы фактора «универсальность»

Код элемента	Наименование	Оценка
Г0701	Оценка зависимости программ от ёмкости оперативной памяти ЭВМ	1
Г0702	Оценка зависимости временных характеристик программы от скорости вычислений ЭВМ	0
Г0703	Оценка зависимости функционирования программы от числа внешних запоминающих устройств и их общей ёмкости	0
Г0704	Оценка зависимости функционирования программы от специальных устройств ввода-вывода	1
Г0801	Применение специальных языков программирования	0
Г0802	Оценка зависимости программы от программ ОС	1
Г0803	Зависимость от других программных средств	0
Г0901	Оценка локализации непереносимой части программы	1

Таблица 1.7 – Оценочные элементы фактора «корректность»

Код элемента	Наименование	Оценка
К0501	Единообразие способов вызова модулей	1
К0502	Единообразие процедур возврата управления из модулей	1
К0503	Единообразие способов сохранения информации для возврата	1
К0504	Единообразие способов восстановления информации для возврата	1
К0505	Единообразие организации списков передаваемых параметров	1
К0601	Единообразие наименования каждой переменной и константы	1
К0602	Все ли одинаковые константы встречаются во всех программах под одинаковыми именами	0,5
К0603	Единообразие определения внешних данных во всех программах	0,5
К0604	Используются ли разные идентификаторы для разных переменных	1
К0605	Все ли общие переменные объявлены как общие переменные	0,5
К0606	Наличие определений одинаковых атрибутов	0
К0801	Соответствие организации и вычислительного процесса эксплуатационной документации	0,5
К0802	Правильность заданий на выполнение программы, правильность написания управляющих и операторов	1
К0803	Отсутствие ошибок в описании действий пользователя	1
К0804	Отсутствие ошибок в описании запуска	1
К0805	Отсутствие ошибок в описании генерации	1
К0806	Отсутствие ошибок в описании настройки	1
К1001	Наличие требований к тестированию программ	1
К1002	Достаточность требований к тестированию программ	1
К1003	Отношение числа модулей, отработавших в процессе тестирования и отладки к общему числу модулей	0,5
К1004	Отношение числа логических блоков, отработавших в процессе тестирования и отладки, к общему числу логических блоков в программе	0,5

1.4.1. Оценка фактора «надёжность».

Определим итоговые значения метрик М1, М2 (в зависимости от номера в таблице этапов ЖЦ).

Метрика М1 «Средства восстановления при ошибках на входе» принимает значение:

М1 = (1+1+1+0,5+1+1+1+1+1+1) / 10 = 0,95

Метрика М2 «Средства восстановления при сбоях оборудования» принимает значение:

М2 = (0+0+1+0+0) / 5 = 0,2

Итоговая оценка фактора «надёжность» имеет значение:

$R^{Ф}$ = (0,8*0,95+0,2*0,2)/0,67 = 0,8/0,67 = 1,19 < 1

Это означает, что разработчик ПС обеспечил установленный уровень качества выполнения этапа ЖЦ «Анализ требований» по надёжности.

1.4.2 Оценка фактора «сопровождаемость».

Определим значения метрик М1:

Метрика М1 «Простота архитектуры проекта» принимает значение:

М1 = (1+0,5)/2 = 0,75

Итоговая оценка фактора «сопровождаемость» имеет значение:

$R^{Ф}$ = (1*0,75)/0,7 = 0,75/0,7 = 1,071 > 1

Это означает, что разработчик ПС обеспечил установленный уровень качества выполнения этапа ЖЦ «Анализ требований» по сопровождаемости.

1.4.3 Оценка фактора «удобство применения»

Определим значения метрик М8, М9, М10, М11, М12:

Метрика М8 «Эксплуатация» принимает значение:

М8 = (1+1+1+0+1)/5 = 0,8

Метрика М9 «Управление меню» принимает значение:

М9 = (1+0)/2 = 0,5

Метрика M10 «Функция поддержки справочной системы»:

M10 = (1+1)/2 = 1

Метрика M11 «Управление данными» принимает значение:

M11 = (1+1)/2 = 1

Метрика M12 «Рабочие процедуры (JOBS)» принимает значение:

M12 = (1+1)/2 = 1

Итоговая оценка фактора «удобство применения» имеет значение:

$R^{Ф}$ = (0,1*0,8+0,1*0,5+0,2*1+0,3*1+0,3*1)/0,8 = 0,93/0,8 = 1,16 > 1

Это означает, что разработчик ПС обеспечил установленный уровень качества выполнения этапа ЖЦ «Анализ требований» по удобству применения.

1.4.4 Оценка фактора «эффективность»

Определим значения метрик M1, M2, M3:

Метрика M1 «Функции автоматизации» принимает значение:

M1 = (0,5+0,5+1+1+1+1+0+0,5+1+0) / 10 = 0,65

Метрика M2 «Затраты времени» принимает значение:

M2 = (1+1+1+0,5+0,5)/5 = 0,8

Метрика M3 «Использование вычислительных ресурсов:

M3 = (1+1+0,5+0,5)/4 = 0,75

Итоговая оценка фактора «удобство применения» имеет значение:

$R^{Ф}$ = (0,3*0,65+0,3*0,8+0,4*0,75)/0,8 = 0,735/0,8 = 0,91 < 1

Это означает, что разработчик ПС не обеспечил установленный уровень качества выполнения этапа ЖЦ «Анализ требований» по эффективности.

1.4.5 Оценка фактора «универсальность»

Определим значения метрик M7, M8, M9:

Метрика M7 «Зависимость от используемого комплекса технических средств» принимает значение:

M7 = (1+0+0+1)/4 = 0,5

Метрика M8 «Зависимость от базового программного обеспечения»:

M8 = (0+1+0)/3 = 0,33

Метрика M9 «Изоляция немобильности» принимает значение:

M9 = (1)/1 = 1

Итоговая оценка фактора «удобство применения» имеет значение:

$R^{Ф}$ = (0,2*0,5+0,5*0,33+0,3*1)/0,7 = 0,565/0,7 = 0,807 < 1

Это означает, что разработчик ПС не обеспечил установленный уровень качества выполнения этапа ЖЦ «Анализ требований» по универсальности.

1.4.6 Оценка фактора «корректность»

Определим значения метрик M5, M6, M8, M10:

Метрика M5 «Требования, предъявляемые к единообразию интерфейсов между модулями и пользователями» принимает значение:

M5 = (1+1+1+1+1)/5 = 1

Метрика M6 «Требования, предъявляемые к единообразию кодирования, символики и определения общих переменных»:

M6 = (1+0,5+0,5+1+0,5+0,5)/6 = 0,66

Метрика M8 «Требования, предъявляемые к соответствию ПС стандартам программирования» принимает значение:

M8 = (1+1+1+1+1+1)/6 = 1

Метрика M10 «Требования, предъявляемые к полноте тестирования» принимает значение:

M10 = (1+1+0,5+0,5)/4 = 0,75

Итоговая оценка фактора «удобство применения» имеет значение:

$R^{Ф}$ = 0,3*((0,4*1+0,4*0,66+0,2*1)/0,86) + 0,7*((1*0,75)/0,75) = 0,3*1,004+0,7*1 = 1,0012 > 1

Это означает, что разработчик ПС обеспечил установленный уровень качества выполнения этапа ЖЦ «Анализ требований» по корректности.

2 ПРОЕКТИРОВАНИЕ АРХИТЕКТУРЫ СИСТЕМЫ

2.1 Анализ ключевых вопросов проектирования

На этапе проектирования программного продукта необходимо определиться с ключевыми вопросами, которые оказывают существенное влияние на архитектуру программного продукта [1].

Для проектируемой системы были рассмотрены следующие ключевые вопросы:

1. Взаимодействие и представление. В этом вопросе рассматривается реакция системы на действия пользователей. Речь здесь идет о реакции системы в ответ на действия пользователей и организации ее отклика с точки зрения внутренней организации взаимодействия.

2. Контроль и обработка событий. В этом вопросе рассматриваются явные и неявные способы обработки событий, происходящих в системе.

3. Обработка ошибок и исключительных ситуаций. В этом вопросе рассматриваются возможные сбои при работе системы и способы их предотвращения.

4. Обеспечение отказоустойчивости. В этом вопросе рассматриваются действия системы для восстановления работоспособности после сбоя в системе.

5. Распределение компонентов. В этом вопросе рассматривается распределение и выполнение по различным узлам обработки в терминах программного обеспечения, сетевой инфраструктуры и т.п. Также в этом вопросе рассматривается использование связующего программного обеспечения.

2.2 Стратегии и методы проектирования ПО

Существуют различные общие стратегии, помогающие в проведении работ по проектированию. В отличие от общих стратегий, методы проектирования более специфичны и, в основном, предлагают и

предоставляют нотации (или наборы нотаций) для использования совместно с этими методами, а также процессы, которым необходимо следовать в рамках используемого метода. Методы в данном контексте это не просто некие «слабоформализованные» или просто частные практические подходы или техники. Методы здесь являются более общими понятиями, это - методологии, сконцентрированные на процессе (в частности, проектирования) и предполагающие следование определенным правилам и соглашениям, в том числе – по используемым выразительным средствам.

Для проектируемой системы была выбрана одна из самых популярных стратегий «разделяй-и-властвуй». В этой стратегии основную задачу разделяют на несколько более простых подзадач, благодаря чему существенно облегчается разработка всей системы.

Проектируемая система была разбита на модули, выполняющие отдельные задачи:

1. Модуль работы с БД.
2. Модуль обработки команд пользователя.
3. Модуль импорта с файла.
4. Модуль приложения пользователя.

Методом проектирования был выбран один из классических методов – объектно-ориентированный. Этот метод самый подходящий для реализации выбранной стратегии.

2.3 Определение целей архитектуры

Наличие чётких целей поможет сосредоточиться на архитектуре и правильном отборе проблем для решения. Цели архитектуры – это задачи и ограничения, очерчивающие архитектуру и процесс проектирования ПО, определяющие объём работ и помогающие понять, когда пора завершить процесс доработки архитектуры [2].

1. Начальное определение задач: задачей архитектуры будет являться разработка архитектуры нового приложения кадрового агентства по трудоустройству выпускников вуза.

2. Потребителями будут выпускники вуза, работодатели, администраторы; архитектура будет представлена заказчику на одобрение, но ему может быть достаточно общей сметы затрат, чтобы утвердить или предложить переработать архитектуру, если предлагаемое решение получится слишком дорогим для него.

3. Основные ограничения, накладываемые на архитектуру: наличие только графического интерфейса; отсутствие интеграции с мобильными устройствами; невозможность работы без предварительного заполнения БД.

2.4 Основные сценарии

Ключевые сценарии – это описание способа взаимодействия между разрабатываемой системой и одним или более действующим лицом (либо её пользователями, либо другими системами). Главной целью при продумывании архитектуры системы должно быть выявление нескольких возможных ключевых сценариев, что поможет в принятии решений по архитектуре, причём основная задача состоит в нахождении баланса между требованиями, предъявляемыми заказчиком [3].

Для проектируемой системы было выделено 3 роли пользователей:

1. Администратор;
2. Пользователь-работодатель;
3. Пользователь-выпускник.

Администратор системы имеет доступ к модулю работы с БД. Главной задачей администратора является поддержание БД в рабочем состоянии, решение проблем, возникающих у пользователей, а также модификация БД в случае необходимости (рис. 2.1).

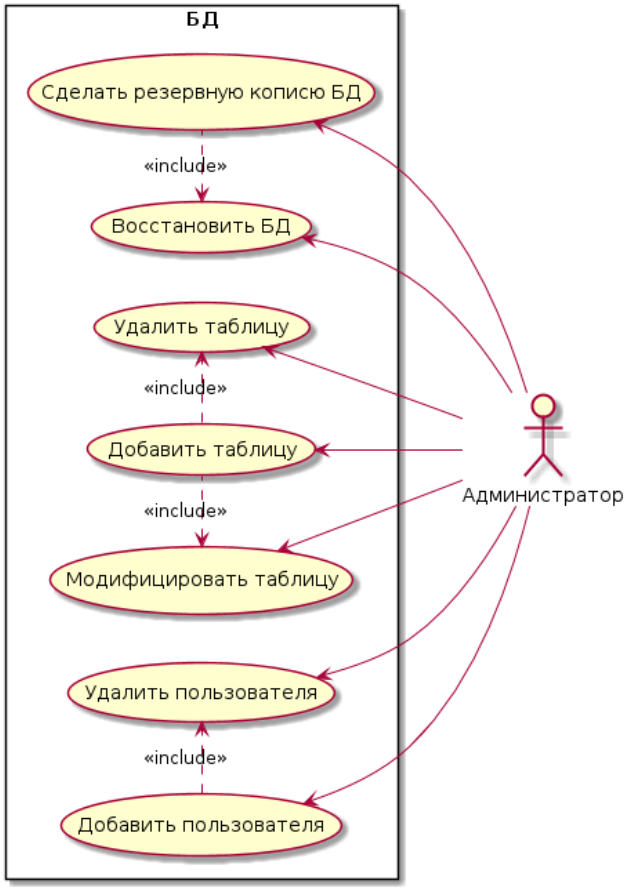

Рисунок 2.1 – Диаграмма прецедентов для Администратора

Метрическая оценка диаграммы с использованием формулы (В.1), а также таблиц В.1 и В.2:

$$S = \frac{5 + 7 + 7 + 8}{1 + 8 + \sqrt{2 + 2}} = 2{,}45.$$

Сравнивая с диапазоном значений в таблице В.3 можем сделать вывод, что оцениваемая диаграмма недостаточно информативна. Это обусловлено тем, что в диаграмме описаны только тривиальные варианты использования системы.

Пользователь-работодатель - это роль, которая позволяет пользователям взаимодействовать с системой от имени работодателя. Пользователь-работодатель имеет возможность оставить заявку, просмотреть список выпускников, отсортировать список выпускников, найти выпускников по

конкретному признаку, отправить приглашение, посмотреть присланные резюме, удалить заявку.

Рисунок 2.2 – Диаграмма прецедентов пользователя-работодателя

Метрическая оценка диаграммы с использованием формулы (В.1), а также таблиц В.1 и В.2:

$$S = \frac{5 + 13 + 7 + 14}{1 + 14 + \sqrt{2 + 3}} = 2{,}26.$$

Сравнивая с диапазоном значений в таблице В.3 можем сделать вывод, что оцениваемая диаграмма недостаточно информативна. Это обусловлено тем, что на диаграмме отображено слишком много однотипных объектов.

Пользователь-выпускник – это роль, которая позволяет пользователям взаимодействовать с системой от имени выпускника. Пользователь-выпускник имеет возможность просмотреть список вакансий, оправить резюме, просмотреть приглашения, удалить себя из БД, отсортировать вакансии и выполнить поиск вакансий по определенному признаку.

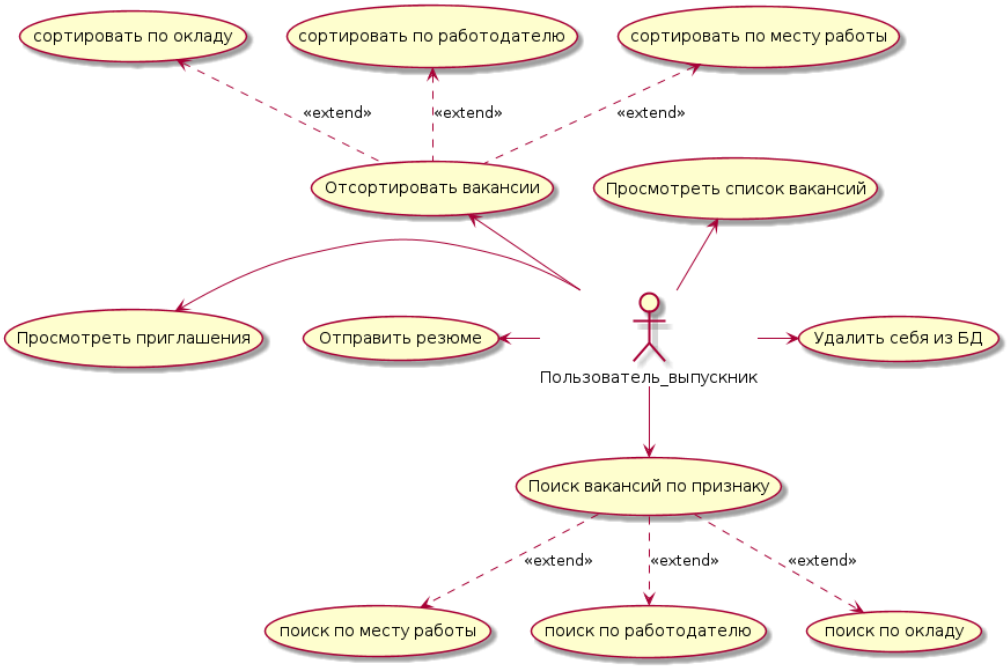

Рисунок 2.3 – Диаграмма прецедентов пользователя-выпускника

Метрическая оценка диаграммы с использованием формулы (В.1), а также таблиц В.1 и В.2:

$$S = \frac{5 + 12 + 6 + 12}{1 + 13 + \sqrt{2 + 2}} = 2{,}19.$$

Сравнивая с диапазоном значений в таблице В.3 можем сделать вывод, что оцениваемая диаграмма недостаточно информативна. Это обусловлено тем, что на диаграмме отображено слишком много однотипных объектов.

2.5 Тип приложения

Типом приложения является классическое desktop-приложение. Для ввода и вывода информации используется графический интерфейс. Такой тип приложения не требует от пользователя большого опыта работы с компьютером.

2.6 Определение ограничений развертывания

Ограничения по аппаратным требованиям:

– 512 или более мб. вторичной памяти;

– периферийные устройства пользователя;

– тактовую частоту процессора 1.2 ГГц.

Ограничения по развёртыванию:

– сервер БД;

– Сервер связи между БД и внешним миром;

– 512 мб. оперативной памяти;

Ограничения по программным требованиям:

– операционная система семейства Windows;

– СУБД PostgreSQL;

– Пакет, поддерживающий средства Microsoft Visual C# 2019.

Повышенные требования безопасности: при входе в аккаунт пользователя необходимо ввести пароль.

2.7 Выбор архитектурного стиля проектирования

Для выполнения курсового проекта была выбрана трехзвенная архитектура «клиент-сервер» («тонкий» клиент – «толстый» сервер), структура которого изображена на рисунке 2.4.

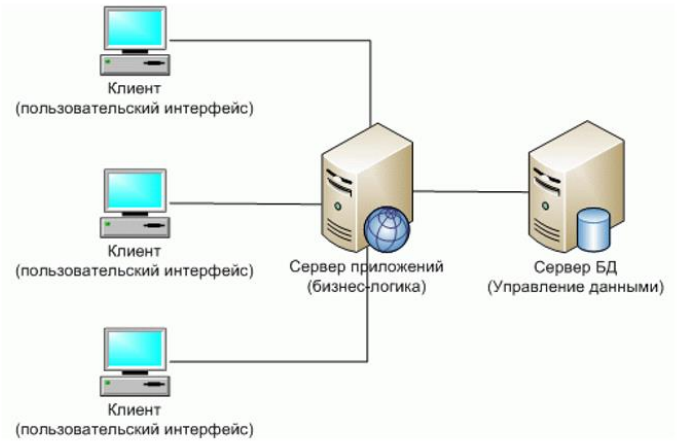

Рисунок 2.4 – Трехзвенная архитектура «клиент-сервер»

В этой архитектуре, кроме клиентской части системы и сервера базы данных, вводится промежуточный сервер приложений.

На стороне клиента выполняются только интерфейсные действия, а вся логика обработки информации поддерживается в сервере приложений.

Также подобная архитектура предотвращает серьезные последствия в случае сбоя в системе. При таком сбое упадет только «тонкий» клиент, который без серьезных последствий для системы можно перезагрузить или заменить.

2.8 Графические представление архитектуры на языке UML

2.8.1 Диаграмма классов

В системе описано 6 классов:

1. Database – класс, отвечающий за непосредственное взаимодействие с БД. Он содержит базовые операции для взаимодействия с БД: добавление записи, удаление записи, изменение записи, выполнить запрос. Принимает команду от сервера приложений и возвращает результат ему же.

2. Appserver – класс, который отвечает за сервер приложений. Содержит объект класса Database и методы для работы с БД. Служит связующим звеном между сервером БД и клиентами. Хранит в своих методах основную бизнес-логику системы, а также выполняет проверку корректности введенных пользователем данных.

3. User – абстрактный класс, который содержит общие поля и виртуальные методы для классов Graduate («Выпускник») и Hirer («Работодатель»).

4. Graduate – класс, который представляет выпускника. Содержит в себе поля: специальность, средний балл, количество запросов, год окончания обучения. А также методы просмотра приглашений и отправки резюме.

5. Hirer – класс, который определяет работодателя. Содержит в себе поле количество присланных резюме. Содержит методы взаимодействия с приглашениями, а также метод просмотра резюме.

6. Request – класс, который описывает структуру запроса на работу (вакансии). В нем содержатся поля: оклад, место работы, работодатель.

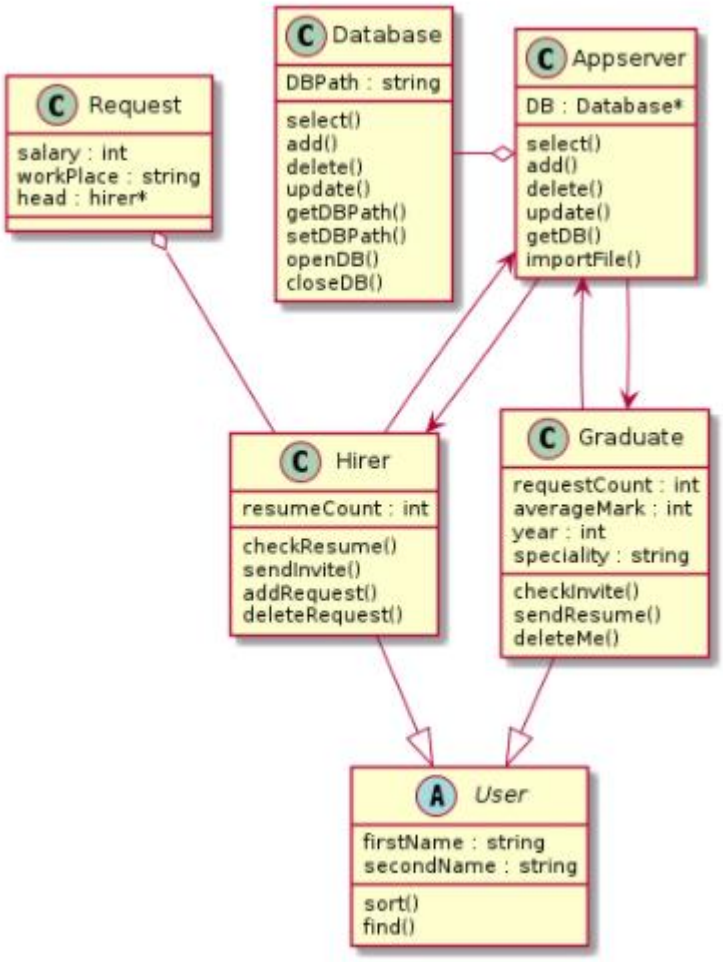

Рисунок 2.5 – Диаграмма классов

Перед метрической оценкой самой диаграммы классов необходимо просчитать оценку для каждого класса системы при помощи формулы (В.2).

Класс Database

$$S_{cls} = \frac{\sqrt{1} + \sqrt{8}}{0.3 * (1 + 8)} = \frac{3.82}{2.7} = 1.41$$

Класс Appserver

$$S_{cls} = \frac{\sqrt{1} + \sqrt{6}}{0.3 * (1 + 6)} = \frac{3.45}{2.1} = 1.64$$

Класс User

$$S_{cls} = \frac{\sqrt{2} + \sqrt{2}}{0.3 * (2 + 2)} = \frac{2.83}{1.2} = 2{,}36$$

Класс Graduate

$$S_{cls} = \frac{\sqrt{4} + \sqrt{3}}{0.3 * (4 + 3)} = \frac{3.73}{2.1} = 1{,}78$$

Класс Hirer

$$S_{cls} = \frac{\sqrt{1} + \sqrt{4}}{0.3 * (1 + 4)} = \frac{3}{1.5} = 2$$

Класс Request

$$S_{cls} = \frac{\sqrt{3} + \sqrt{0}}{0.3 * (3 + 0)} = \frac{1.73}{0.9} = 1{,}92$$

Метрическая оценка диаграммы с использованием формулы (В.1), а также таблиц В.1 и В.2:

$$S = \frac{30 + (1{,}41 + 1{,}64 + 2{,}36 + 1{,}78 + 2 + 1{,}92) + 4 + 8}{1 + 6 + \sqrt{3 + 1}} = 5{,}9.$$

Сравнивая с диапазоном значений в таблице В.3 можем сделать вывод, что оцениваемая диаграмма превышает допустимый интервал, что говорит о ее сложной структуре.

2.8.2 Диаграмма компонентов

Для соответствия трехзвенной архитектуре в системе реализовано 3 компонента (рис. 2.6):

1. Сервер БД
2. Сервер приложения
3. Клиентское приложение

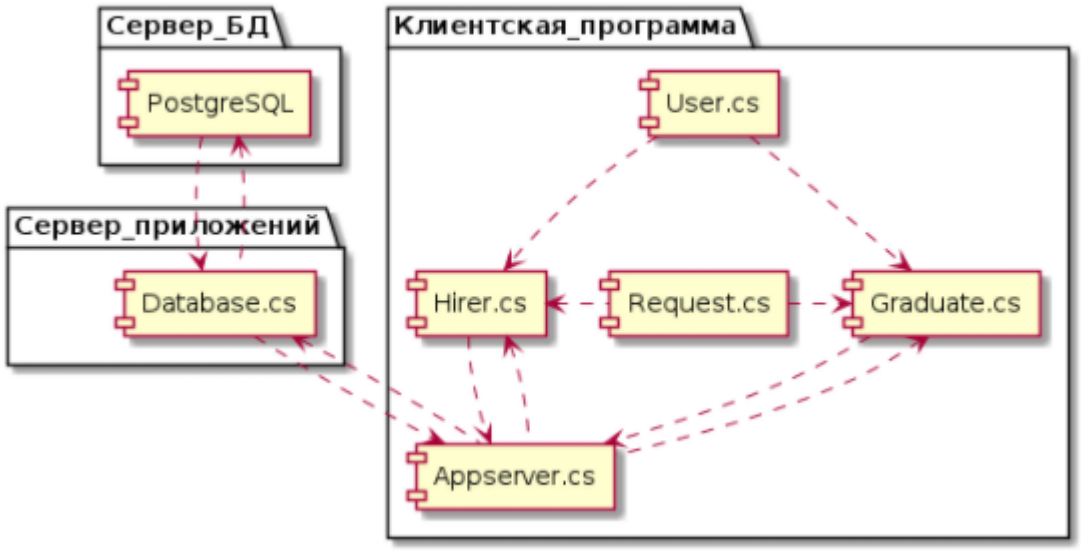

Рисунок 2.6 – Диаграмма компонентов.

Метрическая оценка диаграммы с использованием формулы (В.1), а также таблиц В.1 и В.2:

$$S = \frac{28 + 12 + 6 + 16}{1 + 10 + \sqrt{2 + 1}} = 4{,}87.$$

Сравнивая с диапазоном значений в таблице В.3 можем сделать вывод, что оцениваемая диаграмма превышает допустимый интервал, что говорит о ее сложной структуре.

2.8.3 Диаграмма кооперации

Диаграмма кооперации предназначена для описания поведения системы на уровне отдельных объектов, которые обмениваются между собой сообщениями, чтобы достичь нужной цели или реализовать некоторый вариант использования. Такое представление структуры модели как совокупности взаимодействующих объектов и обеспечивает диаграмма кооперации.

Рисунок 2.7 – Диаграмма кооперации

Диаграмма кооперацій не оценивается, вместо этого оценивается диаграмма деятельности.

2.8.4 Диаграмма деятельности

Диаграммы деятельности можно считать частным случаем диаграмм состояний. Именно они позволяют реализовать в языке UML особенности процедурного и синхронного управления, обусловленного завершением внутренних деятельностей и действий. Метамодель UML предоставляет для этого необходимые термины и семантику. Основным направлением использования диаграмм деятельности является визуализация особенностей реализации операций классов, когда необходимо представить алгоритмы их выполнения. При этом каждое состояние может являться выполнением операции некоторого класса либо ее части, позволяя использовать диаграммы деятельности для описания реакций на внутренние события системы.

В контексте языка UML деятельность (activity) представляет собой некоторую совокупность отдельных вычислений, выполняемых автоматом. При этом отдельные элементарные вычисления могут приводить к некоторому результату или действию (action). На диаграмме деятельности отображается логика или последовательность перехода от одной деятельности к другой, при

этом внимание фиксируется на результате деятельности. Сам же результат может привести к изменению состояния системы или возвращению некоторого значения. [3]

Для диаграммы деятельности был выбран процесс осуществления запроса со стороны пользовательского приложения (рис 2.).

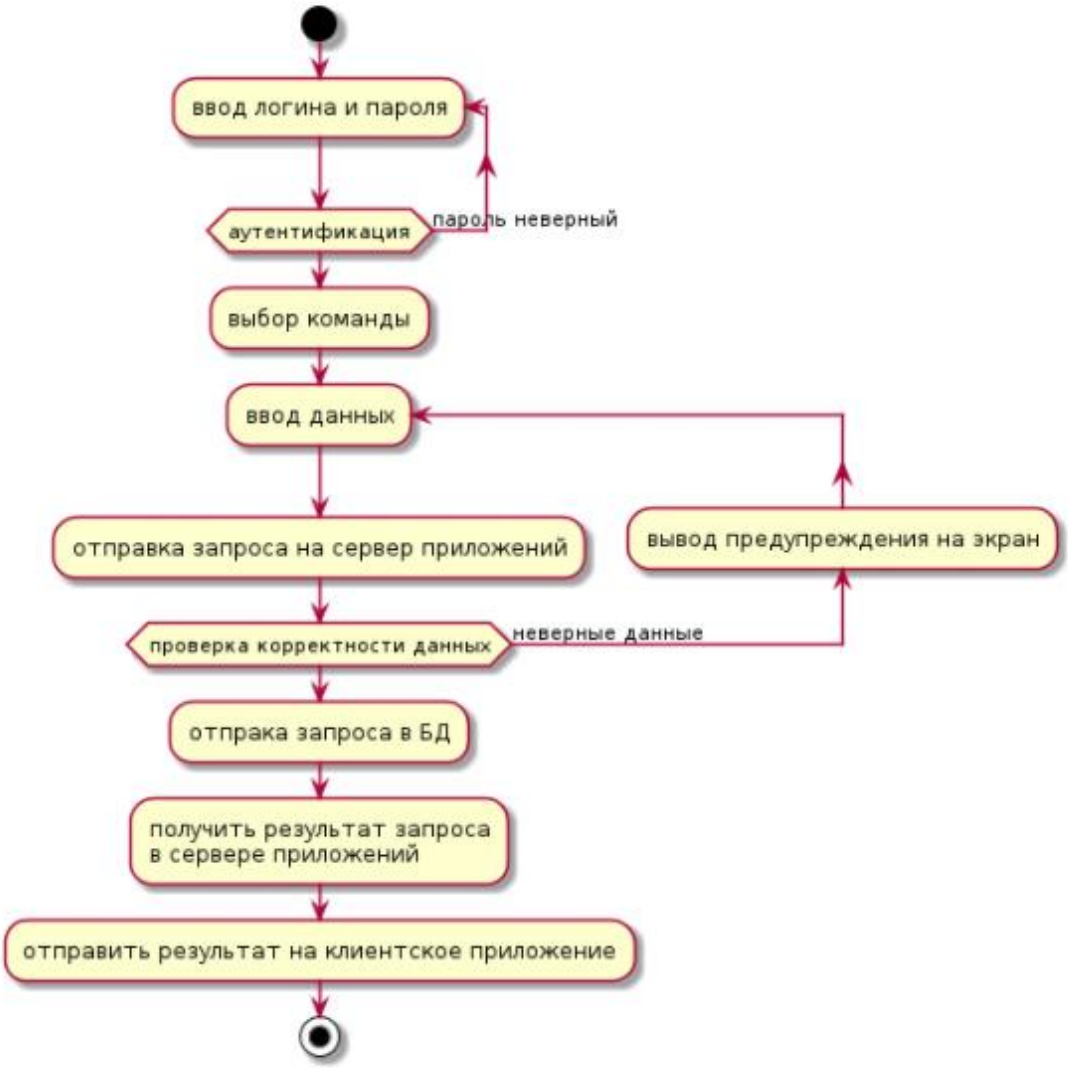

Рисунок 2.8 – Диаграмма деятельности

Метрическая оценка диаграммы с использованием формулы (В.1), а также таблиц В.1 и В.2:

$$S = \frac{40 + 22}{1 + 10 + \sqrt{2 + 1}} = 4{,}86.$$

Сравнивая с диапазоном значений в таблице В.3 можем сделать вывод, что оцениваемая диаграмма превышает допустимый интервал, что говорит о ее сложной структуре.

2.8.5 Диаграмма последовательности

Диаграмма последовательности описывает взаимодействие модулей системы в рамках конкретного прецедента. На рисунке 2.9 представлена диаграмма последовательности для прецедента «Просмотреть список выпускников».

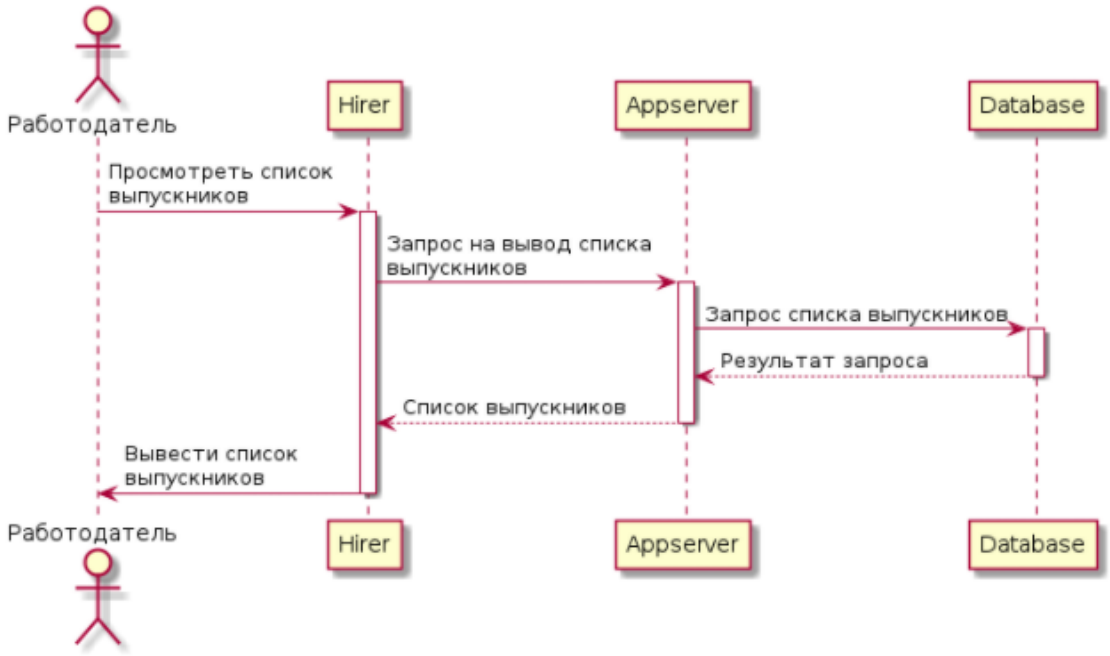

Рисунок 2.9 – Диаграмма последовательности

Метрическая оценка диаграммы с использованием формулы (В.1), а также таблиц В.1 и В.2:

$$S = \frac{20 + 6}{1 + 4 + \sqrt{2 + 2}} = 3{,}71.$$

Сравнивая с диапазоном значений в таблице В.3 можем сделать вывод, что оцениваемая диаграмма превышает допустимый интервал, что говорит о большом количестве промежуточных команд.

2.8.6 Диаграмма развертывания

Диаграмма развёртывания отображает связи между отдельными приложениями или устройствами, а также устанавливает зависимости разных компонентов всей системы (см. рис. 2.10). [4]

Рисунок 2.10 – Диаграмма развертывания

Метрическая оценка диаграммы с использованием формулы (В.1), а также таблиц В.1 и В.2:

$$S = \frac{12 + 8 + 3}{1 + 7 + \sqrt{3 + 2}} = 2{,}24.$$

Сравнивая с диапазоном значений в таблице В.3 можем сделать вывод, что оцениваемая диаграмма попадает в допустимый интервал.

2.8.7 Диаграмма состояний

Диаграмма состояний позволяет увидеть все состояния объекта конкретного класса в течение его жизненного цикла. Эта диаграмма позволяет смоделировать динамические аспекты системы. Диаграмма состояний для экземпляра объекта класса Graduate представлена на рисунке 2.11.

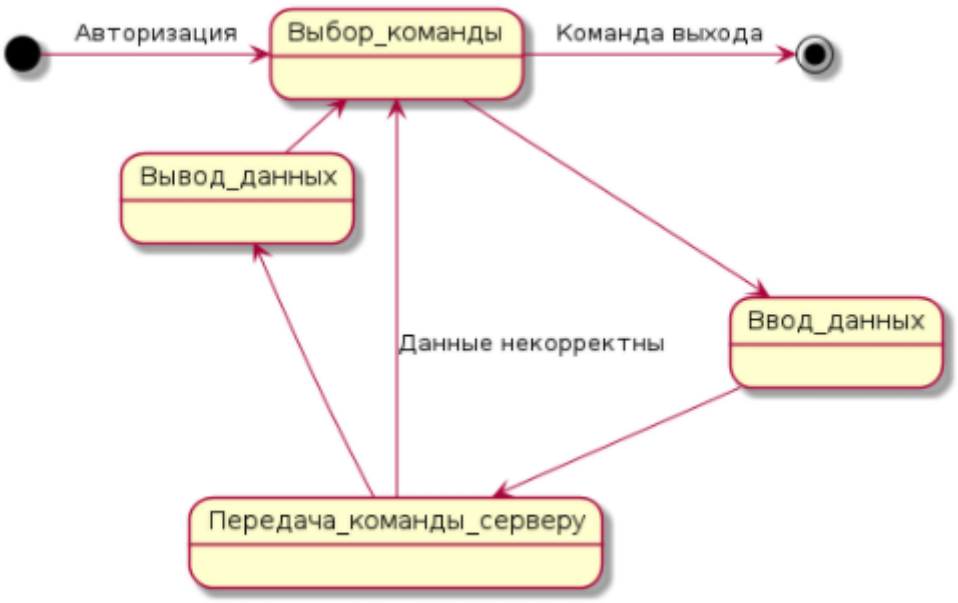

Рисунок 2.11 – Диаграмма состояний

Метрическая оценка диаграммы с использованием формулы (В.1), а также таблиц В.1 и В.2:

$$S = \frac{16 + 7}{1 + 4 + \sqrt{1 + 1}} = 3{,}58.$$

Сравнивая с диапазоном значений в таблице В.3 можем сделать вывод, что оцениваемая диаграмма превышает допустимый интервал, что говорит о большом количестве переходов между состояниями.

2.9 Детализация программ с помощью паттернов проектирования

В проектируемой системе используется такие паттерны проектирования как: наблюдатель, фасад.

Паттерн наблюдатель используется для оповещения пользователей об изменениях в БД, благодаря чему поддерживается актуальность данных, предоставляемых пользователю. При изменении в БД пользовательский интерфейс сообщит пользователю о том, что необходимо произвести повторный запрос для обновления данных.

Паттерн фасад позволяет пользователям не напрямую обращаться к системе, а к простому пользовательскому интерфейсу, который предоставляет система. Это позволяет избежать множества потенциальных проблем, связанных с неверным использованием команд системы.

2.10 Проектирование пользовательского интерфейса

Пользовательский интерфейс – это основное средство взаимодействия между системой и пользователем. Поэтому от простоты, полноты и эффективности интерфейса будет зависеть эффективность и скорость работы пользователей с системой.

При проектировании интерфейса упор был сделан на простоте и интуитивности интерфейса. Такие особенности обусловлены тем, что пользователи системы могут иметь совершенно различный уровень владения компьютером. Помимо этого, простота интерфейса позволяет использовать систему без специальных курсов подготовки.

Для обеспечения полноты интерфейса использовалась диаграммы последовательности (см. рис. 2.2 – 2.3)

На рисунках 2.12, 2.13 представлены прототипы основных форм для работы работодателей и выпускников соответственно.

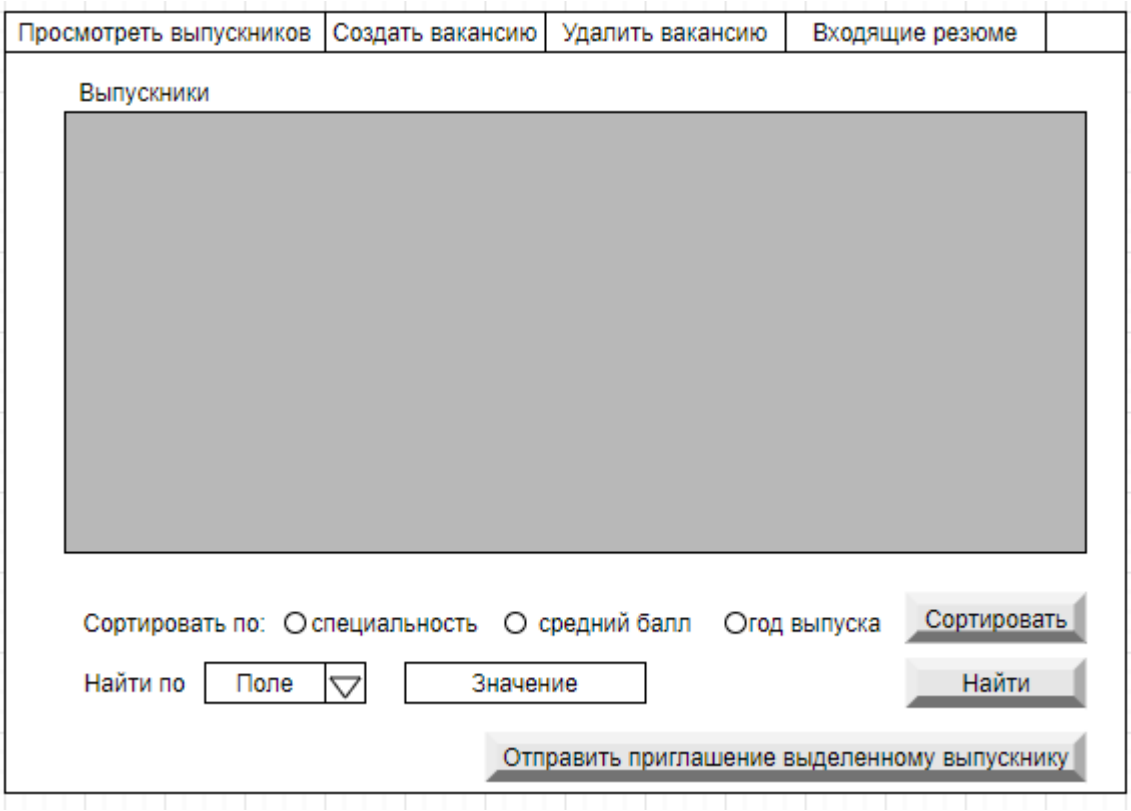

Рисунок 2.12 – Прототип интерфейса работодателя

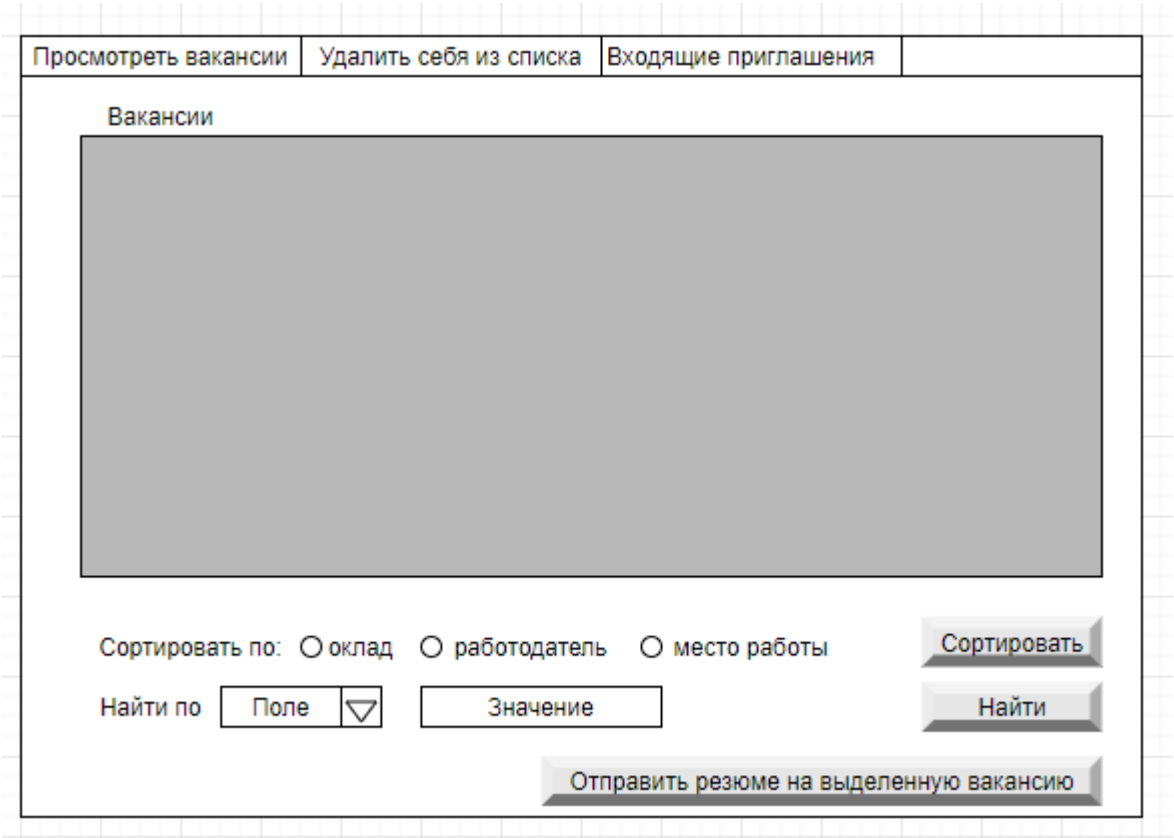

Рисунок 2.13 – Прототип интерфейса выпускника

2.11 Выбор инструментальных средств разработки программы

Для разработки проектируемой системы был выбран язык программирования C#, потому что он имеет такие преимущества как:

- объектно-ориентированный подход;
- хорошая читаемость кода;
- высокая скорость разработки;
- наличие большого количества библиотек;
- удобное использование библиотек;
- строгая типизация;
- совместимость с Windows.

Для разработки взаимодействия с БД была выбрана СУБД PostgreSQL, потому что она имеет такие преимущества как:

- поддержка БД неограниченного размера;
- мощные и надёжные механизмы транзакций и репликации;
- расширяемая система встроенных языков программирования и поддержка загрузки C-совместимых модулей;
- наследование;
- легкая расширяемость;
- возможность создания пользовательских объектов и их поведения.

2.12 Программная реализация базовых модулей системы

Для программной реализации были выбраны модуль обработки команд пользователя (сервер приложений). При написании этого модуля используются такие паттерны как наблюдатель и фасад. Наблюдатель сообщает пользователю об изменении данных в БД. Фасад позволяет использовать набор выделенных в упрощенных команд для взаимодействия пользователя с БД, вместо того, чтобы ему напрямую взаимодействовать с БД. В роли фасада для сервера БД выступает сервер приложений, предоставляющий интерфейс для взаимодействия пользователя и БД. Листинг

кода, в котором представлена реализация выбранных модулей представлен в приложении Б.

2.13 Анализ качества проектирования и оценка программного дизайна

Факторы и метрики, определяющие качество на различных этапах ЖЦ, поддающиеся анализу для группы 5016 – сервисные программы;

Таблица 2.1 – Факторы, критерии и метрики на различных этапах ЖЦ

№	Факторы	Метрики
1	Надёжность	1.Средства восстановления при ошибках на входе
2		2.Средства восстановления при сбоях оборудования
3		3. Реализация управления средствами восстановления
4	Сопровождаемость	1. Простота архитектуры проекта
5		2. Сложность архитектуры проекта
6		3. Межмодульные связи
7	Удобство применения	8. Эксплуатация
8		9. Управление меню
9		10. Функция поддержки справочной системы (помощи HELP)
10		11. Управление данными
11		12. Рабочие процедуры (JOBS)
12	Эффективность	1-10. Функции автоматизации
13	Универсальность	2. Простота архитектуры проекта
14		
15		3. Сложность архитектуры проекта
16	Корректность	3. Непротиворечивость документации разработчика
17		5. Требования, предъявляемые к единообразию интерфейсов между модулями и пользователями
18		6. Требования, предъявляемые к единообразию кодирования, символики и определения общих переменных
19		7. Соответствие документации стандартам

Таблица 2.2 – Оценочные элементы фактора «надёжность»

Код элемента	Наименование	Оценка
Н0101	Наличие требований к программе по устойчивости функционирования при наличии ошибок во входных данных	1
Н0102	Возможность обработки ошибочных ситуаций	1
Н0103	Полнота обработки ошибочных ситуаций	1
Н0104	Наличие тестов для проверки допустимых значений входных данных	0,5
Н0105	Наличие системы контроля полноты входных данных	1
Н0106	Наличие средств контроля корректности входных данных	1
Н0107	Наличие средств контроля непротиворечивости входных данных	1
Н0108	Наличие проверки параметров и адресов по диапазону их значений	1
Н0109	Наличие обработки граничных результатов	1
Н0110	Наличие обработки неопределённостей	0,5
Н0201	Наличие требований к программе по восстановлению процесса выполнения в случае сбоя ОС, процессора, внешних устройств	0
Н0202	Наличие требований к программе по восстановлению результатов при отказах процессора, ОС	0,5
Н0203	Наличие средств восстановления процесса в случае сбоев оборудования	0,5
Н0204	Наличие возможности разделения по времени выполнения отдельных функций программ	0
Н0205	Наличие возможности повторного старта с точки останова	0
Н0301	Наличие централизованного управления процессами, конкурирующими из-за ресурсов	1
Н0302	Наличие возможности автоматически обходить ошибочные ситуации в процессе вычисления	0,5
Н0303	Наличие средств, обеспечивающих завершение процесса решения в случае помех	0,5
Н0304	Наличие средств, обеспечивающих выполнение программы в сокращённом объёме в случае ошибок или помех	0
Н0305	Показатель устойчивости к искажающим воздействиям	0,5

Таблица 2.3 – Оценочные элементы фактора «сопровождаемость»

Код элемента	Наименование	Оценка
С0101	Наличие модульной схемы программы	1
С0102	Оценка программы по числу уникальных модулей	1
С0201	Наличие ограничений на размеры модуля	0,5
С0301	Наличие требований к независимости модулей программы от типов и форматов выходных данных	1
С0302	Наличие проверки корректности передаваемых данных	1
С0303	Оценка простоты программы по числу точек входа и выхода	1
С0304	Осуществляется ли передача результатов работы модуля через вызывающий его модуль	1
С0305	Осуществляется ли контроль за правильностью данных, поступающих в вызывающий модуль от вызываемого	1

Таблица 2.4 – Оценочные элементы фактора «удобство применения»

Код элемента	Наименование	Оценка
У0801	Уровень языка общения пользователя с программой	1
У0802	Лёгкость и быстрота загрузки и запуска программы	1
У0803	Лёгкость и быстрота завершения работы программы	1
У0804	Возможность распечатки содержимого программы	0
У0805	Возможность приостанова и повторного запуска работы без потерь информации	1
У0901	Соответствие меню требованиям пользователя	1
У0902	Возможность прямого перехода вверх и вниз по многоуровневому меню	1
У1001	Возможность управления подробностью получаемых входных данных	1
У1002	Достаточность полученной информации для продолжения работы	1
У1101	Обеспечение удобства ввода данных	1
У1102	Лёгкость восприятия	1
У1201	Обеспечение программой выполнения предусмотренных рабочих процедур	1
У1202	Достаточность информации, выдаваемой программой для составления дополнительных процедур	0.5

Таблица 2.5 – Оценочные элементы фактора «эффективность»

Код элемента	Наименование	Оценка
Э0101	Проблемно-ориентированные функции	0,5
Э0102	Машинно-ориентированные функции	0,5
Э0103	Функции ведения и управления	1
Э0104	Функции ввода/вывода	1
Э0105	Функции защиты и проверки данных	1
Э0106	Функции защиты от несанкционированного доступа	0
Э0107	Функции контроля доступа	0,5
Э0108	Функции защиты от внесения изменений	0,5
Э0109	Наличие соответствующих границ функциональных областей	0,5
Э0110	Число знаков после запятой в результатах вычислений	1
Э0201	Время выполнения программ	1
Э0202	Время реакции и ответов	1
Э0203	Время подготовки	0
Э0205	Затраты времени на защиту данных	0,5
Э0206	Время компиляции	0
Э0301	Требуемый объём внутренней памяти	0,5
Э0302	Требуемый объём внешней памяти	0,5
Э0303	Требуемые периферийные устройства	1
Э0304	Требуемое базовое программное обеспечение	0,5

Таблица 2.6 – Оценочные элементы фактора «универсальность»

Код элемента	Наименование	Оценка
Г0201	Наличие схемы иерархии модулей программы	1
Г0202	Оценка независимости модулей	0,5
Г0203	Оценка числа уникальных элементов/реквизитов	0,5
Г0204	Используется ли в текущем вызове модуля информация, полученная в предыдущем вызове	1
Г0205	Оценка организации точек входа и выхода модуля	0,5
Г0206	Наличие описания атрибутов модуля	0,5
Г0301	Оценка программ по числу переходов и точек ветвления	0,5

Таблица 2.7 – Оценочные элементы фактора «корректность»

Код элемента	Наименование	Оценка
К0301	Отсутствие противоречий в описании частных функций	1
К0302	Отсутствие противоречий в описание основных функций в разных документах	1
К0303	Отсутствие противоречий в описании алгоритмов	1
К0304	Отсутствие противоречий в описании взаимосвязей в системе	1
К0305	Отсутствие противоречий в описании интерфейсов между модулями	1
К0306	Отсутствие противоречий в описании интерфейсов с пользователем	1
К0307	Отсутствие противоречий в описании настройки системы	1
К0308	Отсутствие противоречий в описании иерархической структуры сообщений	1
К0309	Отсутствие противоречий в описании диагностических сообщений	1
К0310	Отсутствие противоречий в описании данных	1
К0501	Единообразие способов вызова модулей	1
К0502	Единообразие процедур возврата управления из модулей	1
К0503	Единообразие способов сохранения информации для возврата	0
К0504	Единообразие способов восстановления информации для возврата	0
К0505	Единообразие организации списков передаваемых параметров	1
К0601	Единообразие наименования каждой переменной и константы	1
К0602	Все ли одинаковые константы встречаются во всех программах под одинаковыми именами	1
К0603	Единообразие определения внешних данных во всех программах	1
К0604	Используются ли разные идентификаторы для разных переменных	0,5
К0605	Все ли общие переменные объявлены как общие переменные	1
К0606	Наличие определений одинаковых атрибутов	0
К0701	Комплектность документации в соответствии со стандартами	1

Продолжение таблицы 2.7

К0702	Правильное оформление частей документов	0,5
К0703	Правильное оформление титульных и заглавных листов документов	0,5
К0704	Наличие в документах всех разделов в соответствии со стандартами	1
К0705	Полнота содержания разделов в соответствии со стандартами	0,5
К0706	Деление документов на структурные элементы	1

По результатам анализа проектирования

2.13.1. Оценка фактора «надёжность».

Определим итоговые значения метрик М1, М2, М3 (в зависимости от номера в таблице этапов ЖЦ).

Метрика М1 «Средства восстановления при ошибках на входе» принимает значение:

М1 = (1+1+1+0,5+1+1+1+1+1+0,5)/10=0,9

Метрика М2 «Средства восстановления при сбоях оборудования» принимает значение:

М2 = (0+0,5+0,5+0+0)/5 = 0,2

Метрика М3 «Реализация управления средствами восстановления» принимает значение:

М3 = (1+0,5+0,5+0+0,5)/5 = 0,5

Итоговая оценка фактора «надёжность» имеет значение:

$R^{Ф}$ = (0,6*0,9 + 0,2*0,2 + 0,2*0,5)/0,65 = 0,68/0,65 = 1.04 > 1

Это означает, что разработчик ПС обеспечил установленный уровень качества выполнения этапа ЖЦ «Проектирование» по надёжности.

2.13.2 Оценка фактора «сопровождаемость»

Определим итоговые значения метрик М1, М2, М3 (в зависимости от номера в таблице этапов ЖЦ).

Определим значения метрик M1:

Метрика M1 «Простота архитектуры проекта» принимает значение:

M1 = (1+1)/2 = 1

Метрика M2 «Сложность архитектуры проекта» принимает значение:

M2 = 0,5

Метрика M3 «Межмодульные связи» принимает значение:

M3 = (1+1+1+1+1) / 5 = 1

Итоговая оценка фактора «сопровождаемость» имеет значение:

$R^{Ф}$ = (0,3*1 + 0,3*0,5 + 0,4*1)/0,7 = 0,85/0,7 = 1,21 > 1

Это означает, что разработчик ПС обеспечил установленный уровень качества выполнения этапа ЖЦ «Проектирования» по сопровождаемости.

2.13.3 Оценка фактора «удобство применения»

Определим значения метрик M8, M9, M10, M11, M12:

Метрика M8 «Эксплуатация» принимает значение:

M8 = (1+1+1+0+1)/5 = 0,8

Метрика M9 «Управление меню» принимает значение:

M9 = (1+1)/2 = 1

Метрика M10 «Функция поддержки справочной системы»:

M10 = (1+1)/2 = 1

Метрика M11 «Управление данными» принимает значение:

M11 = (1+1)/2 = 1

Метрика M12 «Рабочие процедуры (JOBS)» принимает значение:

M12 = (1+0.5)/2 = 0,75

Итоговая оценка фактора «удобство применения» имеет значение:

$R^{Ф}$ = (0,2*0,8+0,2*1+0,2*1+0,2*1+0,2*0,75)/0,8 = 0,926/0,8 = 1,16 > 1

Это означает, что разработчик ПС обеспечил установленный уровень качества выполнения этапа ЖЦ «Проектирование» по удобству применения.

2.13.4 Оценка фактора «эффективность»

Определим значения метрик M1, M2, M3:

Метрика M1 «Функции автоматизации» принимает значение:

M1 = (0,5+0,5+1+1+1+0+0,5+0,5+0,5+1)/10 = 0,65

Метрика M2 «Функции автоматизации» принимает значение:

M2 = (1+1+0+0+0,5)/5 = 0,45

Метрика M3 «Функции автоматизации»:

M3 = (0,5+0,5+1+0,5)/4 = 0,6

Итоговая оценка фактора «эффективность» имеет значение:

$R^{Ф}$ = (0,5*0,65+0,3*0,45+0,2*0,6)/0,57 = 0,58/0,57 = 1.03 < 1

Это означает, что разработчик ПС обеспечил установленный уровень качества выполнения этапа ЖЦ «Проектирование» по эффективности.

2.13.5 Оценка фактора «универсальность»

Определим значения метрик M2, M3:

Метрика M2 «Простота архитектуры проекта» принимает значение:

M2 = (1+0,5+0,5+1+0,5+0,5) / 6 = 0,66

Метрика M3 «Сложность архитектуры проекта»:

M3 = 0,5

Итоговая оценка фактора «универсальность» имеет значение:

$R^{Ф}$ = (0,5 * 0,66 + 0,5 * 0,5) / 0,55 = 0,58/0,55 = 1,05 > 1

Это означает, что разработчик ПС обеспечил установленный уровень качества выполнения этапа ЖЦ «Проектирование» по универсальности.

2.13.6 Оценка фактора «корректность»

Определим значения метрик M3, M5, M6, M7:

Метрика M3 «Непротиворечивость документации разработчика» принимает значение:

M3 = (1+1+1+1+1+1+1+1+1+1) / 10 = 1

Метрика М5 «Требования, предъявляемые к единообразию интерфейсов между модулями и пользователями» принимает значение:

М5 = (1+1+0+0+1) / 5 = 0,6

Метрика М6 «Требования, предъявляемые к единообразию кодирования, символики и определения общих переменных»:

М6 = (1+1+1+0,5+1+0) / 6 = 0,75

Метрика М7 «Соответствие документации стандартам» принимает значение:

М7 = (1+0,5+0,5+1+0,5+1) / 6 = 0,75

Итоговая оценка фактора «корректность» имеет значение:

$R^{Ф}$ = (0,2 * 1 + 0,2 * 0,6 + 0,3 * 0,75 + 0,3 * 0,75) / 0,7 = 1,1 > 1

Это означает, что разработчик ПС обеспечил установленный уровень качества выполнения этапа ЖЦ «Проектирование» по корректности.

По результатам проведенного анализа можно заключить, что система спроектирована успешно. Разработчик обеспечил необходимый уровень качества для системы на этапе проектирования по факторам: надежность, сопровождаемость, удобство применения, эффективность, универсальность, корректность. Для проектируемой системы, которая относится к группе сервисных программ, показатели «Удобство применения» и «Корректность» являются наиболее значимыми.

3 ПРОГРАММНАЯ РЕАЛИЗАЦИЯ МОДУЛЯ ПРОЕКТА

3.1 Описание используемых паттернов

3.1.1 Наблюдатель

Паттерн наблюдатель – это поведенческий паттерн проектирования, который используется для того, чтобы оповещать множество объектов при изменениях в других объектах.

В проектируемой системы паттерн наблюдатель помогает пользователю поддерживать актуальность данных, с которыми он работает, путем оповещения его об изменениях, происходящих в БД.

3.1.2 Фасад

Паттерн фасад – это структурный паттерн проектирования, который представляет упрощенный интерфейс для работы со сложной системой.

В проектируемой системе паттерн фасад позволяет пользователю взаимодействовать не непосредственно с БД, а с сервером приложений, который предоставляет определенный набор команд, необходимых пользователю.

3.2 Структура используемых паттернов

3.2.1 Наблюдатель

Диаграмма классов для паттерна наблюдатель приведена на рисунке 3.1.

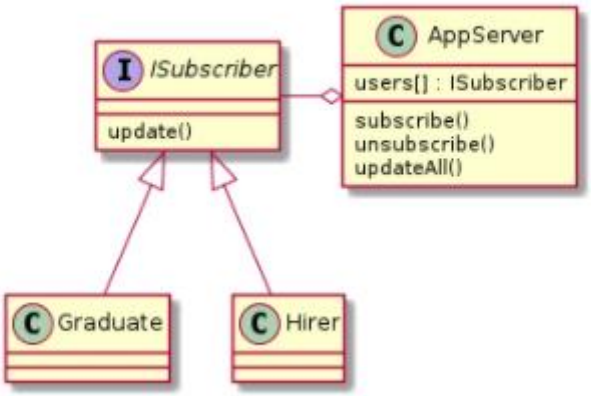

Рисунок 3.1 – Диаграмма классов для паттерна наблюдатель

AppServer – это класс, который владеет информацией об изменениях в БД. Он хранит список пользователей, которые будут получать уведомление об изменениях данных в БД. В его состав входят методы, позволяющие пополнять и сокращать список оповещаемых пользователей, а также сам метод оповещения пользователей. Классы пользователей должны реализовывать интерфейс ISubscriber.

3.2.2 Фасад

Диаграмма классов для паттерна фасад приведена на рисунке 3.2.

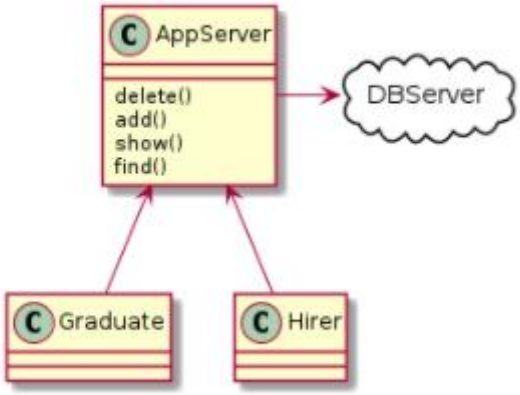

Рисунок 3.2 – Диаграмма классов для паттерна фасад

AppServer предоставляет упрощенный интерфейс для пользователей, а также позволяет пользователю взаимодействовать пользователю с данными системы без прямого взаимодействия со сложной внутренней структурой программы. Пользователи могут использовать методы, которые предоставляет им AppServer.

После внедрения паттернов наблюдатель и фасад в систему общая диаграмма классов изменится (см. рис. 3.3)

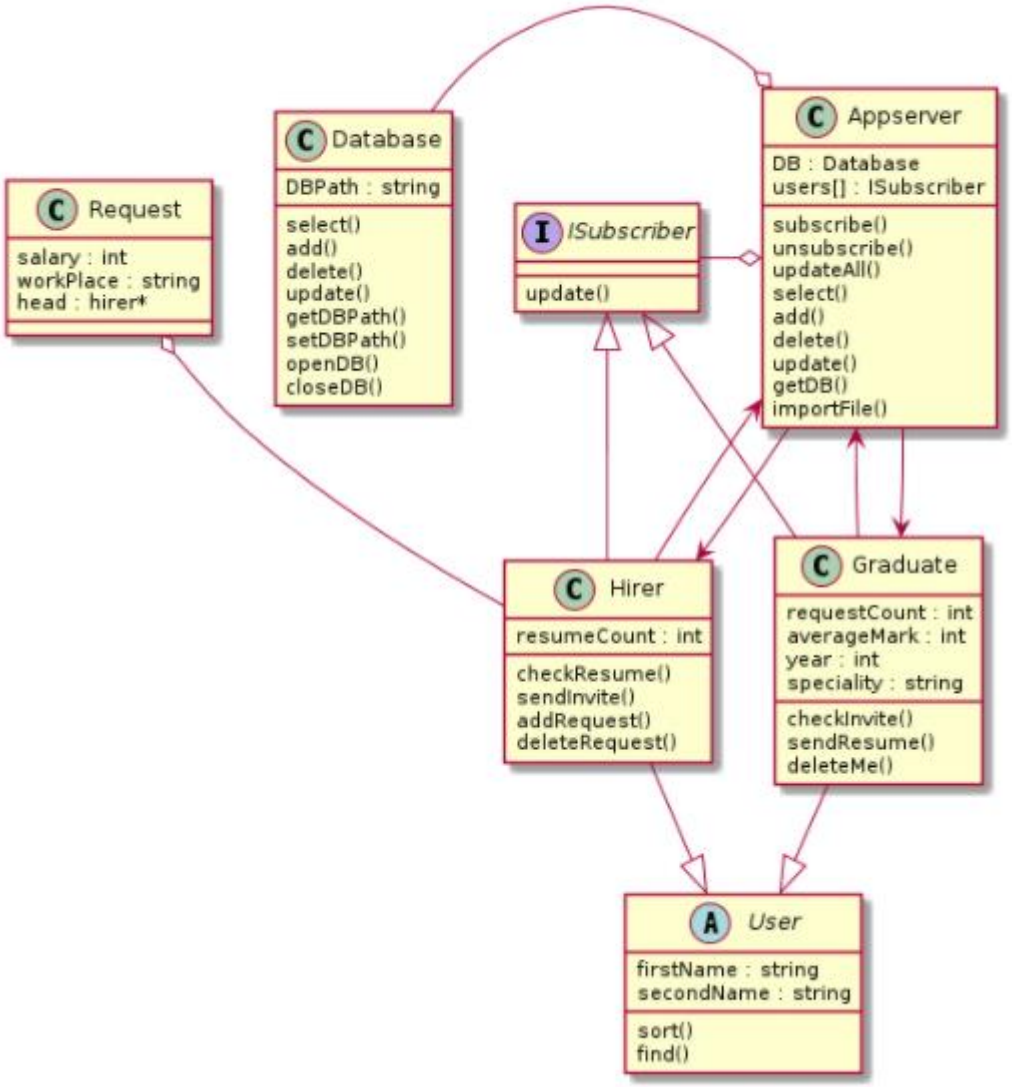

Рисунок 3.3 – Финальная диаграмма классов

3.3 Программный код паттернов

3.3.1 Наблюдатель

Код связанный с реализацией паттерна наблюдатель представлен на рисунках 3.4 – 3.7.

```
interface ISubscriber
{
    void update(string message);
}
```

Рисунок 3.4 – Код интерфейса ISubscriber

```
class Graduate : ISubscriber
{
    public string lastOperation;
    public void update(string message)
    {
        // Отправить повторный запрос на требуемые данные
        lastOperation = message;
    }
}
```

Рисунок 3.5 – Код класса Hirer

```
partial class Hirer : ISubscriber
{
    public string lastOperation;
    public void update(string message)
    {
        // Отправить повторный запрос на требуемые данные
        lastOperation = message;
    }
}
```

Рисунок 3.6 – Код класса Graduate

```
class AppServer
{
    List<ISubscriber> subscribers;
    string message;
    public AppServer()
    {
        subscribers = new List<ISubscriber>();
    }
    public void subscribe(ISubscriber subscriber)
    {
        subscribers.Add(subscriber);
    }

    public void unsubscribe(ISubscriber subscriber)
    {
        subscribers.Remove(subscriber);
    }

    public void updateAll()
    {
        foreach (var subscriber in subscribers)
        {
            subscriber.update(message);

        }
    }
```

Рисунок 3.7 – Код паттерна наблюдатель в классе AppServer

3.3.2 Фасад

Код реализации паттерна фасад представлен на рисунке 3.8

```
public void delete()
{
    // Проверка корректности данных
    // Удаление данных из БД
    message = "Произведено удаление из БД. Данные будут обновлены.";
    updateAll();
}

public void add()
{
    // Проверка корректности данных
    // Добавление данных в БД
    message = "Произведено добавление в БД. Данные будут обновлены.";
    updateAll();
}

public void show()
{
    // Проверка корректности данных
    // Выполнение запроса в БД
    // Прием результата запроса
    // Отправка результата запроса пользователю
}

public void find()
{
    // Проверка корректности данных
    // Выполнение запроса в БД
    // Прием результата запроса
    // Отправка результата запроса пользователю
}
```

Рисунок 3.8 – Код паттерна фасад в классе AppServer

ЗАКЛЮЧЕНИЕ

При выполнении курсового проекта была проанализирована предметная область проекта, изучены и проанализированы назначение, требования к программным системам предложенной предметной области. Выполнено проектирование системы, удовлетворяющей всем указанным требованиям.

Также при проектировании системы были получены теоретические знания и практические навыки в области архитектурного планирования и проектирования ПО.

Выполнен анализ качества таких этапов проектирования как: анализ требований к системе, проектирование системы.

Для улучшения структуры системы были использованы такие паттерны проектирования как: наблюдатель, фасад.

Результатом курсового проекта является спроектированное программное обеспечение для информационной системы «Кадровое агентство выпускников вуза», а также частично реализованный программный продукт с реализацией выбранных паттернов.

Спроектированная система рекомендуется к использованию для упрощения поиска вакансий для студентов-выпускников вуза, а также работодателям для поиска потенциальных работников.

СПИСОК ЛИТЕРАТУРЫ

1. "Основы программной инженерии" Copyright © 2004-2010 Сергей Орлик. SWEBOK Copyright © 2004 by The Institute of Electrical and Electronics Engineers, Inc.

2. Басс Л., Клементс П., Кацман Р. Архитектура программного обеспечения на практике. 2-е изд. – СПб.: Питер, 2006. – 575 с.

3. Леоненков Г. Самоучитель по UML. 4-е изд. – СПб.: Питер 2003. – 321 с.

4. Брауде Э. Технология разработки программного обеспечения: пер. с англ. / Э. Брауде. СПб.: Питер, 2004. - 655 с.

ПРИЛОЖЕНИЕ А. ТЕХНИЧЕСКОЕ ЗАДАНИЕ

ПРИЛОЖЕНИЕ Б. ФУНКЦИОНАЛЬНАЯ МОДЕЛЬ

Описание системы с помощью IDEF0 называется функциональной моделью. Функциональная модель предназначена для описания существующих бизнес-процессов, в котором используются как естественный, так и графический языки. Для передачи информации о конкретной системе источником графического языка является методология IDEF0 (см. рис. Б.1).

Для построения графического представления необходимо произвести декомпозицию процесса. Например, для процедуры добавления новой вакансии необходимо сначала проверить корректность запроса, затем, если запрос верный, добавить новую вакансию в БД (см. рис. Б.1).

Рисунок Б.1 – Функциональная модель

ПРИЛОЖЕНИЕ В. МЕТРИКА ДИАГРАММ UML

Оценка метрики диаграмм – это процесс, который позволяет оценить сложность и информативность диаграммы при помощи присвоения отдельным элементам диаграмм численных значений, а затем вычисления общей оценки диаграммы по приведенным ниже формулам.

В зависимости от типа диаграммы оценка отдельных элементов может меняться, однако формула вычисления общей оценки диаграммы всегда одинакова:

$$S = \frac{\sum S_{obj} + \sum S_{lnk}}{1 + Obj + \sqrt{T_{ob} + T_{lnk}}}, \quad \text{(B.1)}$$

где S - оценка диаграммы;

Sobj - оценки для элементов диаграммы;

Slnk - оценки для связи на диаграмме;

Obj - число объектов на диаграмме;

Tobj - число типов объектов на диаграмме;

Tlnk - число типов связи на диаграмме.

Если на диаграмме классов показаны атрибуты и операции класса, можно учесть их при расчете, при этом оценка прибавляется к оценке соответствующего класса

$$S_{cls} = \frac{\sqrt{Op} + \sqrt{Atr}}{0.3 * (Op + Atr)}, \quad \text{(B.2)}$$

где Scls - оценка операций и атрибутов для класса;

Op – число операций в классе;

Atr - число атрибутов класса.

Оценки для различных типов элементов и связей приведены в таблицах В.1 – В.2.

Таблица В.1 — Оценка элементов UML

Типы элемента	Оценка для элемента
Класс	5
Интерфейс	4
Вариант использования	2
Компонент	4
Узел	3
Процессор	2
Взаимодействие	6
Пакет	4
Состояние	4
Примечание	10

Таблица В.2 — Оценка типов связей UML

Тип связи	Оценка
Зависимость	2
Ассоциация	1
Агрегирование	2
Композиция	3
Обобщение	3
Реализация	2

Типа связей, не вошедшие в таблицу В.2 должны рассматриваться как ассоциации.

Недостатком диаграммы является как слишком низкая оценка (при этом диаграмма недостаточно информативна), так и высокая (при этом диаграмма обычно слишком сложна для понимания). Диапазон оптимальных оценок для основных типов диаграмм приведен в таблице В.3.

Таблица В.3 — Диапазон оценок для диаграмм UML

Тип диаграммы	Диапазон оценок
Диаграмма классов с атрибутами и операциями	5 - 5.5
Диаграмма без атрибутов и операций	3 - 3.5
Диаграммы компонентов	3.5 - 4
Диаграммы вариантов использования (USE CASE)	2.5 - 3
Диаграммы размещения	2 - 2.5

Продолжение таблицы В.3

Диаграммы последовательности	3 - 3.5
Кооперативная диаграмма	3.5 - 4
Диаграммы пактов	3.5 - 4
Диаграммы состояний	2.5 - 3

ПРИЛОЖЕНИЕ Г. ФРАГМЕНТЫ ЛИСТИНГА

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
namespace AiPPO_Kursovaya_Morgunov
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }
    }
}
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace AiPPO_Kursovaya_Morgunov{
    class AppServer    {
        List<ISubscriber> subscribers;
        string message;
        public AppServer()        {
            subscribers = new
List<ISubscriber>();        }
        public void
subscribe(ISubscriber subscriber)
{

subscribers.Add(subscriber); }

        public void
unsubscribe(ISubscriber subscriber)   {

subscribers.Remove(subscriber);
        }

        public void updateAll()
        {
            foreach (var subscriber in
subscribers)
            {
        subscriber.update(message);

            }
        }

        public void delete()
        {
            // Проверка корректности
данных
            // Удаление данных из БД
            message = "Произведено
удаление из БД. Данные будут
обновлены.";
            updateAll();
        }

        public void add()
        {
            // Проверка корректности
данных
            // Добавление данных в БД
            message = "Произведено
добавление в БД. Данные будут
обновлены.";
            updateAll();
        }

        public void show()
        {
            // Проверка корректности
данных
            // Выполнение запроса в БД
            // Прием результата запроса
            // Отправка результата
запроса пользователю
        }

        public void find()
        {
            // Проверка корректности
данных
            // Выполнение запроса в БД
            // Прием результата запроса
            // Отправка результата
запроса пользователю
        }

    }
}


using System;
using System.Collections.Generic;
using System.Linq;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace AiPPO_Kursovaya_Morgunov
{
    static class Program
    {
        /// <summary>
        /// Главная точка входа для
приложения.
        /// </summary>
        [STAThread]
        static void Main()
```

```
        {
            Application.EnableVisualStyles();
            Application.SetCompatibleTextRenderingDefault(false);
            Application.Run(new Form1());
        }
    }
}
```

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace AiPPO_Kursovaya_Morgunov
{
    class Graduate : ISubscriber
    {
        public string lastOperation;
        public void update(string message)
        {
            // Отправить повторный запрос на требуемые данные
            lastOperation = message;
        }
    }
}
```

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace AiPPO_Kursovaya_Morgunov
{
    partial class Hirer : ISubscriber
    {
        public string lastOperation;
        public void update(string message)
        {
            // Отправить повторный запрос на требуемые данные
            lastOperation = message;
        }
    }
}
```

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace AiPPO_Kursovaya_Morgunov
{
    interface ISubscriber
    {
        void update(string message);
    }
}
```