

2020 г.

Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Московский государственный технический университет имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

УТВЕРЖДАЮ

Заведующий кафедрой ИУ6



А.В. Пролетарский

« 2 » марта 2020 г.

ЗАДАНИЕ
на выполнение выпускной квалификационной работы бакалавра

Студент группы ИУ6-83Б

Шашкин Юрий Анатольевич
(Фамилия, имя, отчество)

Тема квалификационной работы Система контроля знаний языка запросов к базе данных

Источник тематики (НИР кафедры, заказ организаций и т.п.)

инициативная НИР кафедры

Тема квалификационной работы утверждена распоряжением по факультету ИУ № 03.02.01-04.03/25 от « 13 » ноября 2019 г.

Часть I. Исследовательская

Выполнить анализ аналогов и уточнить набор функций программной системы. Осуществить выбор наиболее приоритетных функциональных особенностей аналогов для дополнения разрабатываемой системы.

Часть 2. Конструкторская

Выполнить проектирование программной системы. Разработать спецификацию функциональную, функциональную модель IDEF0 программной системы, схемы алгоритмов создания объектов базы данных и сохранения задания, диаграммы классов предметной области, схему структуры базы данных. Разработать интерфейс программной системы. Выполнить отладку и тестирование компонентов программной системы. Выполнить комплексное тестирование программной системы. Выполнить оценочное тестирование программной системы. _____

Часть 3. Технологическая

Разработать технологию тестирования программной системы и привести примеры тестов. Рассмотреть различные варианты дальнейшего развития программного продукта, возможности интеграции с учебной платформой Moodle. _____

Оформление квалификационной работы:

Расчетно-пояснительная записка на 55–65 листах формата А4.

Перечень графического (иллюстративного) материала (чертежи, плакаты, слайды и т.п.)

1. Спецификация функциональная.
2. Функциональная модель IDEF0 программной системы.
3. Схемы алгоритмов создания объектов базы данных и сохранения задания.
4. Диаграмма классов предметной области.
5. Схема структуры базы данных.
6. Результаты тестирования, выводы, перспективы развития.

Дата выдачи задания « 4 » сентября 2019 г.

В соответствии с учебным планом выпускную квалификационную работу выполнить в полном объеме в срок до « 1 » июня 2020 г.

Руководитель квалификационной работы

 20.09.2019 М.М. Фомин
(Подпись, дата) (И.О. Фамилия)

Студент

 - 20.09.2020 Ю.А. Шашкин
(Подпись, дата) (И.О. Фамилия)

Примечание: 1. Задание оформляется в двух экземплярах: один выдается студенту, второй хранится на кафедре.

Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Московский государственный технический университет имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ Информатика и системы управления

УТВЕРЖДАЮ

КАФЕДРА Компьютерные системы и сети

Заведующий кафедрой ИУ6

ГРУППА ИУ6-83Б

А.В. Пролетарский
« 2 » марта 2020 г.

КАЛЕНДАРНЫЙ ПЛАН

выполнения выпускной квалификационной работы бакалавра

студента: Шашкина Юрия Анатольевича
(фамилия, имя, отчество)

Тема квалификационной работы Система контроля знаний языка запросов к базе данных

№ п/п	Наименование этапов выпускной квалификационной работы	Сроки выполнения этапов		Отметка о выполнении	
		план	факт	Должность	ФИО, подпись
1.	Задание на выполнение работы. Формулирование проблемы, цели и задач работы	09.2019 <small>Планируемая дата</small>	09.2019	Руководитель ВКР	М.М. Фомин
2.	1 часть Исследовательская	12.2019 <small>Планируемая дата</small>	12.2019	Руководитель ВКР	М.М. Фомин
3.	Утверждение окончательных формулировок решаемой проблемы, цели работы и перечня задач	02.2020 <small>Планируемая дата</small>	02.2020	Заведующий кафедрой	А.В. Пролетарский
4.	2 часть Конструкторская	05.2020 <small>Планируемая дата</small>	05.2020	Руководитель ВКР	М.М. Фомин
5.	3 часть Технологическая	05.2020 <small>Планируемая дата</small>	05.2020	Руководитель ВКР	М.М. Фомин
6.	1-я редакция работы	05.2020 <small>Планируемая дата</small>	05.2020	Руководитель ВКР	М.М. Фомин
7.	Подготовка доклада и презентации	06.2020 <small>Планируемая дата</small>	06.2020	Руководитель ВКР	М.М. Фомин
8.	Заключение руководителя	06.2020 <small>Планируемая дата</small>	06.2020	Руководитель ВКР	М.М. Фомин
9.	Нормоконтроль	06.2020 <small>Планируемая дата</small>	06.2020	Нормоконтролер	
10.	Внешняя рецензия	06.2020 <small>Планируемая дата</small>	06.2020	Руководитель ВКР	М.М. Фомин
11.	Защита работы на ГЭК	06.2020 <small>Планируемая дата</small>	06.2020	Руководитель ВКР	М.М. Фомин

Студент  2.03.2020
(подпись, дата)

Руководитель работы  2.03.2020
(подпись, дата)

АННОТАЦИЯ

В настоящей выпускной работе бакалавра описан процесс разработки системы контроля знаний языка запросов к базе данных. В ходе работы был проведен анализ существующих аналогов данного веб-сервиса, выделены основные технические характеристики, необходимые для успешного запуска данной системы.

По результатам анализа спроектирована и разработана система контроля знаний языка запросов к базе данных.

Проведено всестороннее тестирование программной системы и проанализированы результаты проведенного тестирования, сделаны выводы. Представлены перспективы развития программного продукта в будущем – развитие других видов тестирования и интеграция с системой электронного образования Moodle.

Abstract

This bachelor's final qualification paper describes the process of developing the system for controlling the knowledge of structured query language. The existing analogs were analyzed during the work. There were found the most important parts of the future system for launch it.

The system was designed and developed on the base of analytics.

According to this was developed the testing technology. Some prospect of development of the system were presented.

РЕФЕРАТ

Записка 82 с., 59 рис., 6 табл., 15 ист., 4 прил.

ЗАПРОСЫ, SQL, ТЕХНИЧЕСКОЕ СОБЕСЕДОВАНИЕ, СИСТЕМА КОНТРОЛЯ ЗНАНИЙ.

Целью работы является веб-сервис СКЗБД для проведения технических собеседований, связанных с оценкой знаний кандидата по базам данных.

В результате работы были проанализированы аналоги, выполнены проектирование, разработка и тестирование системы. Представлены перспективы развития данного продукта.

Пользователями СКЗБД являются менеджеры, проводящие технические собеседования в ИТ-компаниях с целью определения кандидатов, обладающих достаточными знаниями для получения должности, связанной с базами данных.

СОДЕРЖАНИЕ

ОПРЕДЕЛЕНИЯ, ОБОЗНАЧЕНИЯ И СОКРАЩЕНИЯ.....	8
ВВЕДЕНИЕ.....	9
1 АНАЛИЗ АНАЛОГОВ И ВЫБОР ОСНОВНЫХ ФУНКЦИОНАЛЬНЫХ ОСОБЕННОСТЕЙ.....	10
1.1 ВЫБОР ОБЪЕКТА АНАЛИЗА	10
1.2 АНАЛИЗ И ТЕСТИРОВАНИЕ ФУНКЦИОНАЛЬНОСТИ ДЛЯ КАНДИДАТА.....	12
1.2.1 Блок задания	13
1.2.2 Блок ввода решения	15
1.2.3 Блок результатов	16
1.2.4 Блок дополнительных функций	18
1.3 АНАЛИЗ И ТЕСТИРОВАНИЕ ФУНКЦИОНАЛЬНОСТИ ДЛЯ МЕНЕДЖЕРА.....	18
1.3.1 Блок управления.....	19
1.3.2 Блок информации о задании	20
1.3.3 Блок задания	21
1.3.4 Блок решения.....	23
1.3.5 Блок тестов	25
1.4 ВЫБОР НЕОБХОДИМЫХ ФУНКЦИЙ СКЗБД.....	28
2 ПРОЕКТИРОВАНИЕ И РАЗРАБОТКА СИСТЕМЫ	31
2.1 РАЗРАБОТКА ФУНКЦИОНАЛЬНОЙ СПЕЦИФИКАЦИИ.....	31
2.2 ФУНКЦИОНАЛЬНАЯ ДИАГРАММА IDEF0	32
2.2.1 Функциональная модель с точки зрения ресурс-менеджера.....	32
2.2.2 Функциональная модель с точки зрения кандидата.....	40
2.3 ПРОЕКТИРОВАНИЕ СТРУКТУРЫ БАЗЫ ДАННЫХ	42
2.3.1 Концептуальное проектирование.....	43
2.3.2 Логическое проектирование	44
2.3.3 Физическое проектирование.....	46
2.4 ВЫБОР ЯЗЫКА И СРЕДСТВ РАЗРАБОТКИ	47
2.5 РАЗРАБОТКА АЛГОРИТМОВ РАБОТЫ СИСТЕМЫ.....	48
2.5.1 Алгоритм создания объектов базы данных.....	49
2.5.2 Алгоритм сохранения задания.....	51
2.6 РАЗРАБОТКА ФОРМ ИНТЕРФЕЙСА	54
2.7 ОПИСАНИЕ КОМПОНЕНТОВ СИСТЕМЫ.....	58
2.7.1 Выбор архитектурного паттерна системы	58
2.7.2 Работа с БД	59
2.7.3 Проектирование классов системы.....	59
2.7.4 Настройка отправки почты	66
2.7.5 Разработка функций IDE для форм ввода	67
3 ТЕСТИРОВАНИЕ И ПЕРСПЕКТИВЫ РАЗВИТИЯ СИСТЕМЫ	69

3.1 ТЕСТИРОВАНИЕ	69
3.1.1 Автоматическое тестирование	70
3.1.2 Тестирование безопасности	70
3.1.3 Тестирование интерфейса пользователя	72
3.1.4 Тестирование компонентов SQL	74
3.2 ПЕРСПЕКТИВЫ РАЗВИТИЯ	78
3.2.1 Развитие других видов тестирования	78
3.2.2 Интеграция с Moodle	79
ЗАКЛЮЧЕНИЕ	81
СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ	82
ПРИЛОЖЕНИЕ А	84
ПРИЛОЖЕНИЕ Б	94
ПРИЛОЖЕНИЕ В	103
ПРИЛОЖЕНИЕ Г	114

ОПРЕДЕЛЕНИЯ, ОБОЗНАЧЕНИЯ И СОКРАЩЕНИЯ

SQL (Structured Query Language) - декларативный язык программирования, применяемый для создания, модификации и управления данными в реляционной базе данных, управляемой соответствующей системой управления базами данных.

IDE (Integrated Development Environment) — система программных средств, используемая программистами для разработки программного обеспечения.

ORM (англ. Object-Relational Mapping, рус. объектно-реляционное отображение, или преобразование) — технология программирования, которая связывает базы данных с концепциями объектно-ориентированных языков программирования, создавая «виртуальную объектную базу данных».

REST (RESTful) - это общие принципы организации взаимодействия приложения/сайта с сервером посредством протокола HTTP. Особенность REST в том, что сервер не запоминает состояние пользователя между запросами - в каждом запросе передаётся информация, идентифицирующая пользователя (например, token, полученный через OAuth-авторизацию) и все параметры, необходимые для выполнения операции.

API (англ. application programming interface) — описание способов (набор классов, процедур, функций, структур или констант), которыми одна компьютерная программа может взаимодействовать с другой программой.

ВВЕДЕНИЕ

В современных ИТ-компаниях проектные менеджеры достаточно большое количество времени тратят на собеседования, в том числе технические. Подготовка задания, его тестирование, проработка возможных ответов – это все занимает время менеджера, которое могло быть потрачено на развитие проектов компании.

Кроме того, техническая часть собеседования часто проводится на листочке, что дает только поверхностное понимание знаний кандидата, так как в рабочем процессе будет возможность использовать интернет ресурсы, собственные наработки – это все может значительно повысить производительность человека и некорректно лишать кандидата этих возможностей.

Данные проблемы могут быть решены использованием системы контроля знаний, которая позволит менеджерам создавать задания в онлайн форме, сохранять их, а затем, при необходимости, повторно использовать. Кандидату такая система позволит показать свои знания, используя все возможные источники информации, которые могут быть использованы в реальной работе.

1 Анализ аналогов и выбор основных функциональных особенностей

1.1 Выбор объекта анализа

Существует большое количество систем, которые можно рассматривать, как системы контроля знаний. Тестирование может проводиться в различных форматах: от простых тестов до написания сложных скриптов вручную.

Среди русскоязычных ресурсов можно выделить ресурсы «geekbrains» и «proghub». Данные ресурсы позволяют пройти тестирование на знания как языка запросов, так и теоретических основ реляционных баз данных.

Так как оба ресурса являются аналогичными по функциональности, рассмотрим только один из них. На рисунке 1 представлен пример тестового задания с ресурса «Geekbrains» [1].

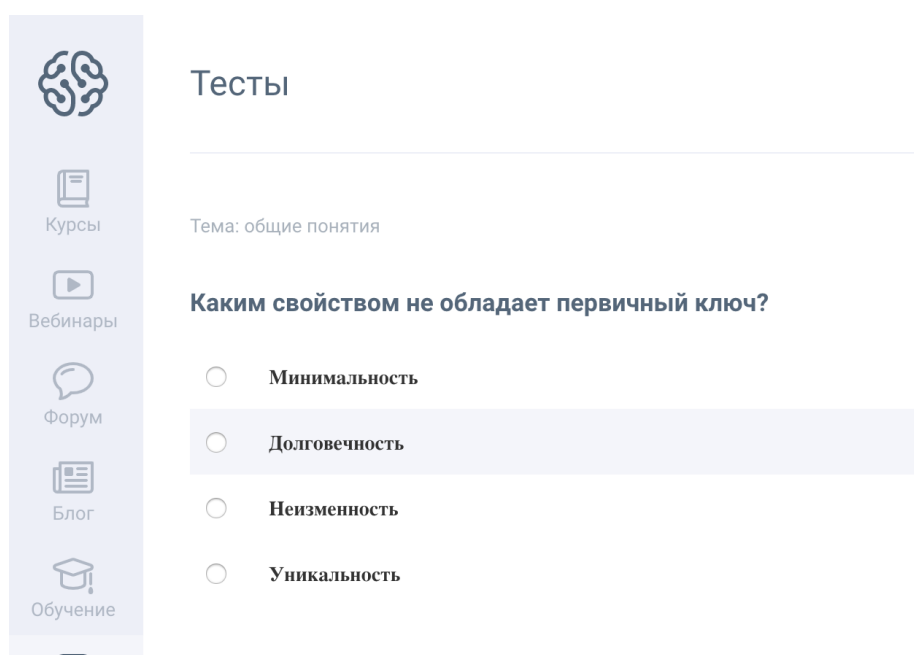


Рисунок 1 – Пример тестового задания с «GeekBrains»

Данный портал позволяет проходить заранее подготовленный тест, который не может быть изменен при необходимости. Задания делятся по темам и могут быть как тестовыми, так и с развернутым ответом. Задания с развернутым ответом являются самыми простейшими и могут быть решены в одну строчку кода. По причине того, что данный ресурс не дает возможности

изменять тесты и решать сложные задания с развернутым ответом, то он не подходит для анализа.

Среди англоязычных ресурсов можно выделить популярный портал «codewars» [2]. Данный портал используется для практики программирования и предоставляет возможности как для обучения программированию на различных языках, так и для проведения собеседований. Версия для проведения собеседований является платной и может быть использована только в коммерческих целях. По информации из открытых источников – функциональность версии для собеседований аналогична функциональности для обучения программированию. Следовательно, вся необходимая функциональность может быть протестирована в бесплатной версии.

Обобщенные критерии сравнения обоих ресурсов с целью анализа функциональности для проведения собеседований представлены в таблице 1 [3].

Таблица 1 – Сравнение «GeekBrains» и «CodeWars»

Критерий	GeekBrains	CodeWars
Задания в формате теста	Да	Нет
Простые задания с написанием запроса в 1-4 строки	Да	Да
Сложные задания с написанием запроса в 5 и более строк	Да	Да
Задания с добавлением картинки	Нет	Нет
Возможность исправления задания	Нет	Да
Возможность добавления заданий	Нет	Да
Возможность отправки заданий по почте	Нет	Нет

Продолжение таблицы 1

Возможность ограничения времени решения конкретного задания/теста в целом	Да	Да
Возможность получения результатов решения (для менеджера)	Нет	Да
Возможность выбора СУБД	Нет	Да
Проверка синтаксиса в заданиях с развернутым ответом	Нет	Да
Возможность сбора статистики по тесту	Нет	Да

По данным из таблицы 1 можно сделать вывод, что портал «CodeeWars» имеет большую часть необходимых для нас функций, поэтому для анализа будем использовать именно его.

1.2 Анализ и тестирование функциональности для кандидата

Типовое окно для тестируемого представлено на рисунке 2.

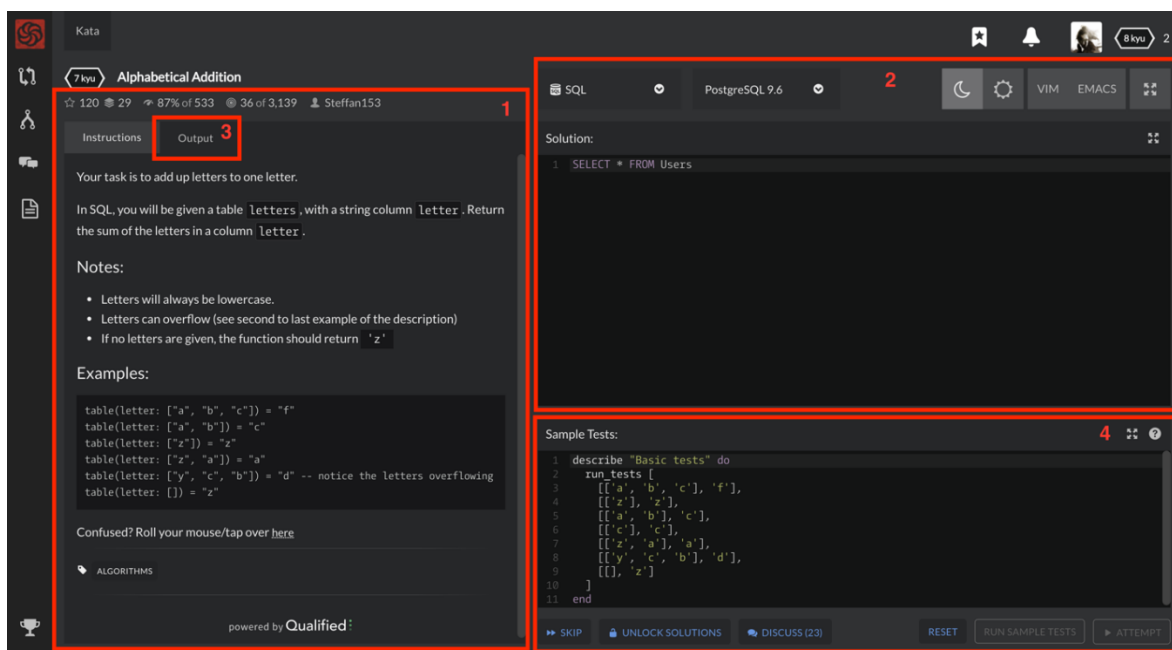


Рисунок 2 – Типовое окно тестируемого на портале «codewars»

Анализ «codewars» для кандидатов будет производиться по следующим пунктам:

- блок текста задания (1);
- блок ввода решения кандидата (2);
- блок результатов, который вызывается при запуске решения и содержит результаты запуска (3);
- блок дополнительных функций (4).

Каждый из данных блоков может быть рассмотрен отдельно.

1.2.1 Блок задания

Блок задания представлен на рисунке 3.

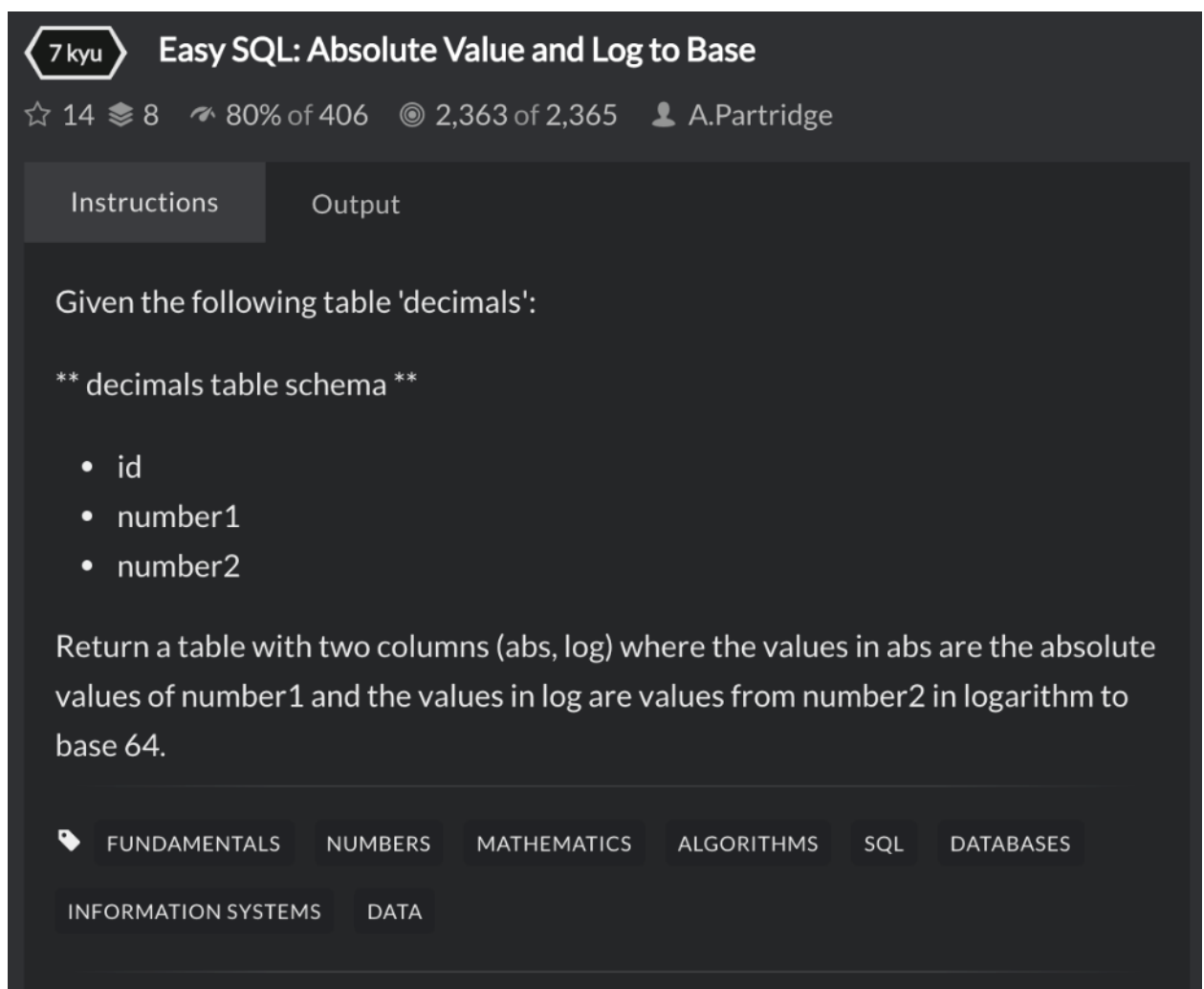


Рисунок 3 – Блок задания

Функционально данный блок можно разделить на 3 основные части: информация о задании, задание и теги задания. Рассмотрим каждую из них.

Информация о задании содержит:

- сколько раз задание добавлено в закладки,
- сколько раз задание добавлено в коллекции,
- пользовательский рейтинг,
- число успешных запусков,
- автор.

В ходе прочтения данной информации можно получить примерное представление о задании и о том, как его восприняло сообщество.

Самая большая часть в данном блоке отведена заданию. На примере, представленном на рисунке 3, задание представлено в текстовом виде и содержит информацию о рассматриваемом объекте и постановку вопроса. Задание может содержать особые выделенные структуры, которые хочет выделить автор (ключевые слова, примеры, куски кода). Пример с выделением таких структур представлен на рисунке 4.

Нижняя часть данного блока выделена для тегов задания. Теги могут быть использованы для сортировки задания (когда, например, пользователь хочет уделить особое внимание алгоритмам) и поиска.

Для проведения собеседований важными блоками будут автор и само задание. Информацию об удачных запусках, пользовательскому рейтингу задания, тегах, возможность добавлять в закладки и коллекции задание кандидату не нужно – это является избыточной для него информацией. Кроме того, для заданий на SQL было бы хорошо, если бы была возможность вставлять изображения в задания (например, схему БД).

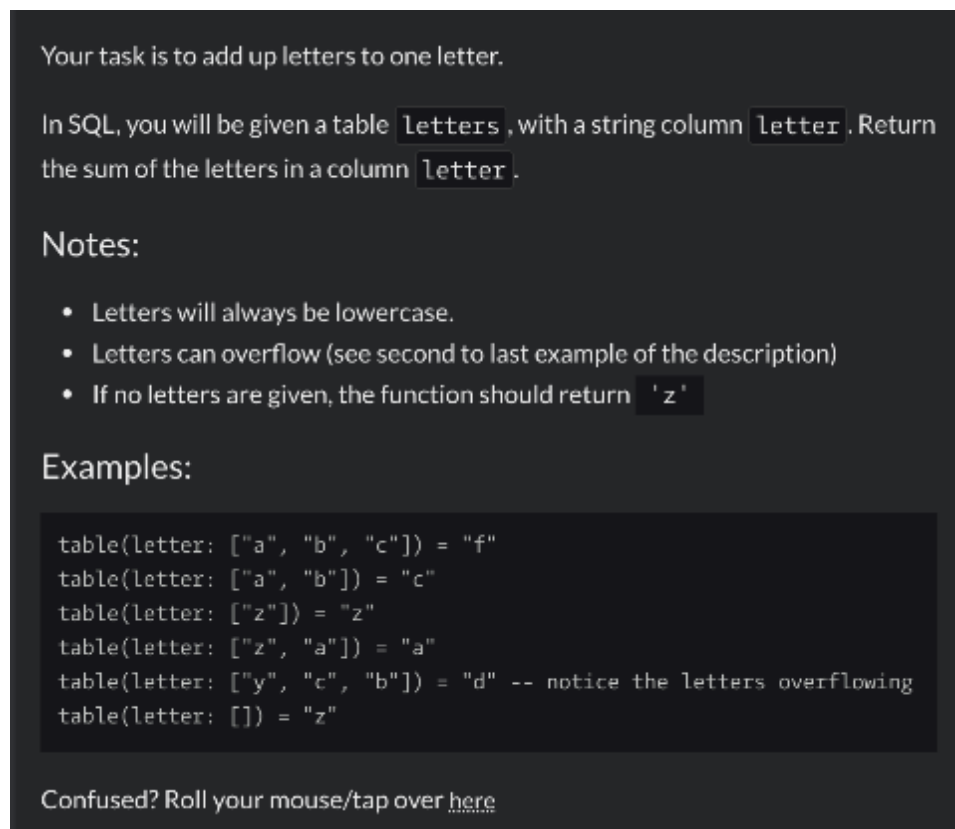


Рисунок 4 – Пример задания с выделенными структурами

1.2.2 Блок ввода решения

Пример блока ввода решения представлен на рисунке 5

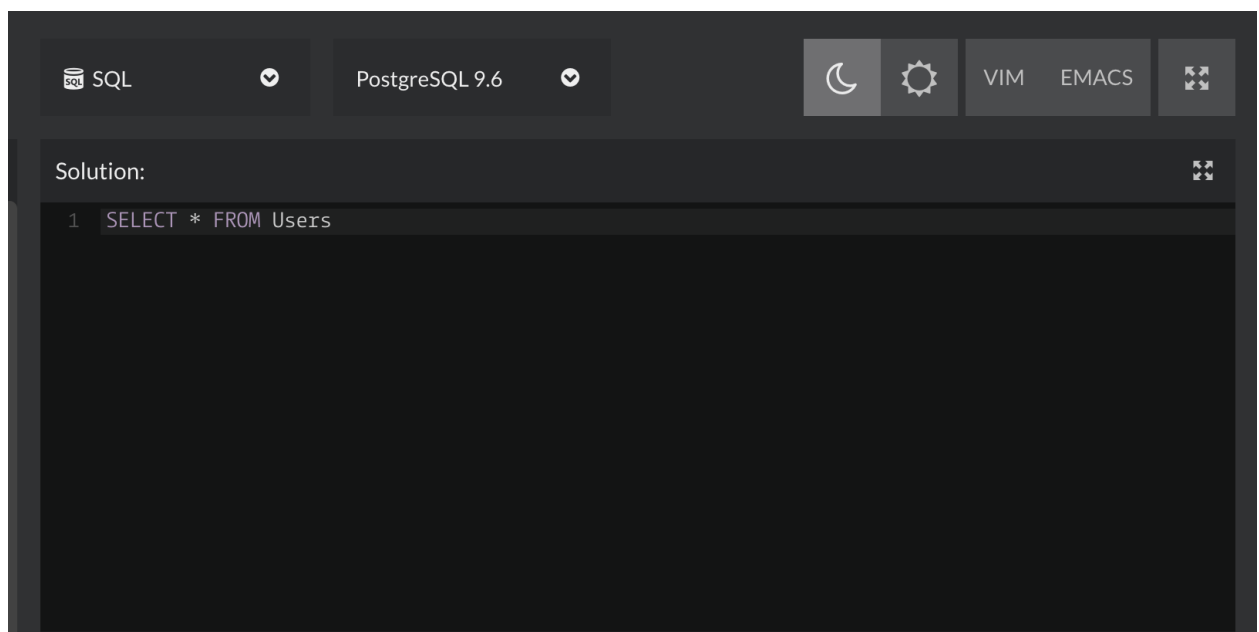


Рисунок 5 – Блок ввода решения

Данный блок содержит информацию о выбранном языке программирования и его версии (в данном случае – SQL, PostgreSQL 9.6). Это дает пользователю возможность выбрать предпочтительную версию/расширения языка для работы. В случае SQL пользователь может выбрать с какой СУБД работать. Данный выбор представлен на рисунке 6.

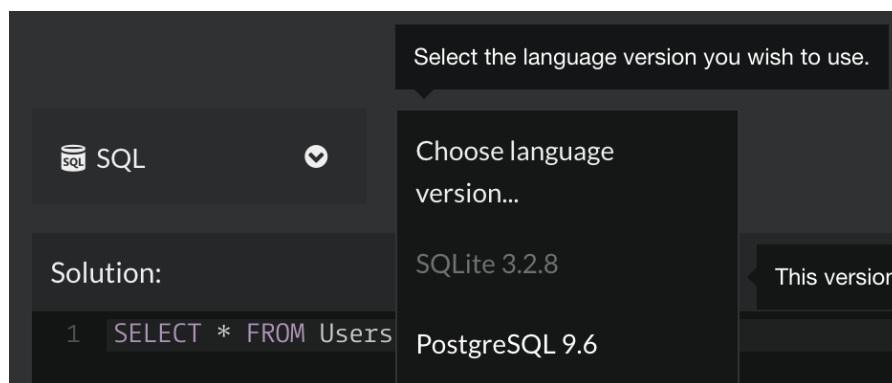


Рисунок 6 – Выбор версии языка программирования

Далее в данном блоке присутствует возможность выбора темы интерфейса (темная или светлая), выбора типа редактора кода (обычный, vim, emacs) и включение полноэкранного режима.

Основную часть данного блока занимает окно ввода кода. Данное окно представлено в виде простейшей IDE, которая подсвечивает служебные слова, нумерует строки, но не имеет более сложных функций типа помощь при вводе и подсветки неправильного синтаксиса.

Для собеседований необходимыми являются функции выбора версии языка (для каких-то заданий изменение версии языка может быть запрещено) и окно ввода кода с простейшими функциями IDE для удобства пользователей.

1.2.3 Блок результатов

Тестирование производится с помощью автотестов, примеры которых представлены пользователю на странице. В блок результатов выводится информация о результатах данных автотестов. На рисунке 7 представлены результаты выполнения запроса.

```
Time: 2642ms Passed: 0 Failed: 214 Errors: 214 Exit Code: 1

Test Results:

  Basic tests

    Testing for letters ["a", "b", "c"]

      There was an error with the SQL query:

      PG::UndefinedTable: ERROR:  relation "users" does
      not exist
      LINE 1: SELECT count(*) AS "count" FROM (SELECT *
      FROM Users) AS "t1...
      ^

      undefined method `[]' for nil:NilClass
      Simplified backtrace: block in run_test

    Testing for letters ["z"]

      There was an error with the SQL query:

      PG::UndefinedTable: ERROR:  relation "users" does
      not exist
      LINE 1: SELECT count(*) AS "count" FROM (SELECT *
      FROM Users) AS "t1...
      ^
```

Рисунок 7 – Результаты выполнения запроса

Из данного блока можно получить информацию о времени тестов, количество успешно пройденных, количество проваленных тестов, количество ошибок, результат. Далее представлены ошибки, которые возникали при выполнении скрипта. Данная информация позволяет пользователю сделать выводы о работе написанного скрипта и понять, какие изменения нужно вносить, если при выполнении возникли ошибки.

Для проведения собеседований кандидату имеет смысл показывать только информацию, если скрипт в принципе не выполняется (синтаксические ошибки). Если же скрипт выдает неверный результат и не проходит автотесты, то кандидату об этом знать не нужно. Такая практика является общепринятой и используется, к примеру, на официальных экзаменах Microsoft по MS SQL Server.

1.2.4 Блок дополнительных функций

Дополнительные функции представлены в виде панели под блоком ввода кода. Блок с данными функциями представлен на рисунке 8.

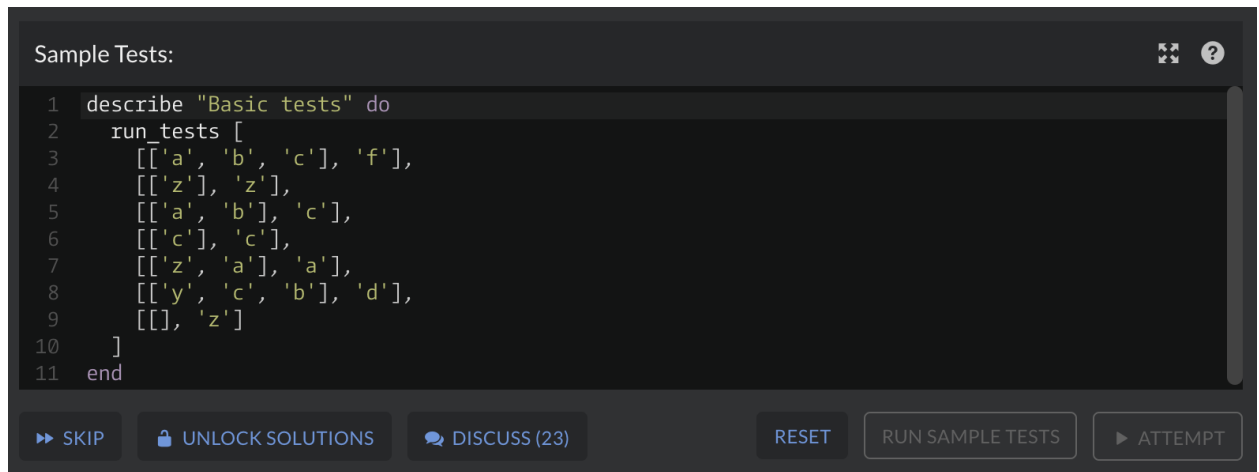


Рисунок 8 – Блок дополнительных функций

В данном блоке присутствует информация о простейших тестах, которыми можно проверить правильность своего решения и функциональные кнопки. С использованием функциональных кнопок можно:

- пропустить задание,
- показать решение,
- обсудить задание,
- перезапустить задание,
- провести простейшие тесты.

Для функционала собеседований необходимыми являются кнопки навигации по заданиям и запуска простейших тестов, то есть тестов синтаксиса и каких-либо других проверок на стороне сервера.

1.3 Анализ и тестирование функциональности для менеджера

Функциональность менеджера на данном портале представлена страницей создания задания. Данная страница представлена на рисунке 9 [4]. Выделим основные функциональные блоки на данной странице:

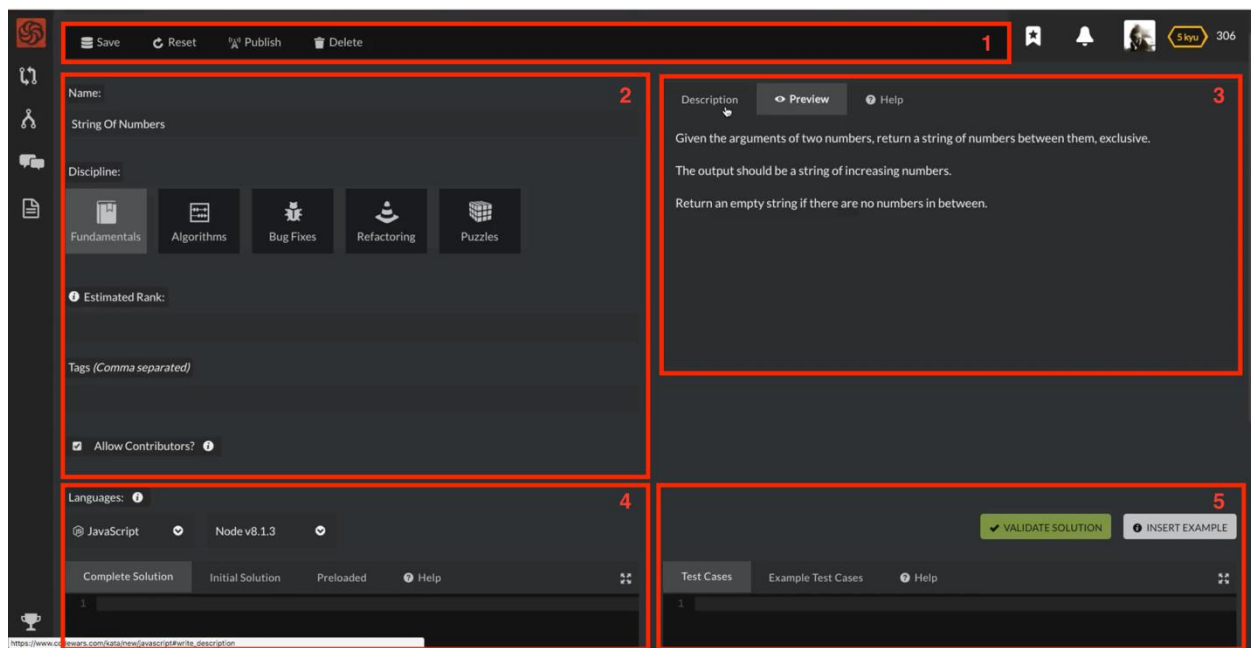


Рисунок 9 – Страница создания задания

- блок управления (1),
- блок информации о задании (2),
- блок задания (3),
- блок решения (4),
- блок тестов (5).

Анализ функциональности страницы создания задания будем проводиться по функциональным блокам, описанным выше.

1.3.1 Блок управления

Блок управления представлен на рисунке 10.

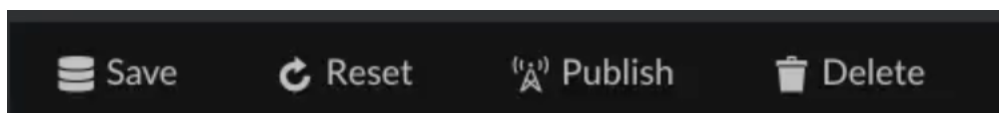


Рисунок 10 – Блок управления

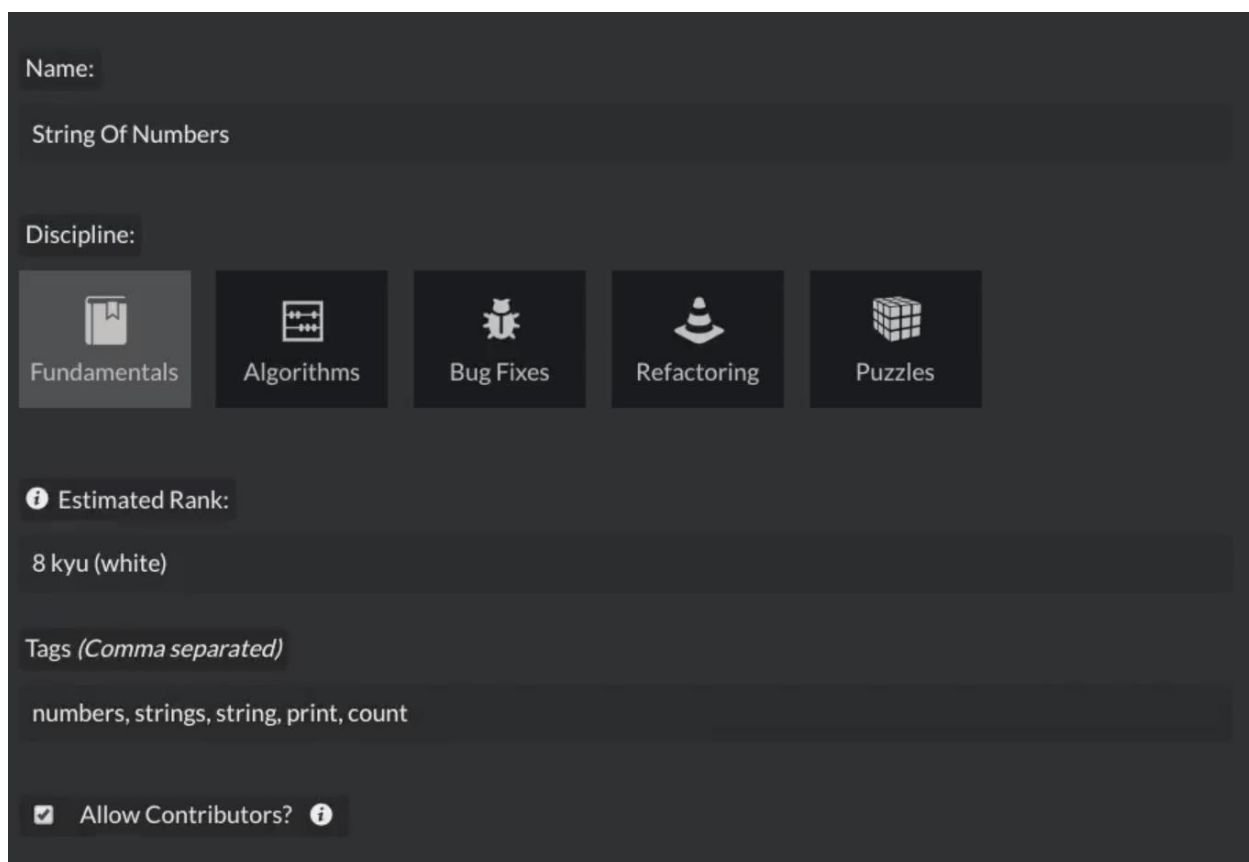
Данный блок содержит 4 функциональных кнопки. Кнопка «Сохранить» - позволяет сохранить задание без его публикации. Кнопка «Сброс» очищает все заполненные поля и можно начать создание задания заново. Кнопка

«Публикация» - осуществляет публикацию задания в открытый доступ. Кнопка «Удалить» - удаляет задание.

Для функционала собеседований необходимы аналогичные кнопки, так как данные 4 кнопки составляют необходимый функционал для управления заданием.

1.3.2 Блок информации о задании

Пример данного блока представлен на рисунке 11 [5].



The screenshot shows a form for creating or editing a task. It includes a 'Name' field with the value 'String Of Numbers'. Below it is a 'Discipline' section with five buttons: 'Fundamentals' (selected), 'Algorithms', 'Bug Fixes', 'Refactoring', and 'Puzzles'. Each button has an icon. The 'Estimated Rank' field shows '8 kyu (white)'. The 'Tags' field contains 'numbers, strings, string, print, count'. At the bottom, there is a checkbox 'Allow Contributors?' which is checked.

Рисунок 11 – Блок информации о задании

В данном блоке можно ввести название задания, которое будет отображаться пользователям. Далее представлена возможность выбора типа задания: основы, алгоритмы, исправление багов, рефакторинг кода, пазлы. Поле Rank позволяет установить минимальный ранг пользователей, которым доступно данное задание (ранг начисляется за верно решенные задания – так данная платформа ранжирует уровень знаний пользователей). Предпоследнее

поле отвечает за теги, которые описывались в пункте 2.1. Завершает данный блок флаг разрешения соавторства, которое позволяет разрешить или запретить другим пользователям предлагать изменения в задание.

С точки зрения менеджера ценность имеет только название задания и разрешение соавторства. Остальные же поля не имеют никакой ценности в условиях проведения технического собеседования.

1.3.3 Блок задания

В блоке задания имеются 3 вкладки: «описание», «предпоказ» и «помощь». Данные вкладки отвечают за редактирование задания, показ задания в виде, в котором оно будет демонстрироваться пользователю и инструкция к работе с этим полем соответственно.

Вкладка «описание» отвечает за полный контроль написания задания, то есть автор может писать в данной вкладке текст, как-либо форматировать его, чтобы получить задание в текстовом виде, которое будет представлено конечному пользователю. На рисунке 12 представлен простейший пример работы с данной вкладкой.

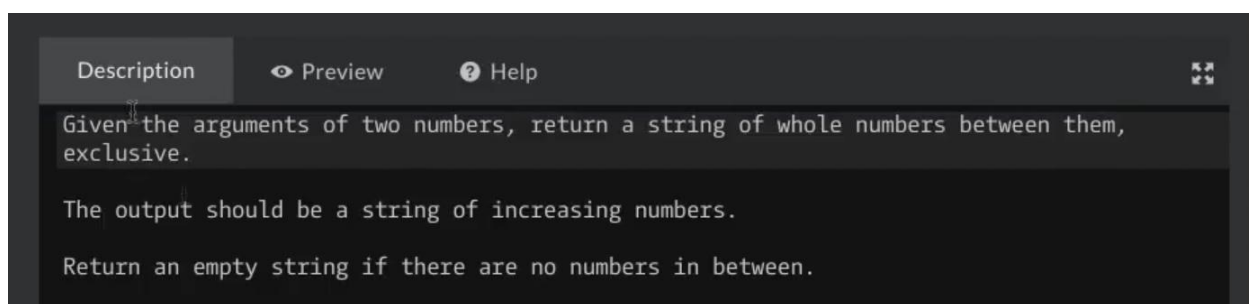


Рисунок 12 – Пример работы с вкладкой «описание»

На вкладке «предпоказ» (рисунок 13) можно увидеть, как будет выглядеть задание, написанное в предыдущей вкладке для конечного пользователя. На данной вкладке представлен текст задания со всеми модификациями (например, выделение текста жирным).

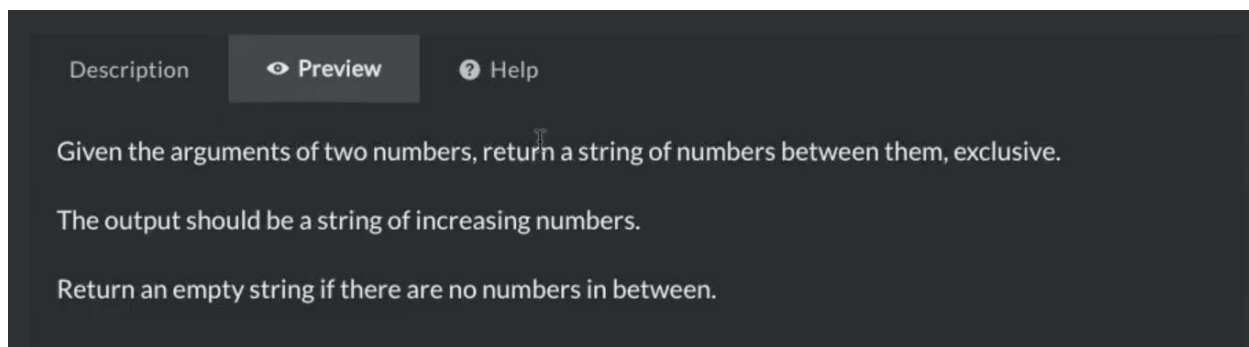


Рисунок 13 – Вкладка «предпоказ»

Вкладка «помощь» (рисунок 14) дает автору возможность получить инструкцию по заполнению поля «описание» и получить полную и исчерпывающую информацию о возможностях, которые могут быть использованы для формирования текста задания.

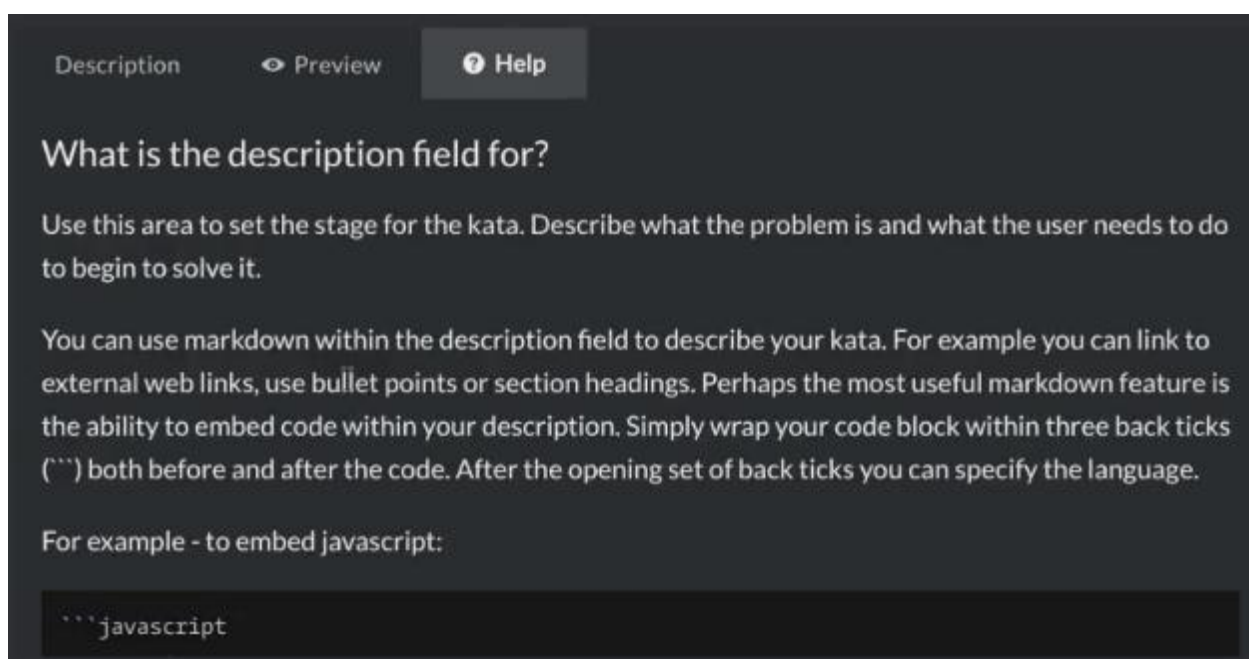


Рисунок 14 – Вкладка «помощь» в блоке задания

Для проведения собеседований все вкладки важны и будут использоваться. Кроме того, необходимо предоставить функцию для добавления картинок в задание. Данная функция имеет ценность именно для заданий по написанию запросов к базе данных, так как с помощью картинки проще всего представить схему объектов базы данных.

1.3.4 Блок решения

Блок решения представляет из себя выбор языка и версии языка программирования и форма с 4 вкладками для создания решения задания. Данный блок представлен на рисунке 15. Выбор языка и версии языка аналогичен по функциональности выбору для пользователя, представленному на рисунке 6 и был описан ранее. Форма ввода решения состоит из вкладок: «полное решение», «начальное решение», «предзагруженное решение» и «помощь». Данная форма ввода решения так же, как блок решения у пользователя имеет функции простейшей IDE [6].

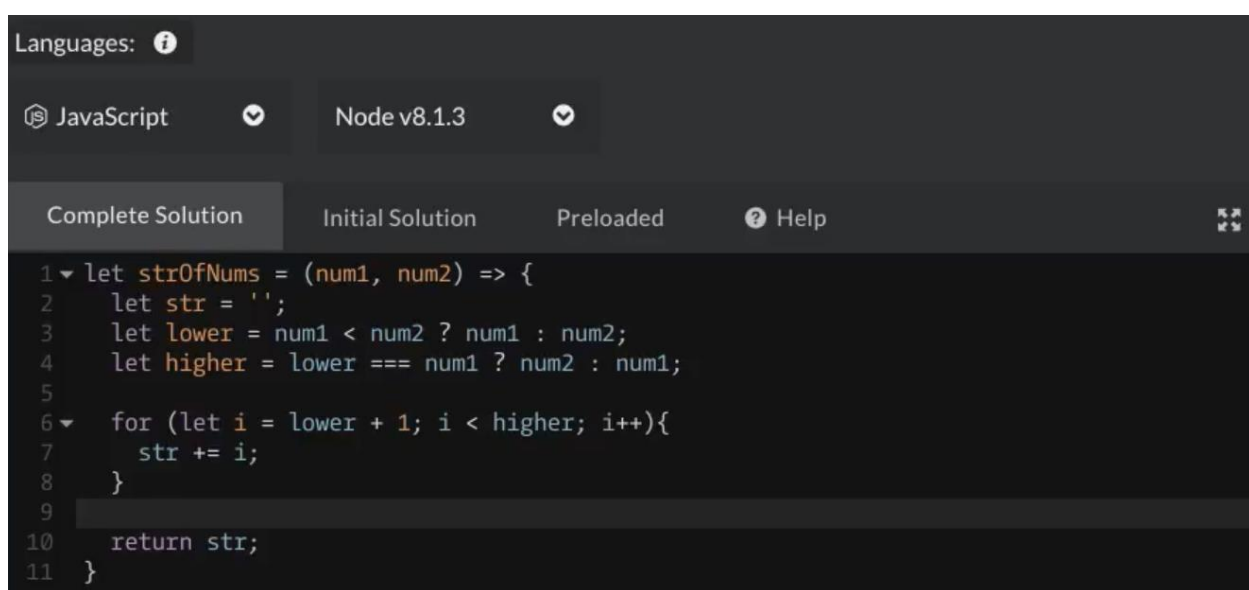


Рисунок 15 – Блок решения

Вкладка «полное решение» (рисунок 16) позволяет ввести эталонное решение задания, написанного прежде. Это решение будет показываться пользователям, если они решат пропустить задание, закончат его решение или нажмут на кнопку показа решения.

Рисунок 16 – Вкладка «полное решение»

На вкладке «начальное решение» (рисунок 17) автору теста (менеджеру) необходимо написать то, что будет показываться пользователю в блоке написания решения. Это может быть кусок кода, который будет служить подсказкой для решения задания, или комментариев, упрощающий решение задания. При желании и необходимости можно оставить данный блок пустым, тогда у пользователя будет просто пустая первая строка в блоке с решением.

Рисунок 17 – Вкладка «начальное решение»

Вкладка «предзагруженное решение» позволяет автору увидеть, как пользователь будет видеть его решение (аналогично с вкладкой «предпоказ» в блоке задания).

Вкладка «помощь» содержит инструкции по работе с данным блоком, для чего он нужен и информацию по правильному написанию эталонных скриптов. Данная вкладка представлена на рисунке 18.

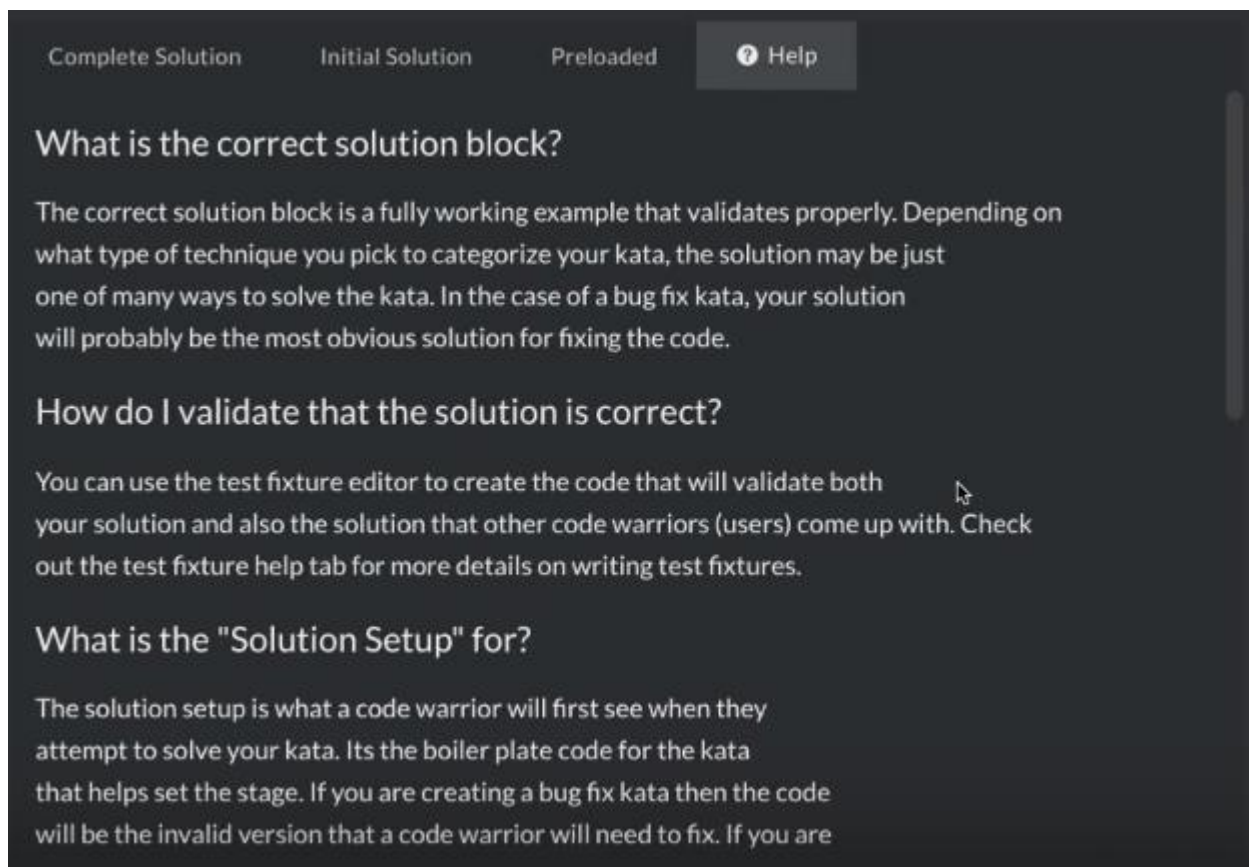


Рисунок 18 – Вкладка «помощь» в блоке решения

Для собеседований являются необходимыми вкладки «полное решение», «начальное решение» и «помощь». Также автору необходима возможность добавлять скрипты для создания объектов базы данных в файловом виде или повторно использовать уже созданные объекты.

1.3.5 Блок тестов

Блок тестов используется для написания функционального тестирования скрипта. В данном блоке автор пишет тесты, которые будут запускаться при нажатии пользователем кнопки для запуска его скрипта. Данный блок состоит из кнопок запуска тестов, помощи, которые запускают тесты и вставляют примеры тестов соответственно, и 3 вкладок: «тесты», «примеры тестов» и «помощь». Данный блок представлен на рисунке 19 [7].

Кнопка запуска тестов запускает все написанные автором тестом, которые указаны во вкладке «тесты». Кнопка помощи покажет примеры кода, которые

могут быть использованы для написания тестов. Функция особенно необходима для авторов, которые пишут тесты впервые.

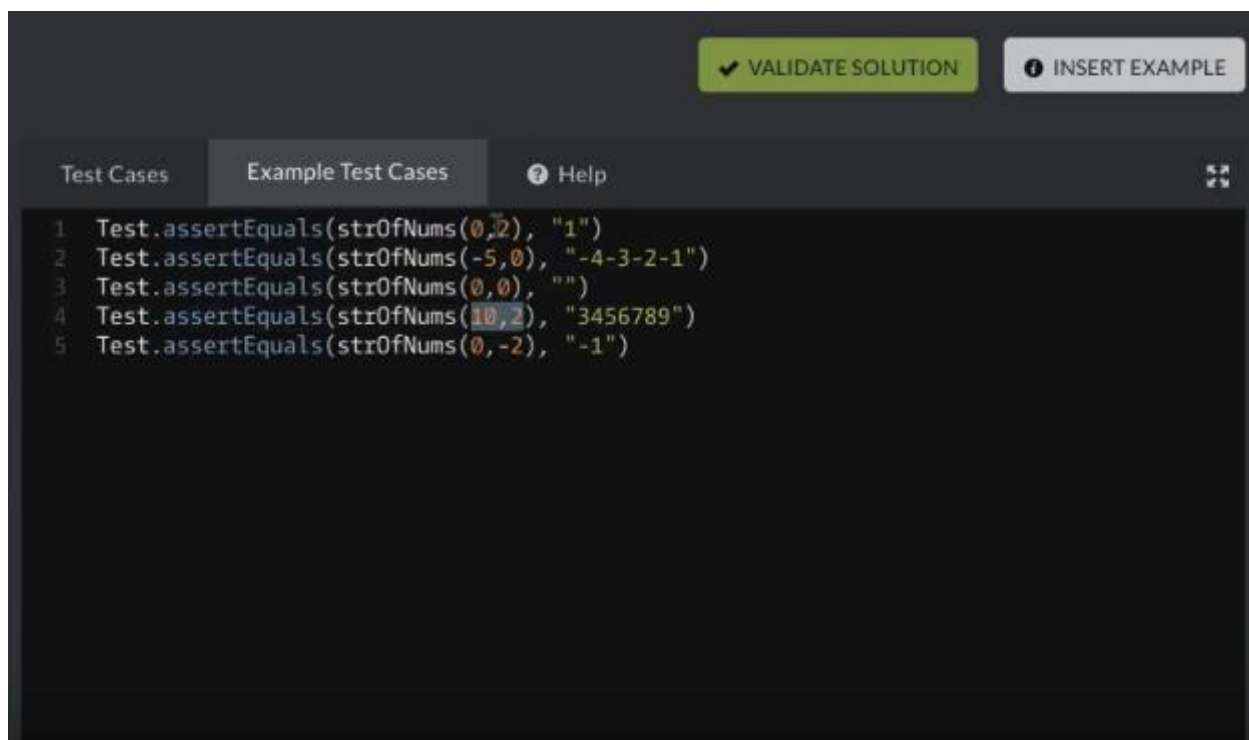


Рисунок 19 – Блок тестов

Вкладка «тесты» (рисунок 20) содержит информацию о тестах, которые должны быть проведены для получения исчерпывающей информации о правильности работы пользовательского скрипта. Скрипт, написанный автором, должен проходить все тесты без ошибок для того, чтобы опубликовать задание. Считается хорошим тоном, если количество тестов приближается к нескольким сотням, содержат «граничные условия» и обработку неверных условий.

В вкладке «примеры тестов» (рисунок 21) находится информация о тестах, которые будут показаны пользователю в виде примера того, как будет проверяться его решение. Данная вкладка аналогична вкладке «тесты», но содержит лишь малую часть запускаемых тестов.

Вкладка «помощь» содержит инструкцию о работе с остальными вкладками в данном блоке, которая объяснить пользователю цель данного блока, что и как нужно писать в данном блоке и как это должно работать.

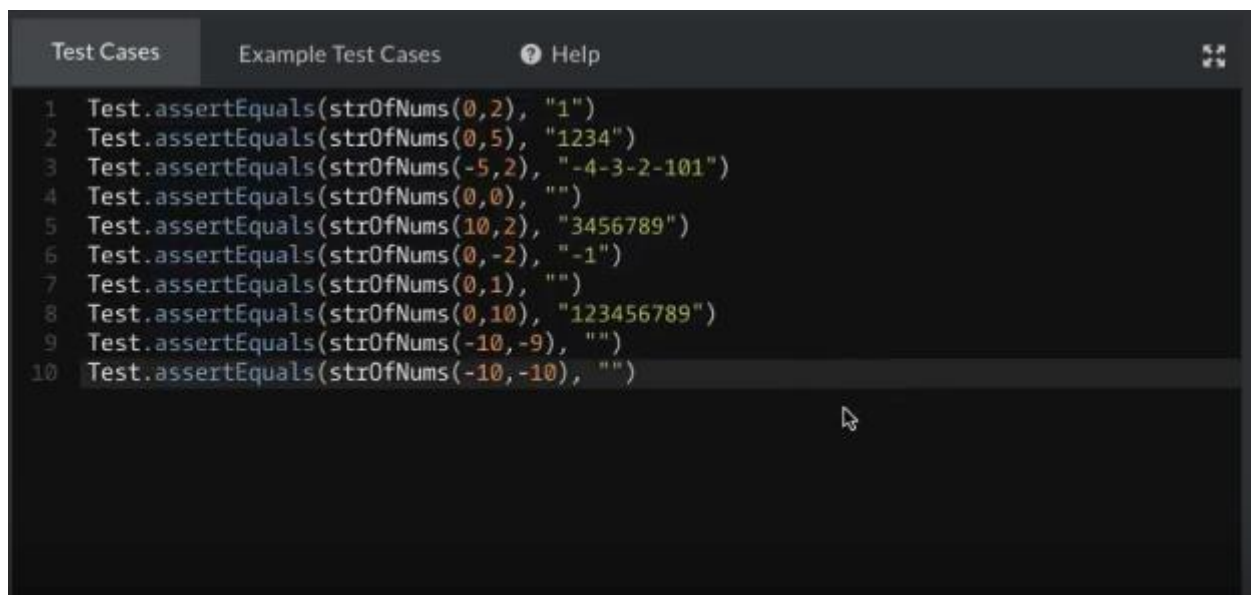


Рисунок 20 – Вкладка «тесты»

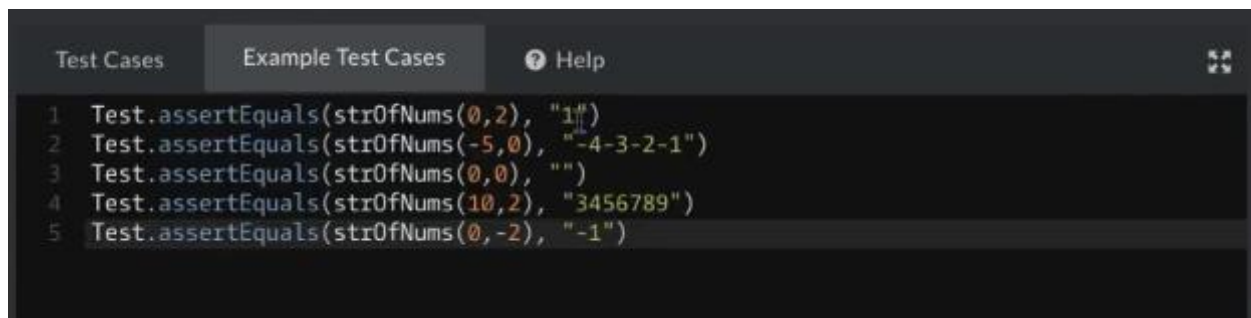


Рисунок 21 – Вкладка «пример тестов»

Для контроля знаний языка написания запросов данный блок не является необходимым в связи с тем, что результатом работы SQL запроса обычно является таблица, поэтому системе достаточно проверить соответствие таблицы, полученной пользователем к таблице, полученной в результате выполнения запроса автора.

1.4 Выбор необходимых функций СКЗБД

Анализ систем контроля знаний языка запросов к базам данных, выполненный в данной главе, заключался в подробном рассмотрении систем, представленных на российском и зарубежном рынке. Наиболее подходящей системой оказалась система «codewars», которая в соревновательном формате позволяет изучать различные языки программирования, в том числе и интересующий нас SQL с различными версиями.

Анализ системы был проведен по всем функциональным блокам, присутствующим на портале как с точки зрения пользователя, так и с точки зрения автора заданий. В результате работы можно выделить основные пункты, необходимые для успешного использования СКЗБД в ходе технических собеседований. Данные функции, совместно со степенью их важности (от 1 – блокирующий до 5 – незначительный) для проектируемой системы вынесены в таблицу 2.

Таблица 2 – Функции системы контроля знаний

Функция	Приоритет	CodeWars	Проектируемая система
Для кандидата:			
Имя автора задания	2	Да	Да
Задание в текстовом формате	1	Да	Да
Нумерация строк в блоке решения	3	Да	Да
Выделение ключевых слов в блоке решения	3	Да	Да
Возможность менять темы интерфейса (темная/светлая)	5	Да	Да

Продолжение таблицы 2

Проверка синтаксиса написанного запроса	2	Да	Да
Навигация по заданиям	1	Да	Да
Ограничение по времени на выполнение задания	1	Да	Да
Для менеджера:			
Блок управления заданием (сохранить, удалить, опубликовать)	1	Да	Да
Разрешение соавторства	2	Да	Да
Название задания	1	Да	Да
Теги задания	3	Да	Да
Задание в текстовой форме	1	Да	Да
Задание в тестовой форме	4	Нет	Да
Возможность загружать картинки в условие задания	1	Нет	Да
Возможность выделения особых блоков в тесте задания (фрагментов кода)	2	Да	Да
Возможность предποказа составленного задания	3	Да	Да
Инструкция по созданию задания	1	Да	Да

Продолжение таблицы 2

Блок для ввода эталонного решения	1	Да	Да
Возможность загружать скрипт создания объектов базы данных	1	Нет	Да
Возможность изменять созданные объекты из интерфейса системы	4	Нет	Да
Показ логина и пароля от пользователя БД для прямого подключения к ней	1	Нет	Да
Возможность повторного использования объектов БД	1	Нет	Да
Возможность ввода начального решения для пользователя	4	Да	Да
Вкладка помощи при написании эталонного запроса	1	Да	Да
Возможность формирования работы из нескольких заданий	2	Нет	Да
Возможность отправки на почту кандидату заданий	1	Нет	Да
Сбор статистики по выполненным заданиям	1	Да	Да
Отчет по выполненному заданию кандидата	1	Нет	Да

2 Проектирование и разработка системы

2.1 Разработка функциональной спецификации

Разработка функциональной спецификации состоит из анализа технического задания и определения основных функциональных характеристик будущей системы, целей работы и решаемых задач.

Начнем следует разбора целевых групп пользователей разрабатываемой системы. В соответствии с техническим заданием, в системе будет 2 типа пользователей: менеджер и кандидат. Менеджеру система позволит уменьшить трудозатраты на подготовку к техническим собеседованиям с кандидатами, получить возможность использования заданий заново и сбора статистики по решениям. Кроме того, система позволит уменьшить порог вхождения в должность для новых менеджеров. Кандидат получит возможность выполнения заданий технического собеседования в онлайн формате с использованием собственных источников информации и меньшим стрессом. На рисунке 22 представлены схемы действий менеджера и кандидата до использования системы и после.

В результате, основной ценностью разрабатываемой системы является уменьшение трудозатрат менеджера на технические собеседования и, как следствие, уменьшение денежных затрат предприятия на рутинные задачи менеджеров.

До использования СКЗБД



После начала использования СКЗБД



Рисунок 22 – Схема работы менеджера и кандидата до и после внедрения СКЗБД

2.2 Функциональная диаграмма IDEF0

Структурно-функциональная модель системы на основе методики IDEF0 позволит досконально разобраться в функциональных характеристиках разрабатываемой системы и рассмотреть все бизнес-процессы системы. Рассмотрим СКЗБД с нескольких точек зрения: с точки зрения ресурс-менеджера компании, который имеет потребность в оценке знаний кандидатов, и с точки зрения кандидата, который проходит тестирование в системе.

2.2.1 Функциональная модель с точки зрения ресурс-менеджера

В соответствии с требованиями, предъявляемыми к разрабатываемой системе, можно выделить несколько входных данных:

- информация о кандидате – информация о навыках кандидата полученная из резюме или общения с сотрудником отдела кадров.
- Пул существующих вопросов – список вопросов, которые были созданы прежде и могут быть использованы для комплекта заданий.
- Пул существующих объектов БД – информация об объектах, которые существуют в базе данных и могут быть использованы для новых заданий.
- Практический опыт – опыт менеджера, полученный в ходе повседневной работы.

В качестве управляющих данных можно выделить информацию о навыках, необходимых для вакансии, так как данная информация играет решающую роль в выборе кандидата.

Устройствами для выполнения задач (механизмами) являются две основные системы:

- Тестирующая система – разрабатываемая система.
- СУБД – системы управления базами данных, которые будут использоваться для проверки правильности выполнения задания. Определяются в соответствии с техническим заданием.

Выходными данными являются результаты тестирования, которые будут использованы для принятия решения о соответствии кандидата на должность.

На основе представленных данных может быть разработана начальная контекстная диаграмма (рисунок 23).



Рисунок 23 – Начальная контекстная диаграмма A-0

При детализации диаграммы A-0 можно выделить несколько основных функциональных блоков:

- 1) Просмотр существующих заданий – получение списка уже существующих заданий и информации о них.
- 2) Создание нового задания – создание объектов, написание задания и эталонного решения данного задания.
- 3) Создание комплекта заданий – сборка нескольких заданий в комплект для отправки кандидату.
- 4) Отправка комплекта заданий кандидату
- 5) Проведение тестирования
- 6) Получение результатов – проверка полученного решения кандидата и сравнение с эталонным решением.

Диаграмма A0 представлена на рисунке 24.

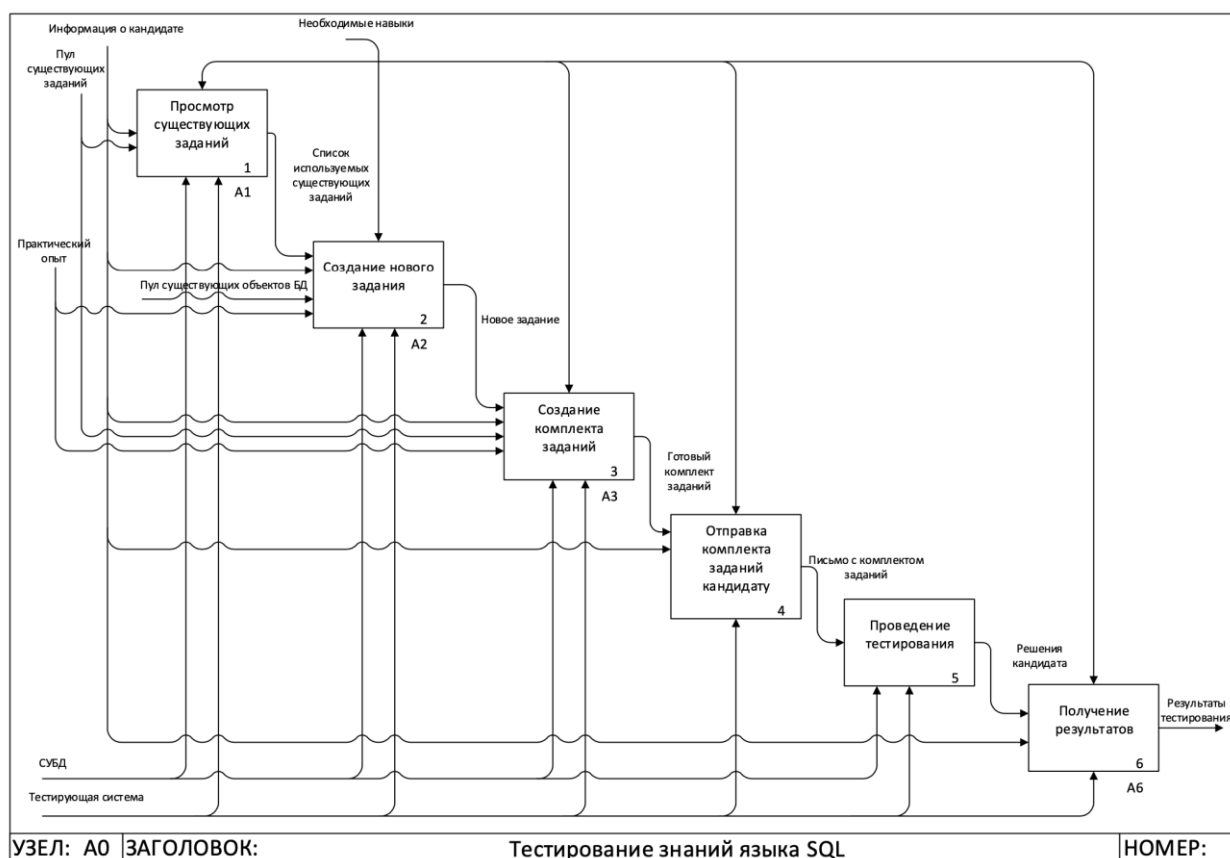


Рисунок 24 – Диаграмма A0. Тестирование знаний языка SQL

Функциональный блок «Просмотр существующих заданий» может быть детализирован в следующие функциональные блоки:

- 1) Просмотр ранее созданных заданий
- 2) Просмотр заданий других менеджеров с открытым доступом
- 3) Принятие решения об использовании уже созданных заданий

Данная детализация представлена на рисунке 25.

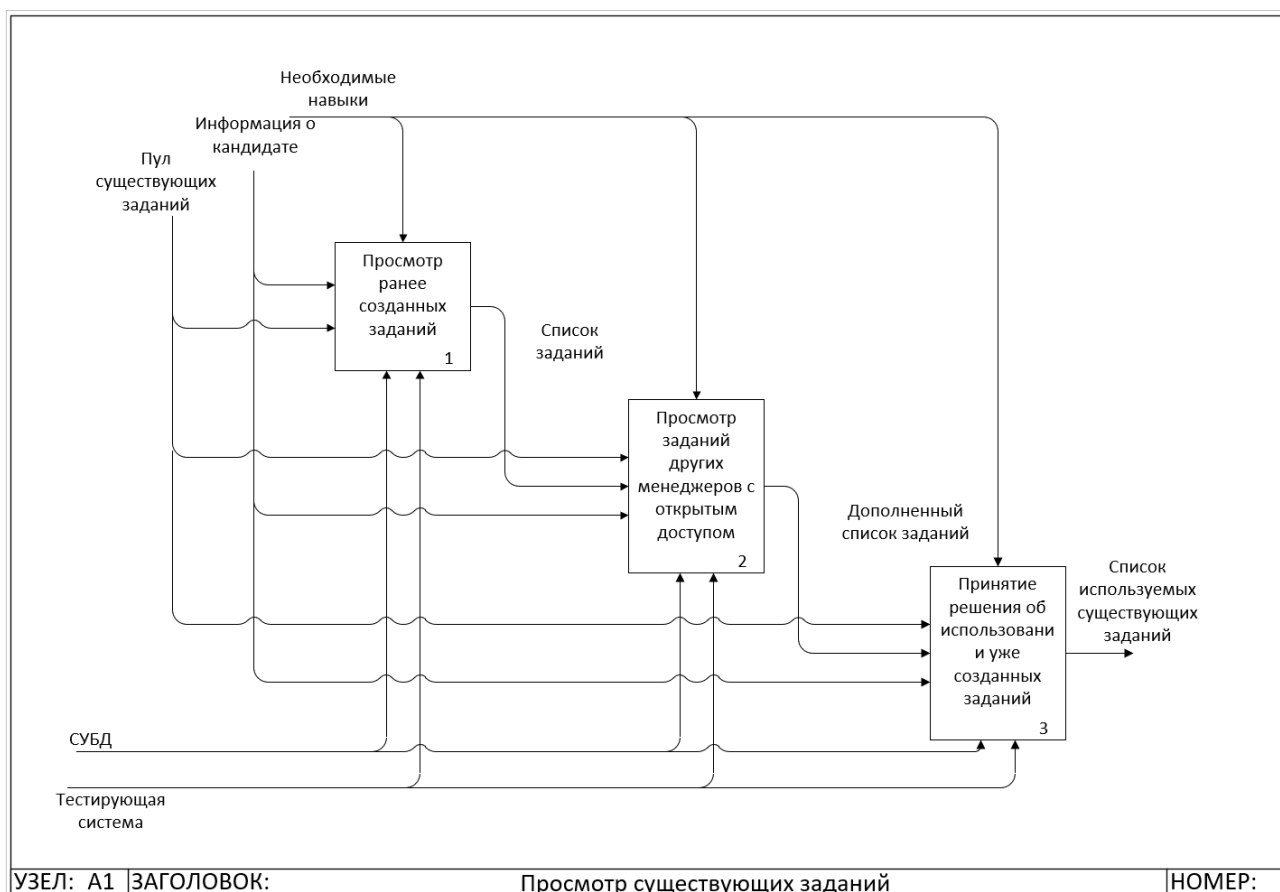
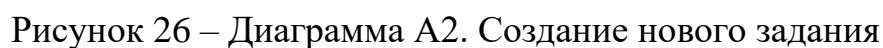


Рисунок 25 – Диаграмма A1. Просмотр существующих заданий

Самым сложным функциональным блоком системы является блок о создании новых заданий. Данный блок состоит из следующих шагов:

- 1) Выбор схемы БД или создание новой – может быть выбрана схема с объектами, которая уже использовалась для создания заданий или может быть создана совершенно новая схема без объектов. В результате данного шага будут получены логин с паролем для прямого доступа к базе данных для точечной модификации объектов и данных.
- 2) Создание объектов – добавление объектов в базу данных для последующего составления заданий на их основе. Может быть сделано посредством загрузки SQL файла в интерфейсе тестирующей системы или с помощью прямого подключения к базе данных через программное обеспечение соответствующих СУБД. Данный функциональный блок будет дополнительно детализирован далее.

- Диаграмма А2 представлена на рисунке 26.



Как было упомянуто ранее, блок «Создание объектов БД» требует дополнительной детализации. Данный функциональный блок состоит из:

- 1) Получение логина и пароля для подключения к БД.
- 2) Анализ существующих объектов.
- 3) Добавление объектов с помощью SQL файла – загрузка и выполнение SQL файла через разрабатываемую систему.
- 4) Точечное исправление объектов и данных в БД – исправление объектов БД с помощью прямого подключения к базе данных через программное обеспечение соответствующих СУБД.

Данная детализация представлена на рисунке 27.

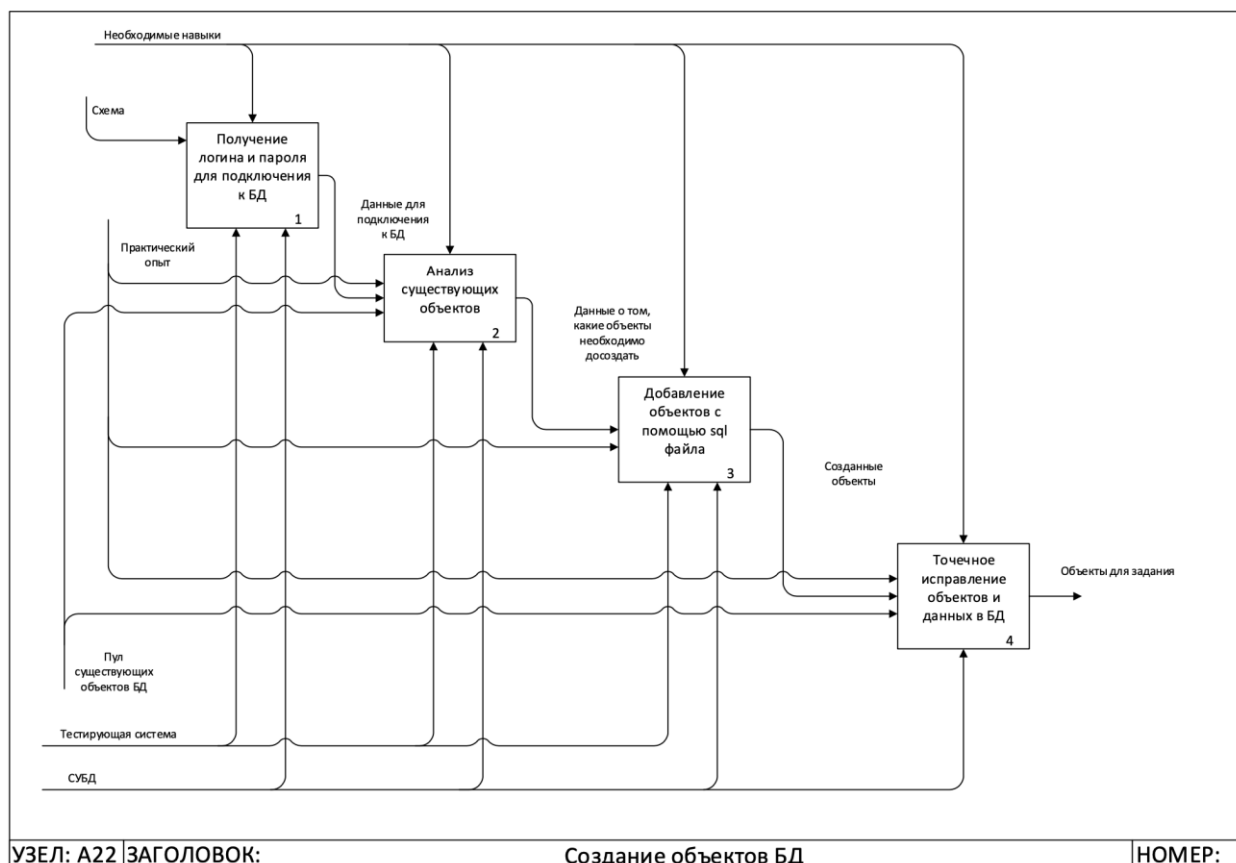


Рисунок 27 – Диаграмма A22. Сохранение объектов в БД

Блок «Создание комплекта заданий» (рисунок 24) может быть детализирован до следующих блоков:

- 1) Выбор заданий для комплекта.

- 2) Расположение их по порядку.
- 3) Сохранение комплекта.

Описанная детализация представлена на рисунке 28.

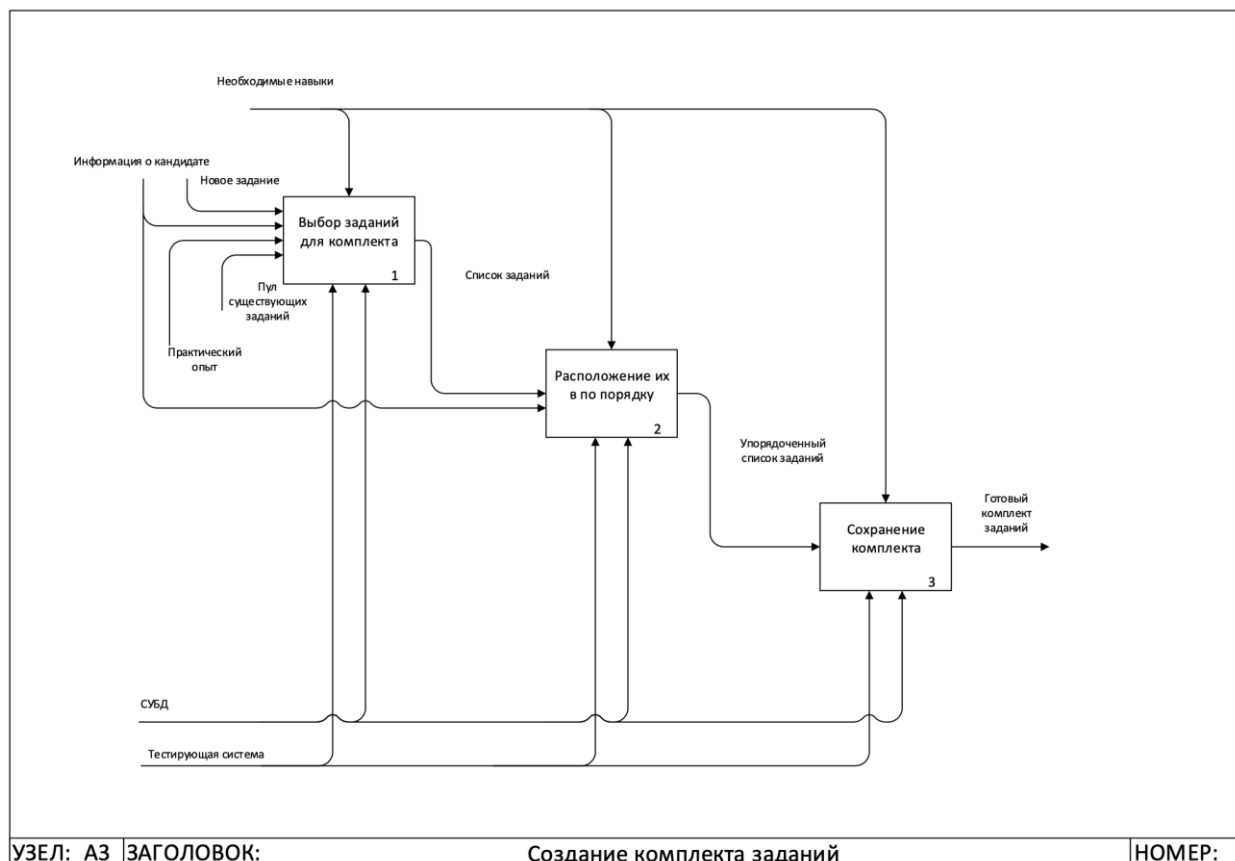


Рисунок 28 – Диаграмма А3. Создание комплекта заданий

Последним блоком, нуждающимся в детализации, является блок «Получение результатов» (рисунок 24). Блоки детализации представлены далее:

- 1) Сверка решений кандидата с эталонными запросами.
- 2) Сбор статистики.
- 3) Показ результатов кандидатов.

Детализация представлена на рисунке 29.

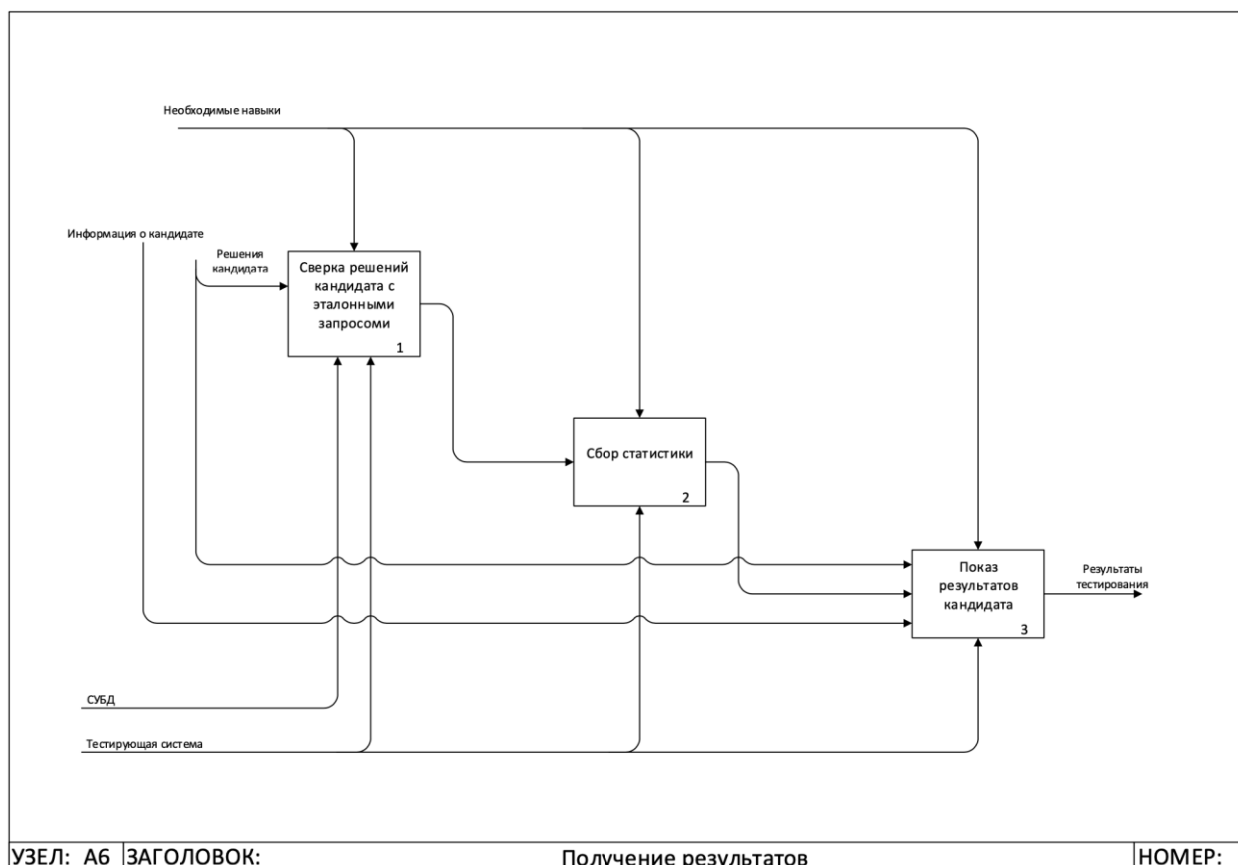


Рисунок 29 – Диаграмма А6. Получение результатов

2.2.2 Функциональная модель с точки зрения кандидата

С точки зрения кандидата разрабатываемая система намного проще, чем с точки зрения ресурс-менеджера, так как соискателю необходимо лишь пройти уже составленный тест, который был получен по электронной почте.

Кандидат имеет на вход письмо от работодателя с ссылкой на индивидуальный тест в тестирующей системе. Механизмом проведения тестирования является непосредственно разрабатываемая система. В качестве управляющих данных кандидат имеет собственные знания в рассматриваемой предметной области. Выходными данными считаются результаты тестирования, которые будут получены в устной или письменной форме от ресурс-менеджера на собеседовании или по электронной почте соответственно. Полученная начальная контекстная диаграмма представлена на рисунке 30.

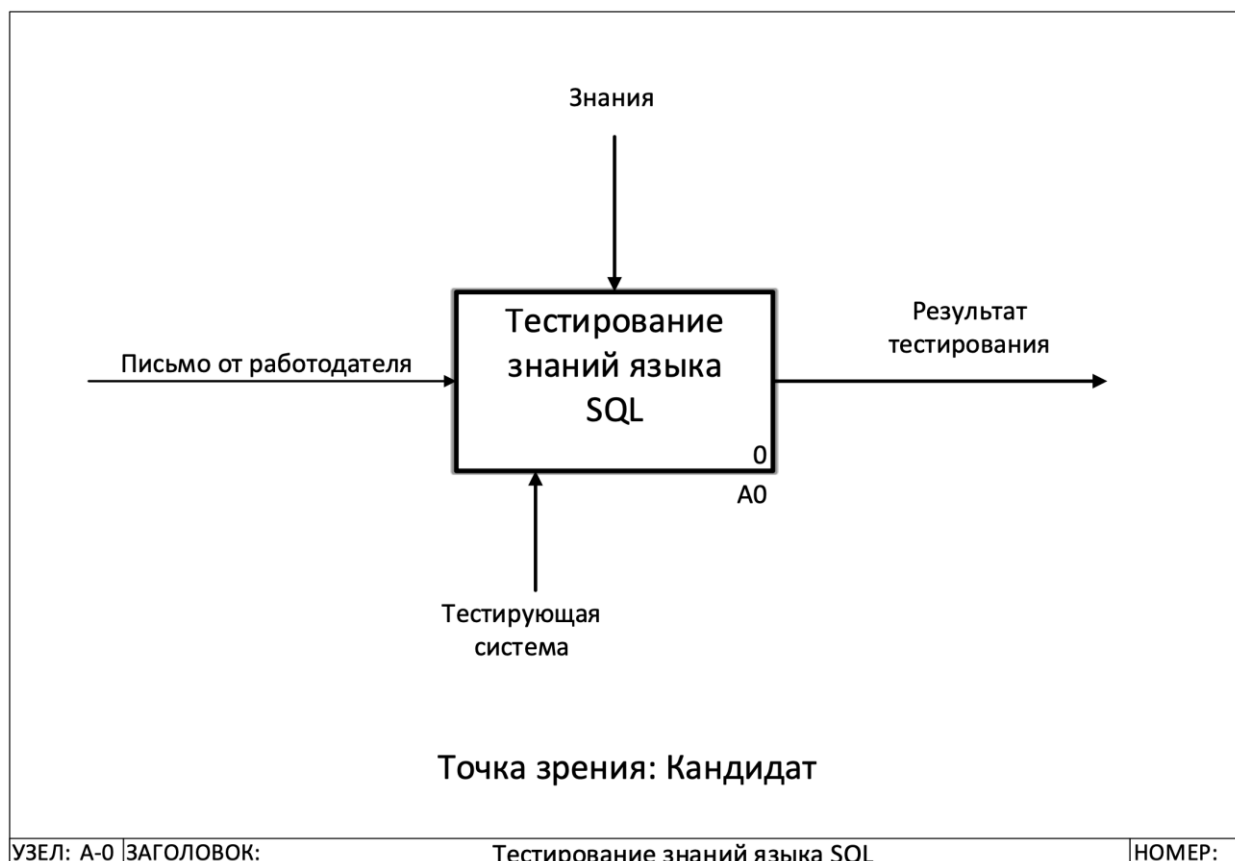


Рисунок 30 – Начальная контекстная диаграмма A-0 с точки зрения кандидата

Представленная начальная контекстная диаграмма может быть детализирована до следующих функциональных блоков:

- 1) Переход по ссылке из письма.
- 2) Прочтение инструкций к тестированию.
- 3) Прохождение тестирования.
- 4) Сохранение ответов.
- 5) Получение результатов после рассмотрения ответов менеджером.

Описанная детализация представлена на рисунке 31. Ни один из описанных выше функциональных блоков не может быть более детализирован.

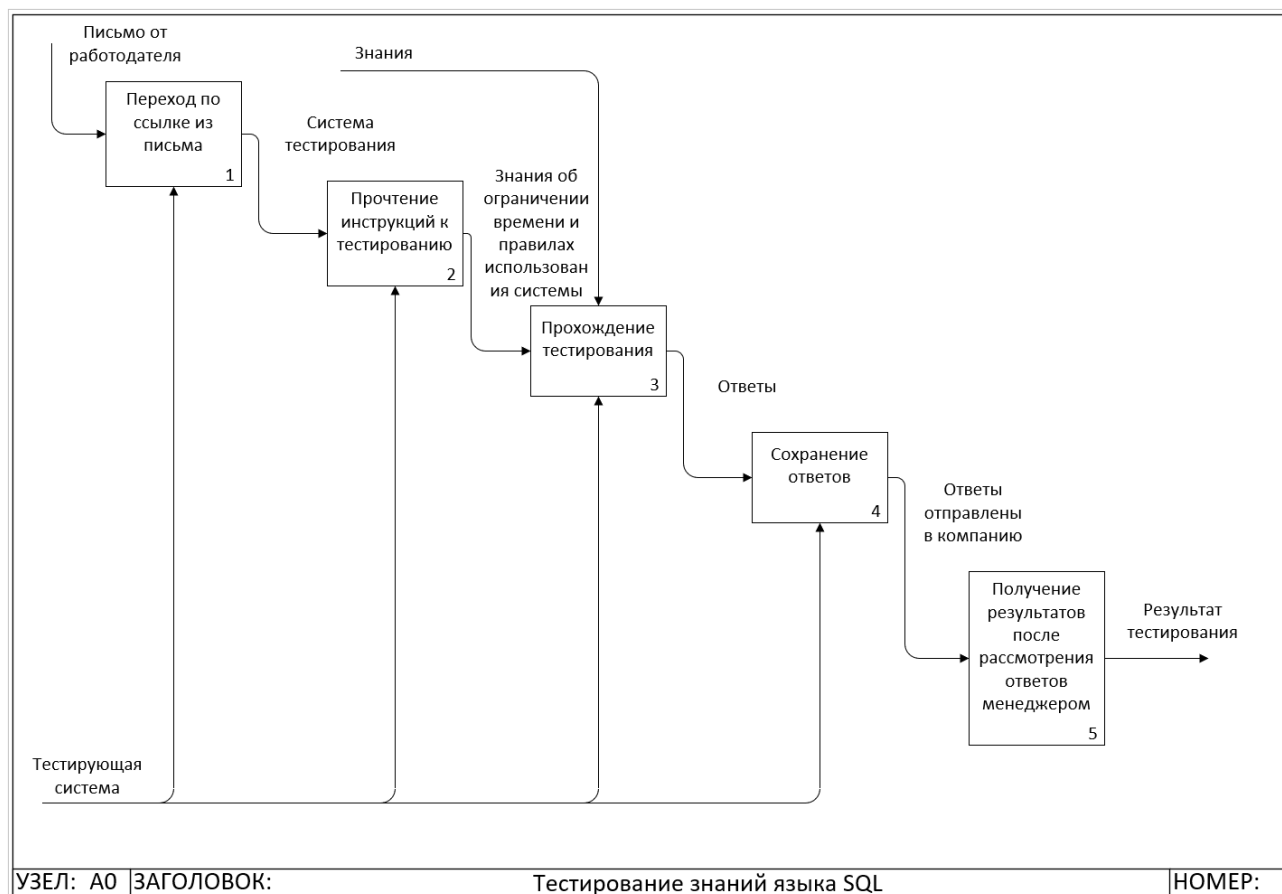


Рисунок 31 – Диаграмма A0. Тестирование знаний языка SQL с точки зрения кандидата

2.3 Проектирование структуры базы данных

СКЗБД должна содержать объекты, отвечающие различным бизнес-потребностям и используемые для различных целей: часть объектов необходимы системе для ее функционирования (системные объекты) и объекты для тестирования (тестовые объекты). Из соображений безопасности имеет смысл разместить различные типы объектов в различных базах данных, в ином случае существует вероятность использования SQL инъекций, которые могут принести непоправимый вред системе. Кроме того, тестовые объекты могут находиться в различных СУБД, однако системные объекты должны быть доступны разрабатываемой системе постоянно и находиться в одном месте. Размещение тестовых и системных объектов в различных базах способно решить и эту проблему.

Алгоритм работы с тестовыми объектами будет описан в последующих главах, в данной главе рассмотрим системную базу данных.

2.3.1 Концептуальное проектирование

Концептуальное проектирование состоит в построении информационной модели самого высокого уровня. Так как бизнес-процессы разрабатываемой системы были описаны с помощью нотации IDEF0, то могут быть получены основные сущности БД с помощью выделения объектов со своим уникальным набором свойств и бизнес смыслом. Кроме того, следует определить связи выделенных объектов между собой. Выделенные сущности представлены далее:

- Пользователи (Users) – в таблице будут храниться пользователи (ресурс-менеджеры) с логинами и паролями, информация о времени последней авторизации.
- Департамент (OrgUnit) – в таблице будут храниться департаменты компании, в которых может находиться пользователь.
- Схемы (Schemes) – названия схем тестовых объектов с указанием СУБД, создателя схемы и описанием, заполненным создателем схемы.
- Задания (Tasks) – текст задания, дополнительная информация о нем, эталонное решение, информация об используемой схеме.
- Метки (Tags) – таблица с метками которые могут быть использованы в заданиях.
- Метки задания (TaskTags) – связь заданий с метками.
- Комплекты заданий (Packs) – описание комплекта заданий, информация о времени создания комплекта и его создателе, количестве заданий.
- Таблица для связи комплектов и заданий (TasksOfPacks) – таблица будет использоваться для хранения информации о заданиях, которые находятся в комплекте заданий, их порядковом номере в комплекте.
- Отправки комплектов заданий (PacksSendings) – информация об отправках комплектов заданий на электронные почты кандидатов

- История изменений (ChangeHistory) – таблица для логирования любой информации об изменениях.
- Результаты решений (TestResults) – результаты тестирования кандидатов: запрос кандидата и результаты его проверки.

Все описанные сущности вместе со связями могут быть представлены в виде инфологической схемы, представленной на рисунке 32.

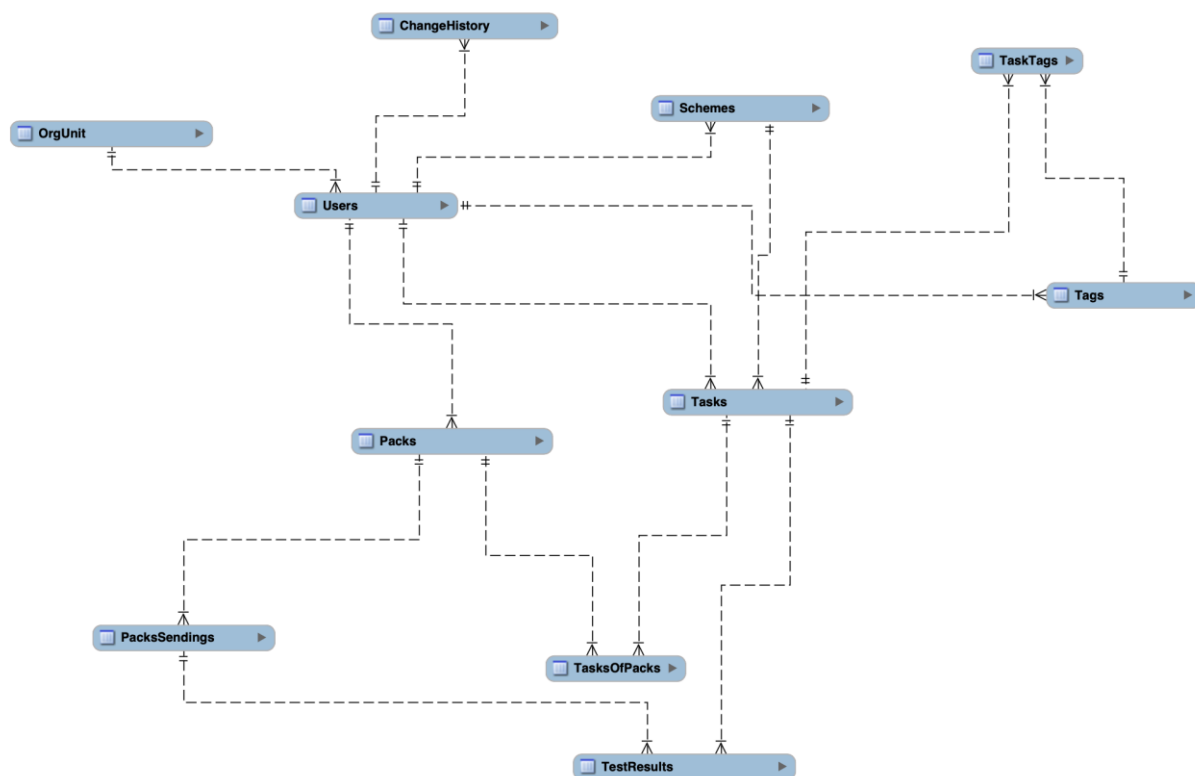


Рисунок 32 – Инфологическая модель системной базы данных

2.3.2 Логическое проектирование

Этап логического проектирования для данной базы данных состоит в подробном описании полей таблиц, ограничителей целостности. Даталогическая модель базы данных представлена на рисунке 33.

Большая часть полей соответствует техническому заданию и не нуждается в дополнительных комментариях. Далее будут описаны лишь поля, цель использования которых невозможно понять из ТЗ и бизнес-процессов системы:

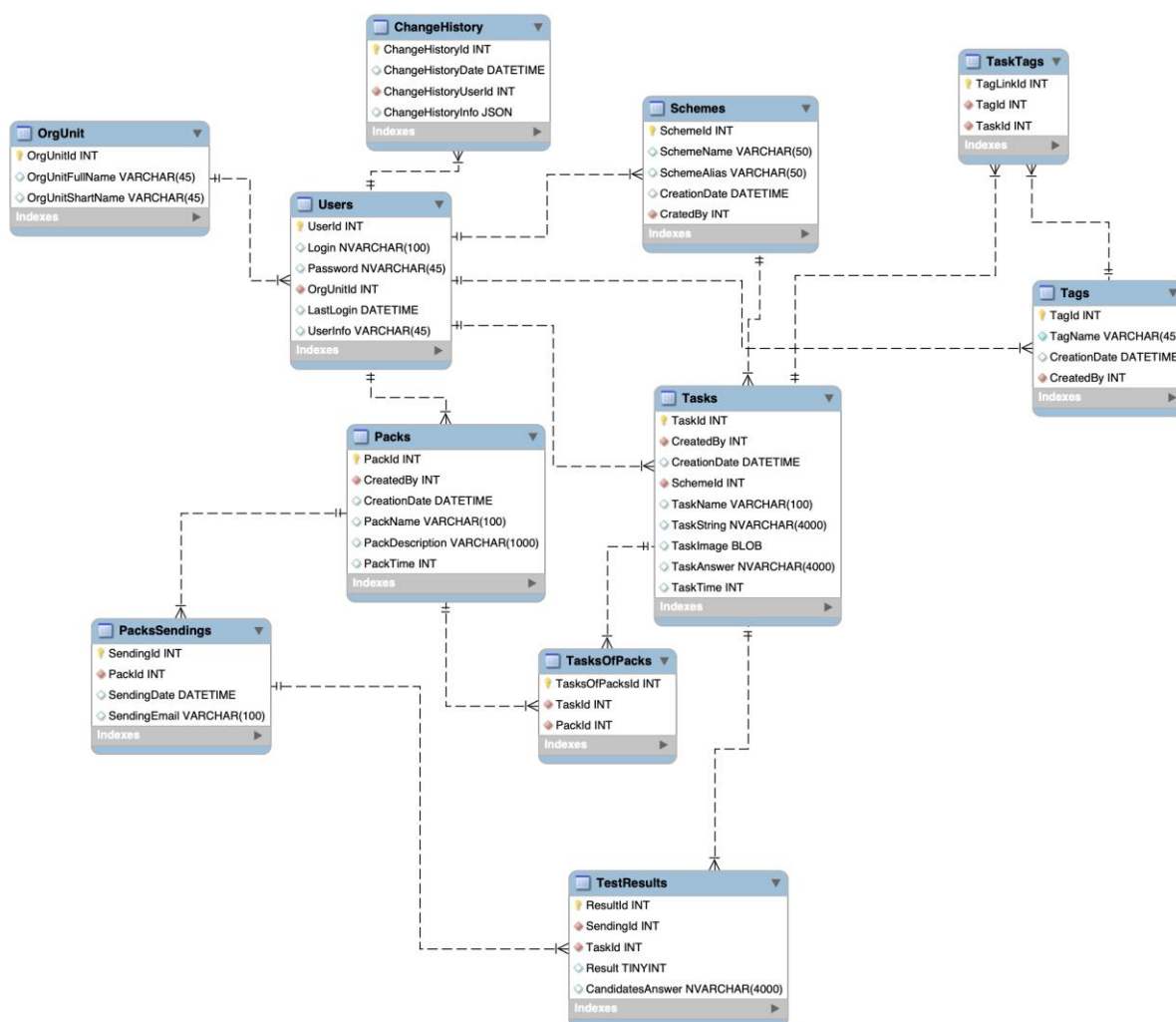


Рисунок 33 – Логическая схема системной базы данных

- CreationDate (DATETIME) – дата создания задания. Будет использоваться для указания даты и времени создания задания в системе, для понимания актуальности задания.
- CreatedBy (User) – пользователь, создавший задание. Может быть использовано для личного контакта при необходимости.
- ChangeHistoryInfo (JSON) – любые изменения в системе, которые должны логироваться. Хранится в формате JSON в связи с тем, что изменения могут быть какие угодно и их структура может быть независимой друг от друга.

- `SchemaName` (NVARCHAR(50)) – имя схемы базы данных для тестовых объектов для прямого обращения к данной схеме.
- `SchemaAlias` (NVARCHAR(50)) – пользовательское имя схемы базы данных, указывается при создании.
- `PackTime` (INT) – общее время на выполнение комплекта заданий. Определяется как сумма времени на выполнение всех заданий.

Основными ограничениями целостности являются первичные и внешние ключи, которые представлены на даталогической схеме (рисунок 11). В качестве первичных ключей в системе используются авто инкрементируемые суррогатные ключи, так как они позволяют со 100% вероятностью обеспечить целостность данных. Уникальные поля будут иметь ограничитель целостности UNIQUE, что позволит контролировать их уникальность без риска нарушения целостности данных.

Работа с базой данных из разрабатываемой системы будет осуществляться в пределах транзакций, что позволит исключить вероятность лишних блокировок и позволит поддерживать данные всегда в актуальном состоянии.

2.3.3 Физическое проектирование

Основная задача физического этапа проектирования в разрабатываемой базе данных – выбор СУБД.

На основе технического задания система должна работать с системами Oracle 11g или выше и MS SQL Server 2012 ли выше. Сравним основные преимущества обеих СУБД.

Преимущества MS SQL Server:

- Простота использования
- Утилиты (SQL Server Profiler, SQL Server Management Studio, BI tools и Database Tuning Advisor)
- Онлайн поддержка

- Простота и скорость восстановления (в сравнении с другими СУБД)
- Цена

Преимущества Oracle:

- Возможность обновлений без перезагрузок
- Кроссплатформенность
- Скорость работы с большими объемами данных (в сравнении с другими СУБД)
- Средства виртуализации

В целом, данные СУБД похожи друг на друга и имеют лишь небольшие различия, не имеющие значения для не высоконагруженных систем. В связи с этим выбор следует делать из соображений квалификации проектной команды. По данному критерию выбор будет сделан в пользу MS SQL Server.

Так как система не является высоконагруженной, то использование дополнительных индексов и каких-либо группировок данных не требуется. Снятие резервных будет производиться средствами системы управления базами данных и операционной системы в автоматическом режиме с помощью отложенного регулярного задания ОС.

2.4 Выбор языка и средств разработки

За многолетний опыт разработки программного обеспечения людьми были разработано достаточно большое количество различных подходов к разработке, каждый из которых имеет свои преимущества. В настоящее время наибольшую популярность приобрели гибкие методологии. Данные методологии позволяют создавать продукт более эффективно, нежели традиционные. Это происходит за счет того, что наибольшую ценность составляет бизнес ценность для заказчика программного продукта, а не сам программный продукт. По данной причине для разработки данной системы будет выбран именно Agile модель и методология Scrum, суть которой заключается в работе специальными временными

промежутками (1-2 недели) – спринтами, в результате которых производится показ результатов работы заказчику.

Выбор языка разработки серверной части программного обеспечения осуществляется по нескольким критериям:

- наличие библиотек для интеграции с различными СУБД,
- обширное сообщество,
- возможности масштабирования,
- возможности многопоточности.

Выбор осуществлялся среди 3 самых популярных языков для веб разработки: Java, .NET, Python. Данные языки имеют примерно одинаковые характеристики по выделенным критериям, в связи с чем было принято решение использовать Java, так как разработчик знает этот язык лучше, чем другие из представленных.

Язык для фронтенд разработки будет выбран по популярности языка среди сообщества, так как разработчик системы не имеет обширных знаний ни в одном из языков фронтенд разработки, а самый популярный язык имеет наибольшее количество обучающих видео и форумов. По данному критерию был выбран язык JavaScript, библиотека React.js.

2.5 Разработка алгоритмов работы системы

Большую часть алгоритмов работы системы являются тривиальными и вполне очевидны из функциональных диаграмм IDEF0. Однако, особого внимания требуют алгоритмы, отвечающие за самый главный модуль системы – за задания. В связи с этим, рассмотрим более детально следующие алгоритмы:

- алгоритм сохранения объектов базы данных;
- алгоритм сохранения задания.

Данные алгоритмы будут рассмотрены далее.

2.5.1 Алгоритм создания объектов базы данных

Алгоритм сохранения объектов базы данных состоит из нескольких основных функциональных блоков: выбора схемы и создания объектов.

Создание пользователя происходит с помощью команды, записанной в отдельном файле. Это происходит в несколько этапов: создание пользователя вместе со схемой данных, ограничение прав пользователя.

Алгоритм представлен на рисунках 34-36.

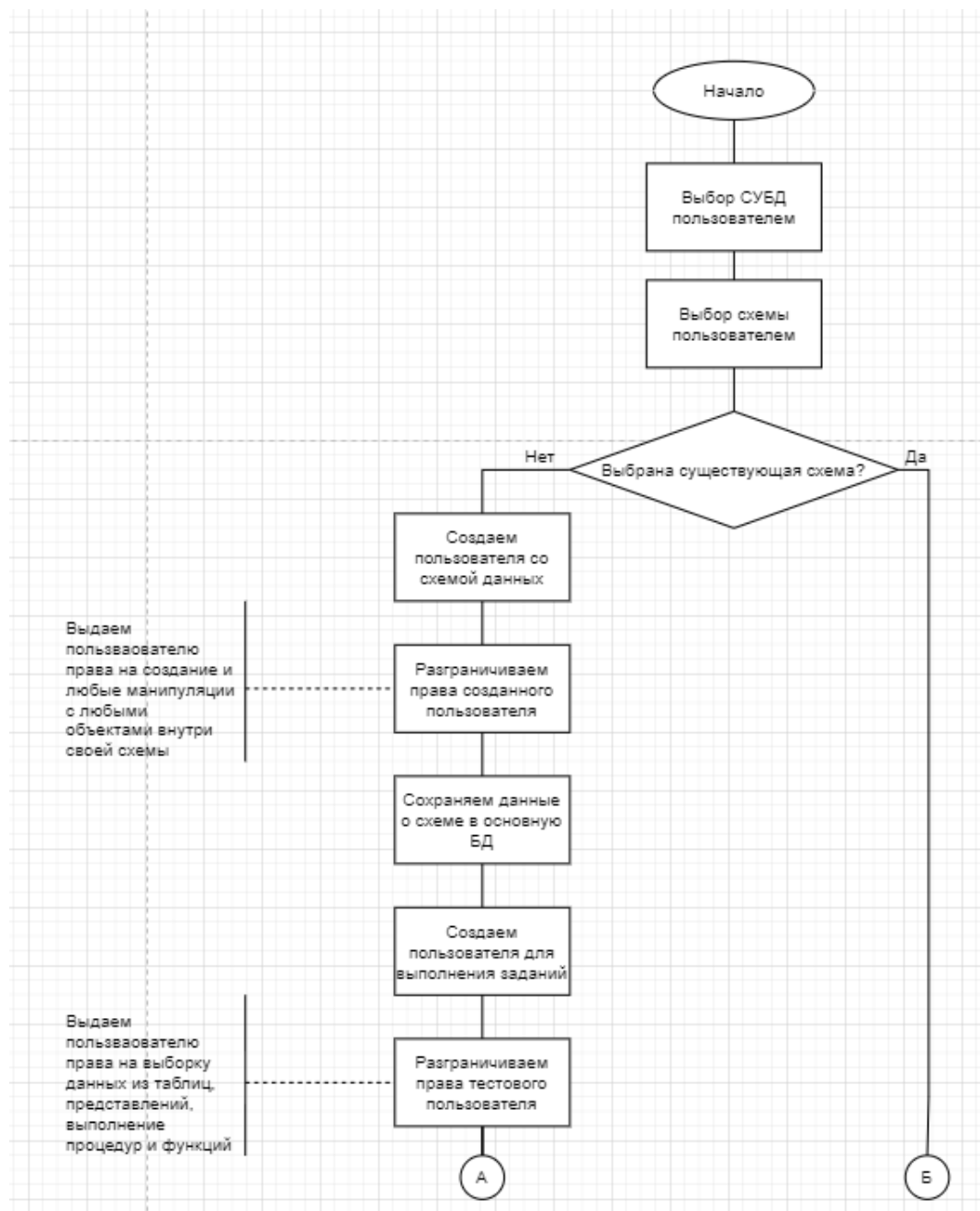


Рисунок 34 – Алгоритм создания объектов базы данных

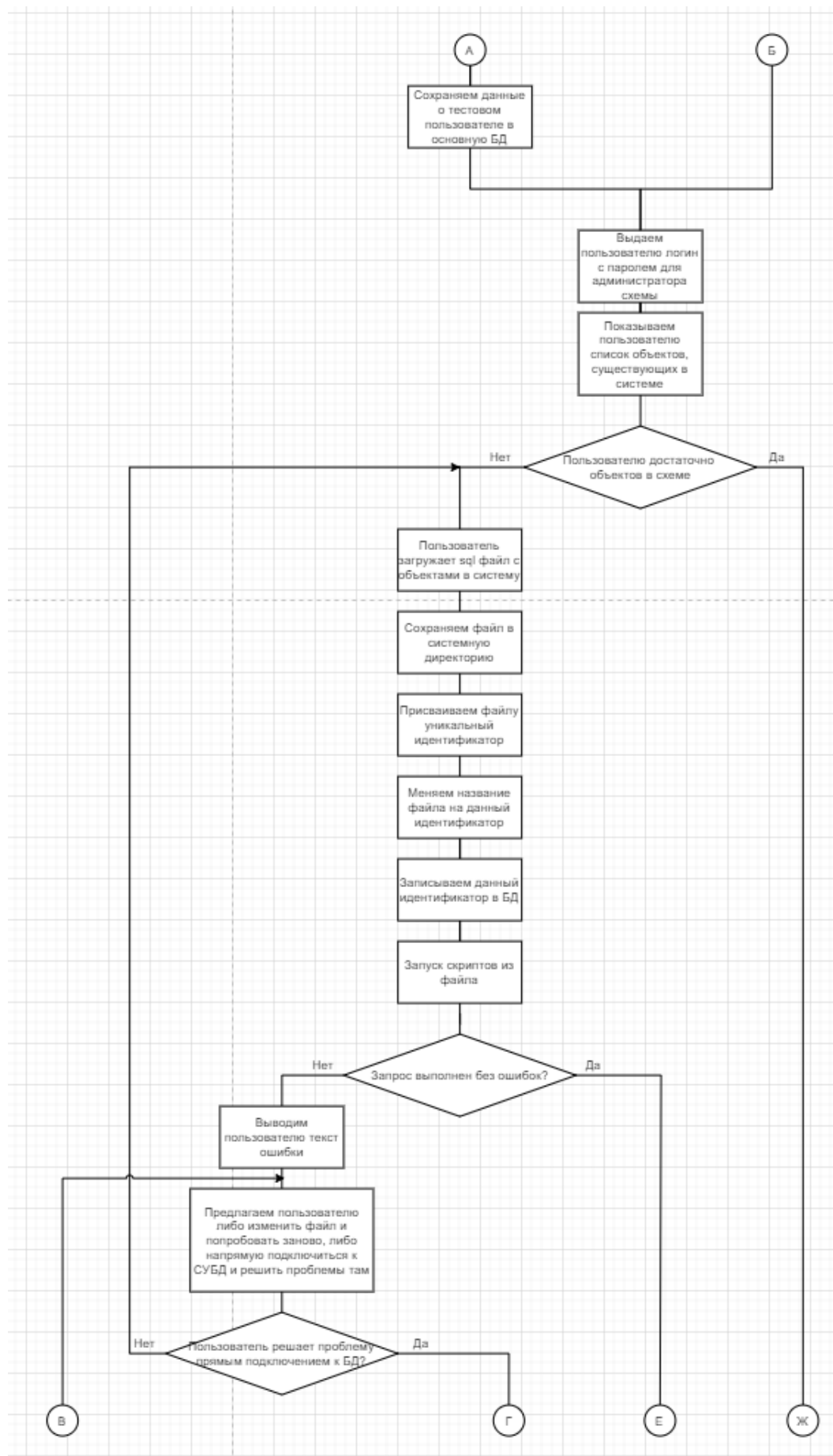


Рисунок 35 – Алгоритм создания объектов базы данных (продолжение)

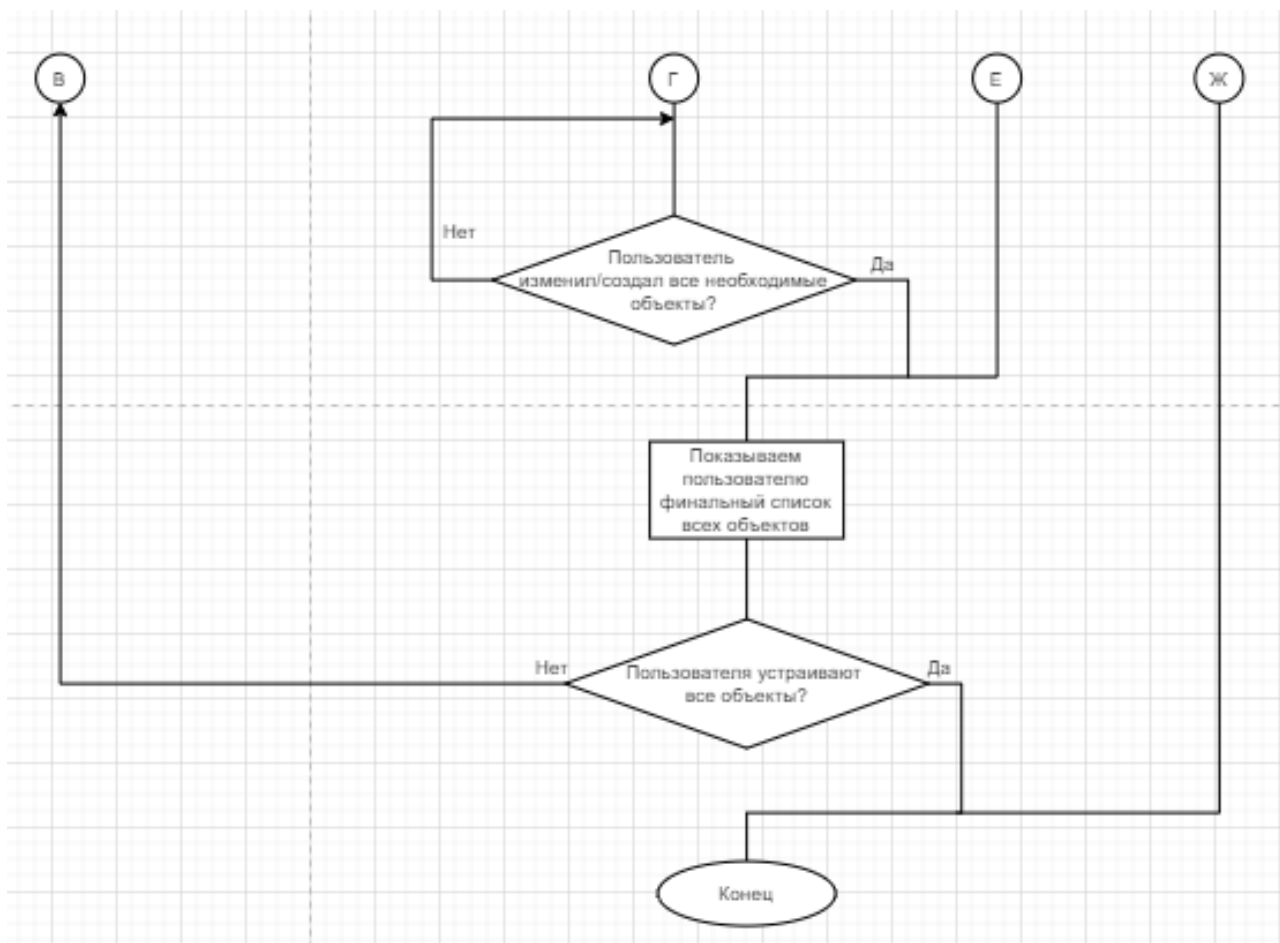


Рисунок 36 – Алгоритм создания объектов базы данных (продолжение)

2.5.2 Алгоритм сохранения задания

Алгоритм сохранения задания является одним из завершающих при формировании задания. Сначала производится добавление файла (изображения) в систему в качестве вложения. Затем происходит проверка эталонного решения, написанного менеджером. Только после этого происходит полноценное сохранение всего задания целиком.

Алгоритм представлен на рисунках 37-38.

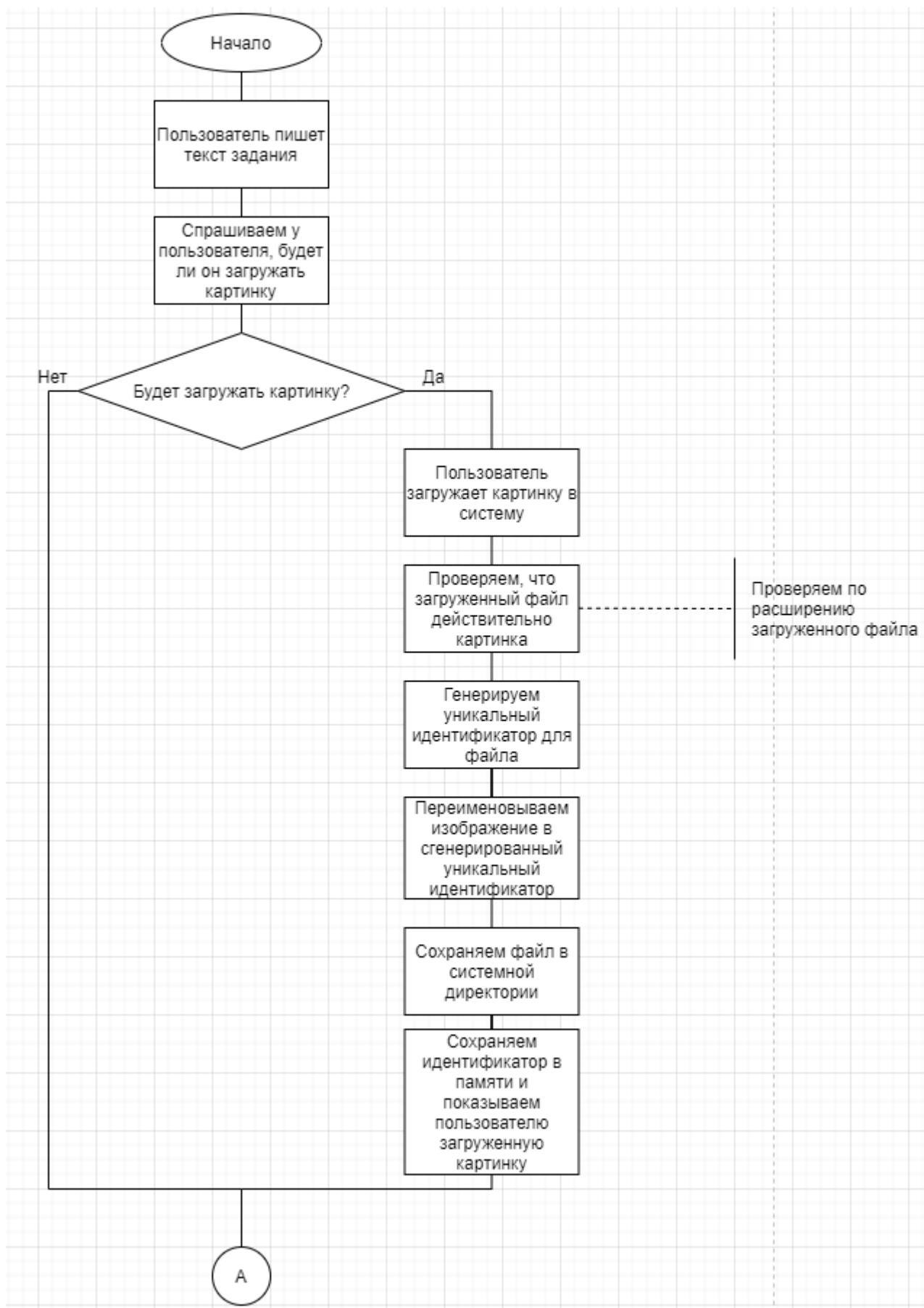


Рисунок 37 – Алгоритм сохранения задания

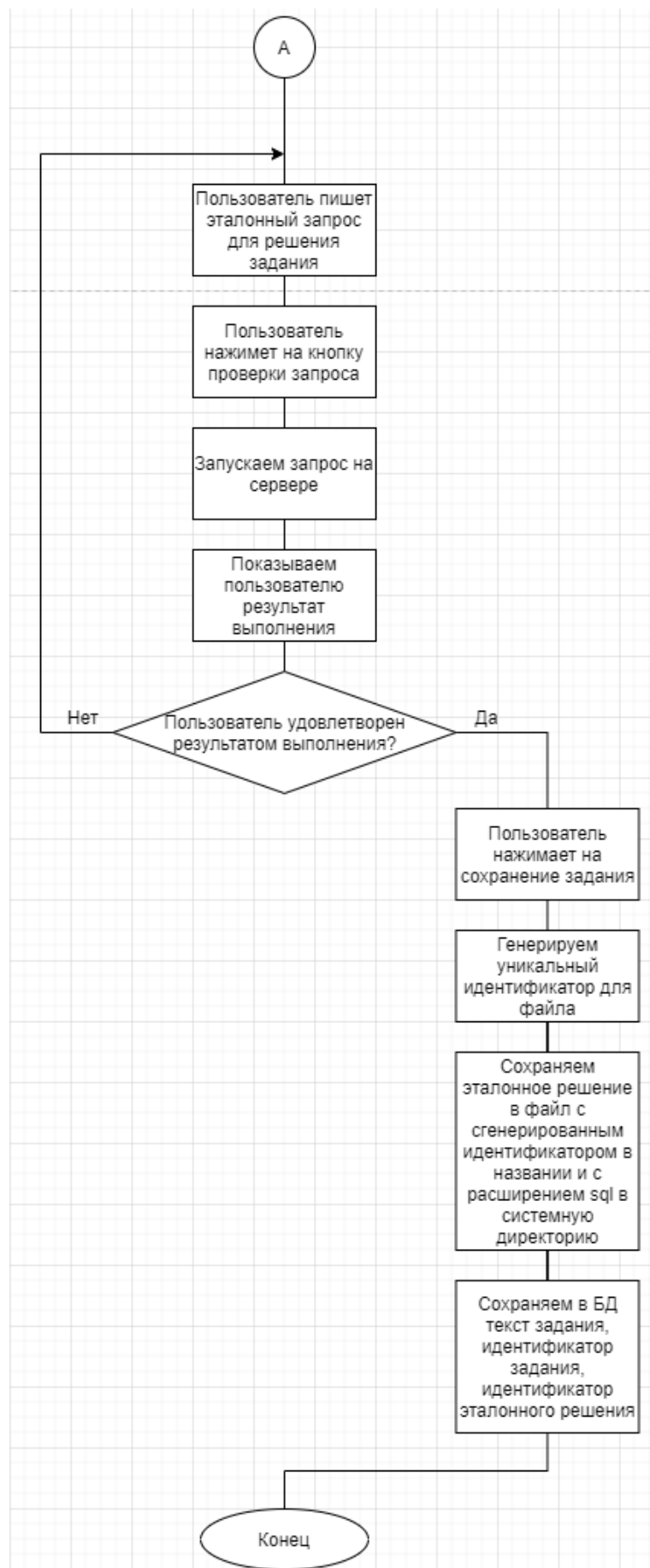


Рисунок 38 – Алгоритм сохранения задания (продолжение)

2.6 Разработка форм интерфейса

Разработка интерфейса программных систем ведется в несколько основных этапов:

- исследование;
- пользовательские сценарии;
- структура интерфейса;
- прототипирование интерфейса;
- определение стилистики;
- дизайн концепция;
- оформление всех экранов;
- анимация интерфейса;
- подготовка материалов для разработчиков.

Графически все эти этапы выглядят так [8]:



Рисунок 39 – этапы разработки интерфейсов

Этапы исследования и разработки пользовательских сценариев были пройдены при разработке функциональной спецификации IDEF0. В итоге, после 2 этапов имеем пользовательский опыт и пожелания будущих пользователей системы к тому, как система должна себя вести.

Структура интерфейса будет тривиальной, так как система не выполняет сложных функций. На экране выделим два блока: навигационный и основной. Навигационный блок будет представлять из себя панель в верхней части страницы, в которой будут ссылки для перехода по основным разделам системы и другая критически важная для пользователя информация (например, таймер

для кандидата). Основной блок будет содержать поля ввода, кнопки и другую информацию, для которой предназначена описываемая страница.

В связи с ограниченностью ресурсов и отсутствием группы людей для работы с прототипами и анализа их успешности, этап прототипирования может быть пропущен.

На этапе определения стилистики готовятся несколько наборов изображений (moodboards). Эти наборы могут быть представлены страницами сайтов, иллюстрациями, кнопками, шрифтовыми композициями, связанными между собой стилистически. В нашем случае данный этап можно объединить с этапом дизайн концепции, так как содержание интерфейса определено техническим заданием и диаграммой IDEF0. На рисунках 40-41 представлены несколько наборов изображений, которые позволят получить полное представление о визуальной стилистике системы.

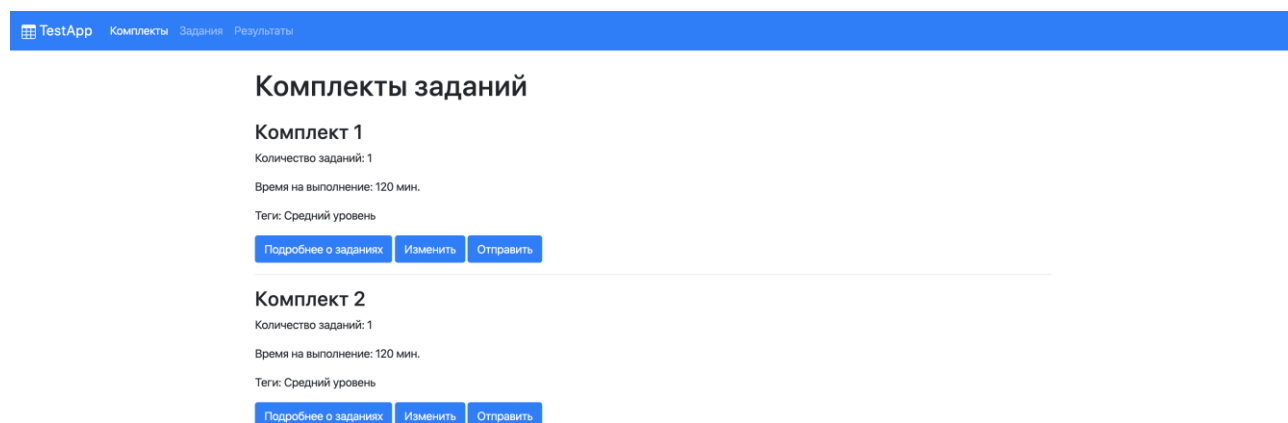


Рисунок 40 – Пример страницы с комплектами заданий

Задание 2 из 4

25:45

1 Покажите иерархию узлов и компонентов продукции, необходимой для создания велосипеда для ProductAssemblyID = 800.

2

3 Выведите следующие поля на экран:

4 - AssemblyID

5 - ComponentID

6 - Name

7 - PerAssemblyQty

8 - EndDate

9 - ComponentLevel

10

11 Вы можете использовать таблицы BillOfMaterials и Product, которые представлены на изображении

12

13 Подсказка:

14 Вы можете использовать общее табличное выражение

15

1 WITH PartsC

2 AssemblyID,

3 ComponentID,

4 PerAssemblyQty,

5 EndDate,

6 ComponentLevel

7) AS

8 (

9 SELECT b.ProductAssemblyID,

10 b.ComponentID,

11 b.PerAssemblyQty,

12 b.EndDate, 0 AS ComponentLevel

13 FROM Production.BillOfMaterials AS b

14 WHERE b.ProductAssemblyID = 800

15 AND b.EndDate IS NULL

16

17 UNION ALL

18

19 SELECT bom.ProductAssemblyID,

20 bom.ComponentID,

21 p.PerAssemblyQty,

22 bom.EndDate,

23 ComponentLevel + 1

24 FROM Production.BillOfMaterials AS bom

25 INNER JOIN Parts AS p

26 ON bom.ProductAssemblyID = p.ComponentID

27 AND bom.EndDate IS NULL

28)

29

30 SELECT AssemblyID,

31 ComponentID,

32 Name,

33 PerAssemblyQty,

34 EndDate,

35 ComponentLevel

36 FROM Parts AS p

37 INNER JOIN Production.Product AS pr

38 ON p.ComponentID = pr.ProductID

39 ORDER BY ComponentLevel, AssemblyID, ComponentID;

40

Скачать изображение

Предыдущее задание

Следующее задание

Завершить тестирование

Рисунок 41 – Пример страницы решения задания

Все дальнейшие этапы проектирования сводятся к разработке всех форм интерфейса и их подготовке для разработки. Далее приведены скриншоты форм интерфейса системы.

TestApp

Комплекты

Задания

Результаты

Выберите СУБД:

MS SQL 2016

Выберите схему:

test

TABLE

Users

Sales

Purchases

Создать новую схему

Выбор

Рисунок 42 – Форма выбора СУБД и схемы (создание задания)

TestApp

Комплекты

Задания

Результаты

1

Покажите иерархию узлов и компонентов продукции, необходимой для создания велосипеда для ProductAssemblyID = 800.

2

Выведите следующие поля на экран:

3

- AssemblyID

4

- ComponentID

5

- Name

6

- PerAssemblyQty

7

- EndDate

8

- ComponentLevel

9

Вы можете использовать таблицы BillOfMaterials и Product, которые представлены на изображении

10

Подсказка:

11

Вы можете использовать общее табличное выражение

12

13

14

15

1

WITH Parts(

2

AssemblyID,

3

ComponentID,

4

PerAssemblyQty,

5

EndDate,

6

ComponentLevel

7

) AS

8

(

9

SELECT b.ProductAssemblyID,

10

b.ComponentID,

11

b.PerAssemblyQty,

12

b.EndDate, 0 AS ComponentLevel

13

FROM Production.BillOfMaterials AS b

14

WHERE b.ProductAssemblyID = 800

15

AND b.EndDate IS NULL

16

UNION ALL

17

SELECT bom.ProductAssemblyID,

18

bom.ComponentID,

19

p.PerAssemblyQty,

20

bom.EndDate,

21

ComponentLevel + 1

22

FROM Production.BillOfMaterials AS bom

23

INNER JOIN Parts AS p

24

ON bom.ProductAssemblyID = p.ComponentID

25

AND bom.EndDate IS NULL

26

)

27

SELECT AssemblyID,

28

ComponentID, |

29

Name,

30

PerAssemblyQty,

31

EndDate,

32

ComponentLevel

33

FROM Parts AS p

34

INNER JOIN Production.Product AS pr

35

ON p.ComponentID = pr.ProductID

36

ORDER BY ComponentLevel, AssemblyID, ComponentID;

Выберите изображение

Обзор

☐

Разрешить другим менеджерам использовать данное задание

Сохранить

Рисунок 43 – Форма создания задания

TestApp

Комплекты

Задания

Результаты

Время на выполнение задания:

90

мин.

Теги:

> Средний уровень

Сохранить

Рисунок 44 – Форма дополнительной информации (создание задания)

TestApp

Комплекты

Задания

Результаты

Задания комплекта 1

700 x 300

Powered by HTML5.COM

Задание 1

Описание: Выведите все записи из таблицы USERS

Теги: Средняя сложность

Подробнее

Рисунок 45 – Просмотр заданий комплекта

Результаты

E-Mail	ФИО кандидата	Дата отправки	Дата решения	Верно решено	Подробнее
yshashkin@yandex.ru	Шашкин Юрий Анатольевич	12.05.2020 4:00	12.05.2020 4:14	1 из 1	Посмотреть решение
ivanov@mail.ru	Иванов Иван Иванович	12.05.2020 4:20	Задание еще не решено		

Рисунок 46 – Просмотр результатов

2.7 Описание компонентов системы

2.7.1 Выбор архитектурного паттерна системы

В настоящее время имеется огромное количество различных архитектурных паттернов для разработки веб-приложений. Наибольшей популярностью пользуется паттерн MVC [9] (модель-представление-контроллер). Рассмотрим каждый из компонентов данного паттерна:

- Модель предоставляет данные и методы работы с ними: запросы в базу данных, проверка на корректность. Модель не зависит от представления (не знает как данные визуализировать) и контроллера (не имеет точек взаимодействия с пользователем), просто предоставляя доступ к данным и управлению ими.
- Представление отвечает за получение необходимых данных из модели и отправляет их пользователю. Представление не обрабатывает введенные данные пользователя.
- Контроллер обеспечивает «связь» между пользователем и системой. Контролирует и направляет данные от пользователя к системе и наоборот. Использует модель и представление для реализации необходимого действия.

Данный паттерн подходит для разработки СКЗБД, так как с его помощью могут быть реализованы все компоненты и каждый из классов системы может быть классифицирован как один и компонентов паттерна.

2.7.2 Работа с БД

Существует два основных способа работы с базой данных в современных веб-приложениях: прямое подключение к базе данных и использование ORM (объектно-реляционное отображение). Прямое подключение к базе данных позволяет создавать сложную логику на языке запросов к базам данных. ORM обеспечивают простоту общения бизнес-приложения с реляционными данными, а так же обеспечивают дополнительную защиту данных (в частности – от SQL инъекций).

В разрабатываемой системе имеет смысл использовать оба варианта работы с БД. Для работы с тестовыми объектами, эталонными запросами, решениями кандидатов имеет смысл использовать прямое подключение к БД, так как объекты не фиксированы, и работа с ними непосредственно из бизнес-приложения не происходит. Для системной базы данных, объекты которой имеют фиксированную структуру, и не изменяются со временем, лучше использовать ORM, так как с ее помощью значительно облегчается работа с данными из разрабатываемой системы [10].

В качестве ORM на Java была выбрана «Hibernate ORM», в связи с тем, что она является самой обширной библиотекой, реализующей необходимые функции, и имеет огромное количество документации.

2.7.3 Проектирование классов системы

Проектирование классов следует начинать с проектирования пакетов. В соответствии с архитектурным паттерном у нас имеется 2 основных пакета: модели и контроллеры (рисунок 47).

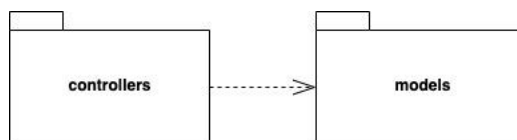


Рисунок 47 - Диаграмма пакетов верхнего уровня

Далее детализируем пакеты до следующего уровня. Пакет с моделями не имеет смысла детализировать, так как на этапе проектирования базы данных мы получили 11 таблиц, и они не нуждаются в каком-либо разделении. С контроллерами несколько иная ситуация. Детализируем контроллеры и опишем каждый из пакетов низкого уровня:

- authentication – механизм обработки входа, выхода и регистрации в системе. В перспективе возможно использование технологии ADFS (Active Directory Federation Services - это программный компонент, разработанный Microsoft, который можно установить в операционных системах Windows Server, чтобы предоставить пользователям единый вход в системы и приложения, расположенные за пределами организации [11]), так как это является более надежной технологией с возможностью гибкой настройки, в том числе с возможностью двухфакторной аутентификации.
- Results – вывод на экран менеджера результатов тестирования кандидатов, а так же детализация результатов до исходного решения кандидата.
- Connections – классы для соединения с тестовыми базами данных и работы с конфигурационными файлами.
- Mail – соединение с почтовым сервером и отправка почтовых уведомлений кандидатам.
- Packages – комплекты заданий, их составление, просмотр и редактирование.
- Rest – классы, реализующие предоставление данных через REST API.
- Tasks – классы для работы с заданиями, их составление, просмотр и изменение.
- Tests – работа с тестовыми заданиями с точки зрения кандидата. Открытие полученного по электронной почте комплекта заданий, сохранение выполненного задания, отслеживание прогресса выполнения и проверка выполненных заданий.

На рисунке 48 представлена диаграмма пакетов с описанными прежде пакетами.

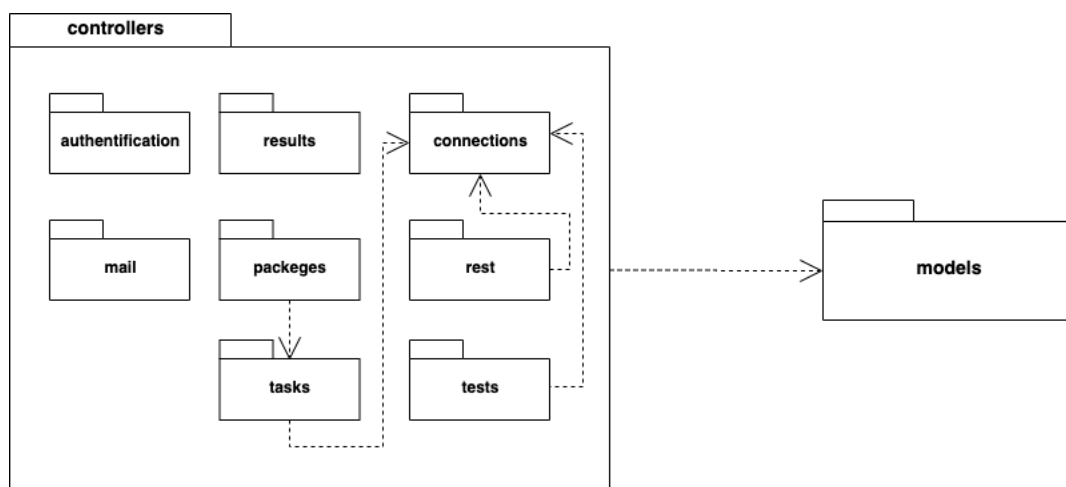


Рисунок 48 – Диаграмма пакетов

Имея диаграмму пакетов, можно задуматься о диаграмме классов. Так как мы имеем дело с веб приложением, то число используемых библиотечных классов может быть очень велико и не представляет интереса на диаграмме классов, то не будем отражать их на диаграмме классов и оставим решение об использовании таких классов на совести разработчиков. [12] В связи с тем, что в каждом из пакетов будет находиться несколько классов и связи вне пакета классы имеют в основном с моделями, то рассмотрим каждый пакет по-отдельности.

На рисунке 49 представлена диаграмма классов моделей. Классы моделей не имеют сложной логики обработки данных – в данных классах реализуются лишь функции сохранения и получения данных их БД. Название каждого класса соответствует названию таблицы базы данных. Параметры каждого класса – столбцы таблицы и поля, на которые ссылаются вторичные ключи. Методы, реализуемые в данных классах, представляют из себя «геттеры» и «сеттеры» (методы для получения и записи данных в private переменные).

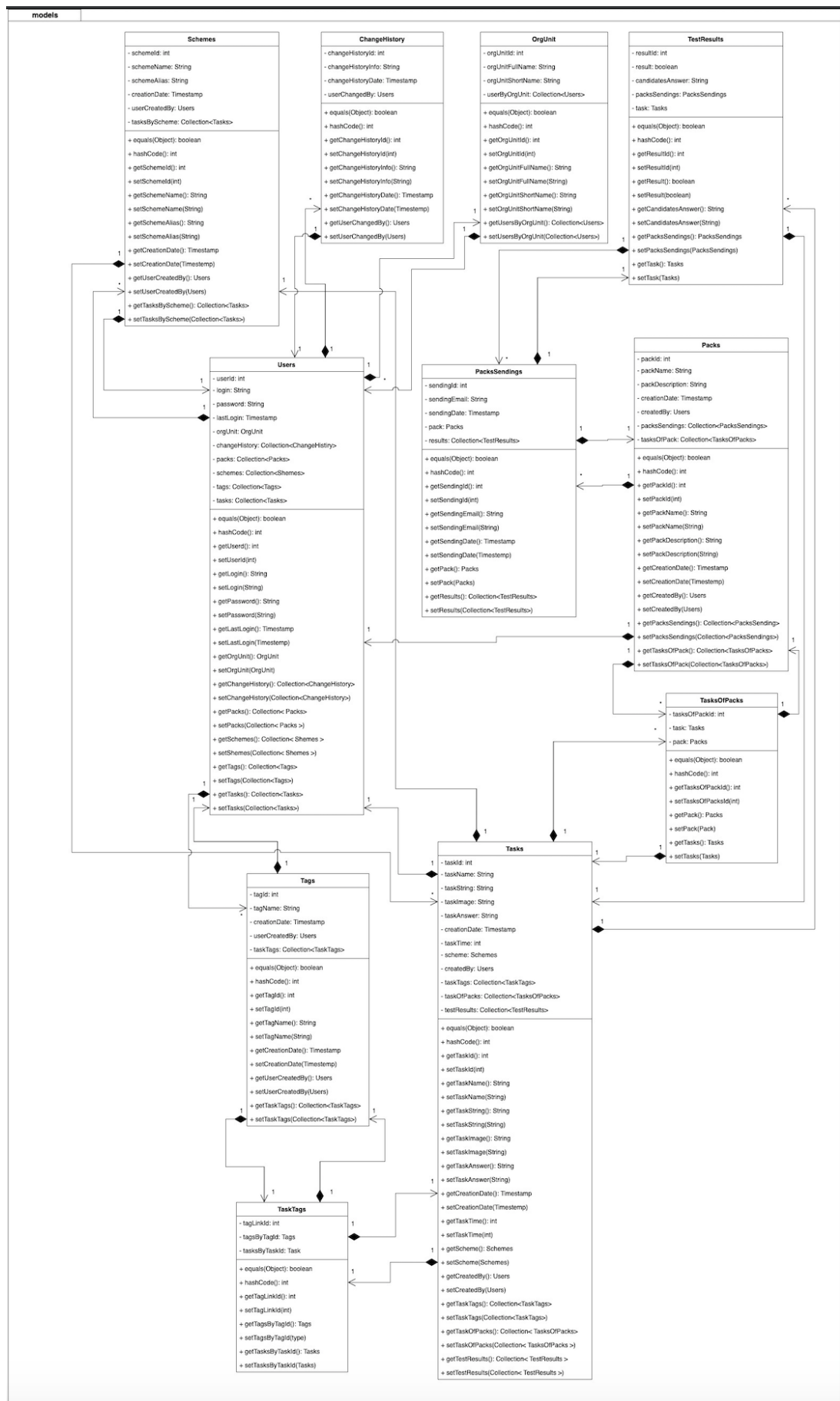


Рисунок 49 – Диаграмма классов пакета models

На рисунках 50-57 представлены диаграммы классов пакета controllers. Классы, представленные на данных рисунках, содержат небольшое количество методов и их названия полностью отражают выполняемые ими функции, в связи с чем дополнительное их описание не требуется.

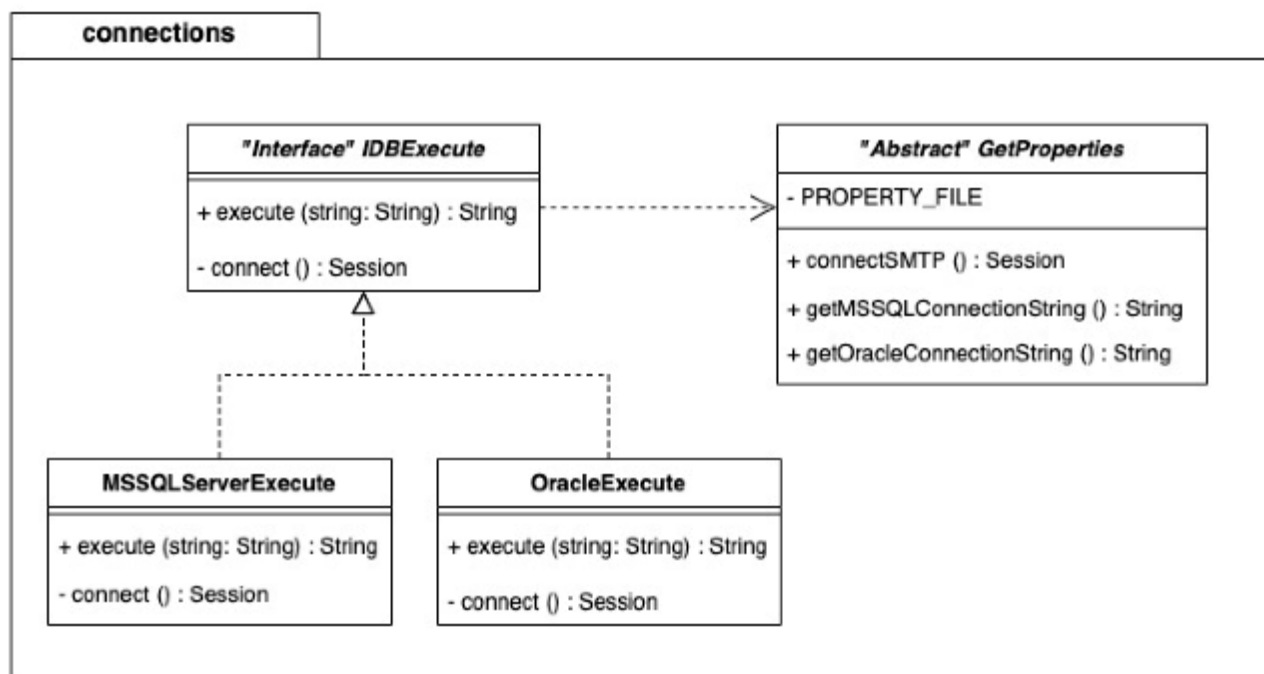


Рисунок 50 – Диаграмма классов пакета connections

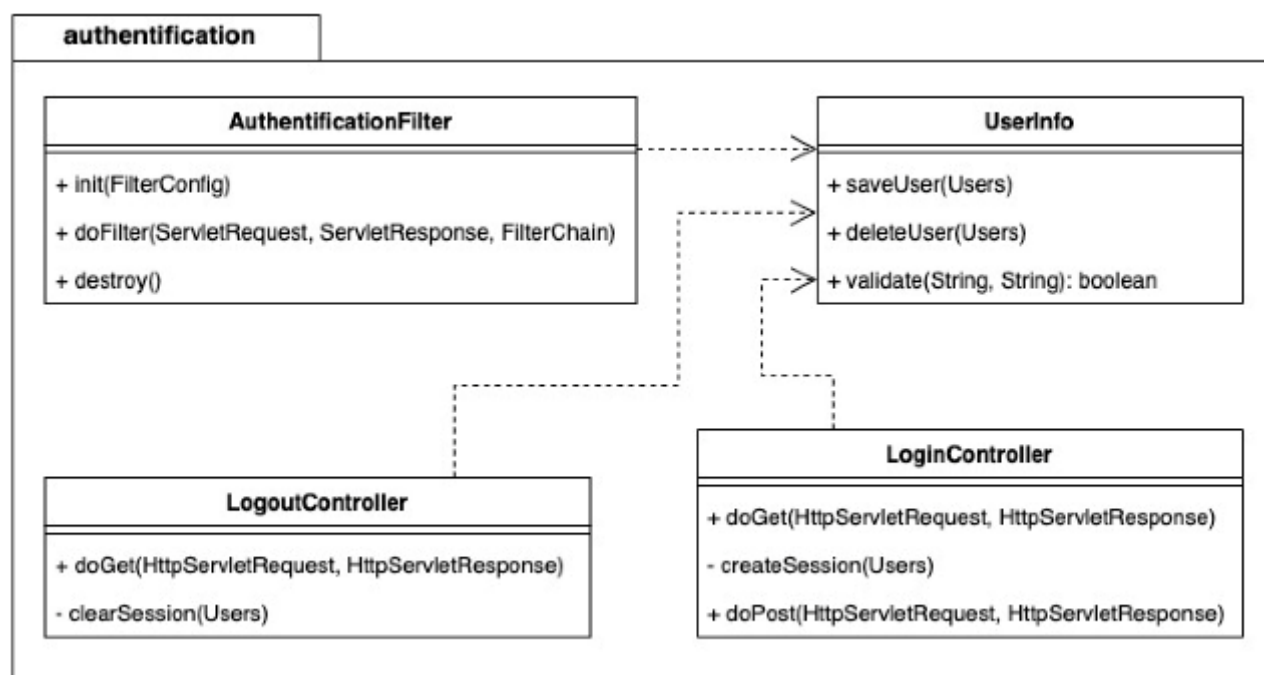


Рисунок 51 – Диаграмма классов пакета authentication

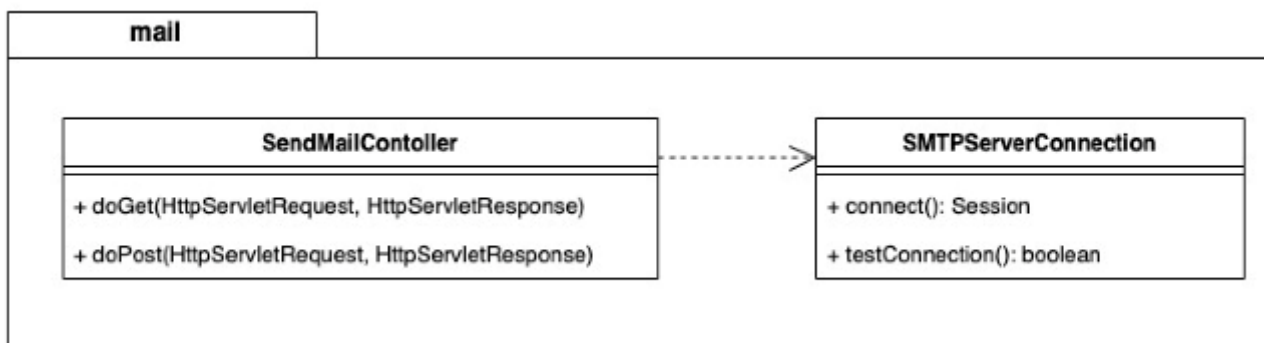


Рисунок 52 – Диаграмма классов пакета mail

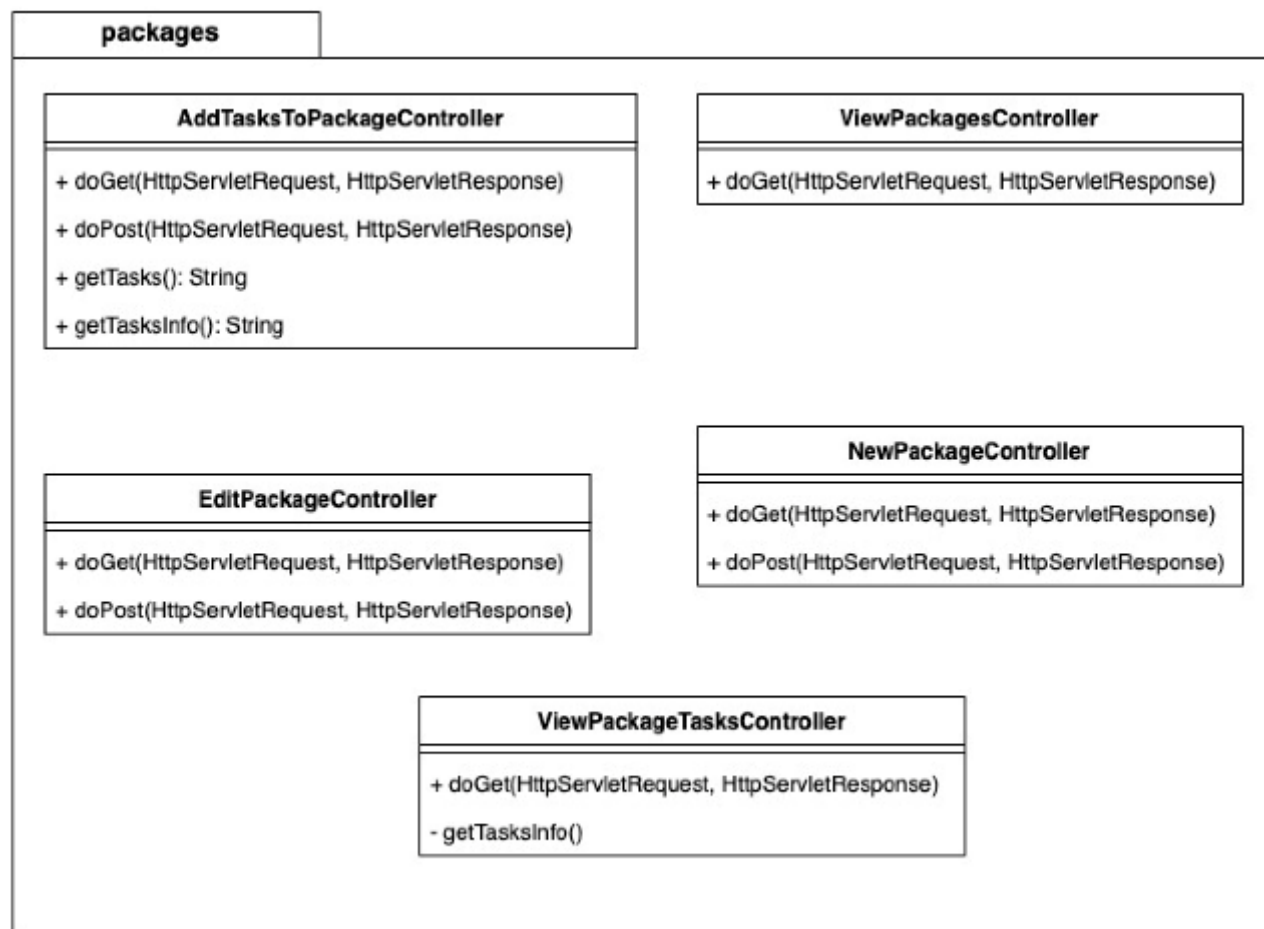


Рисунок 53 – Диаграмма классов пакета packages

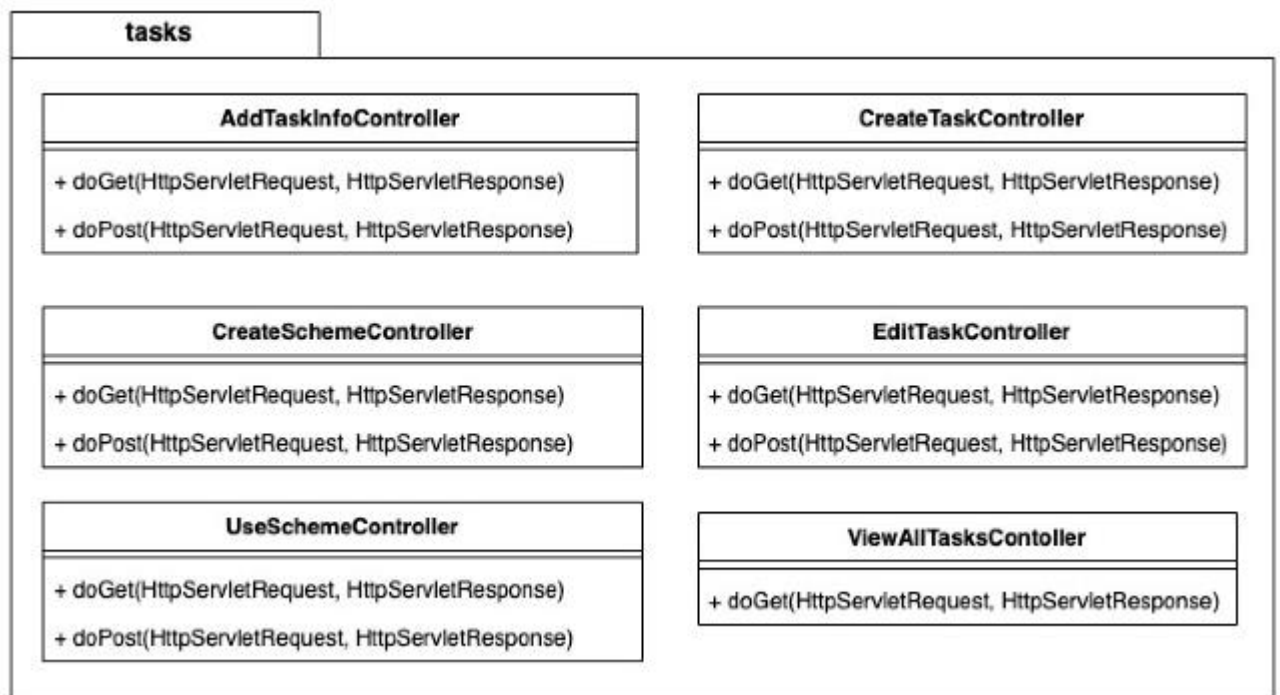


Рисунок 54 – Диаграмма классов пакета tasks

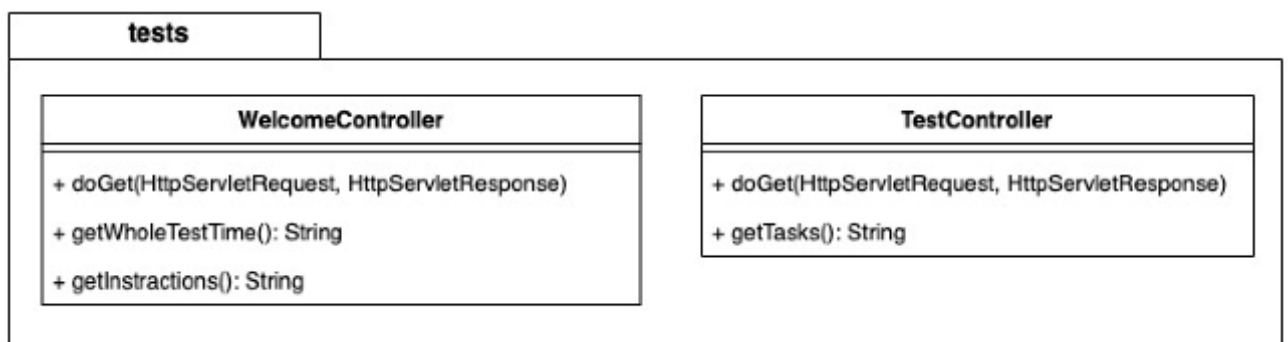


Рисунок 55 – Диаграмма классов пакета tests



Рисунок 56 – Диаграмма классов пакета results

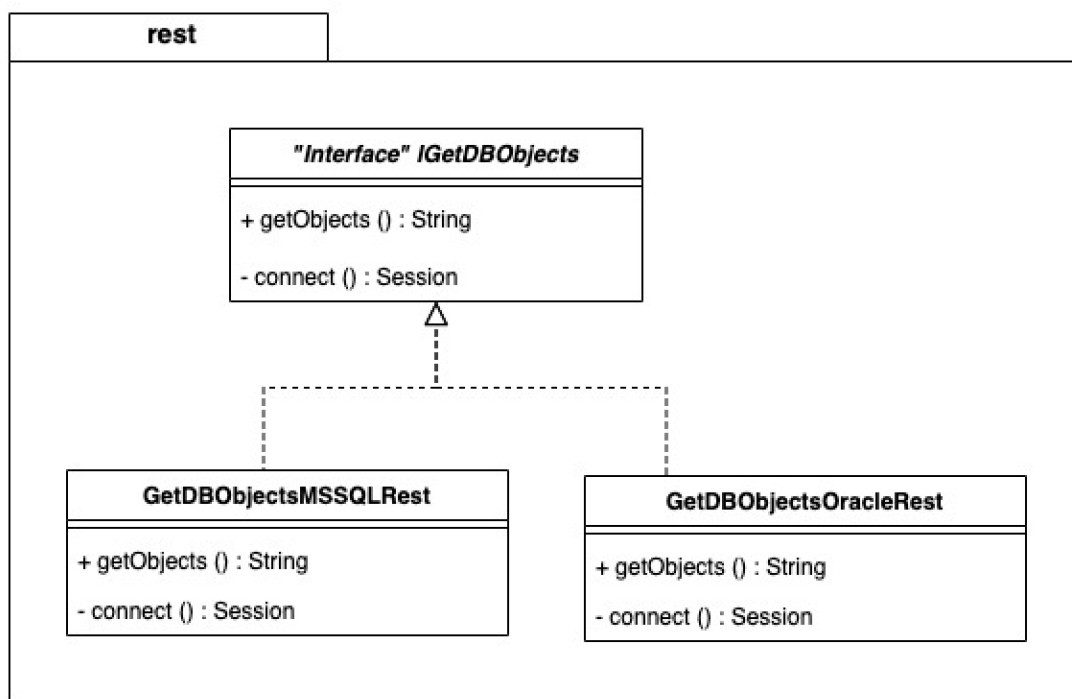


Рисунок 57 – Диаграмма классов пакета rest

2.7.4 Настройка отправки почты

Отправка почты может быть реализована с помощью различных протоколов. Самым простым в реализации протоколом является протокол SMTP (Simple Mail Transfer Protocol). Данный протокол не подходит для передачи конфиденциальной информации, однако, данные, которые будут передаваться по электронной почте не будут содержать информации такого рода. В связи с этим, использование данного протокола для отправки электронной почты является приемлемым.

Для настройки подключения к почтовому серверу требуются данные для соединения. Сохраним данные в специальный конфигурационный файл, который будет храниться в системе и читаться при запуске сервера, а затем сохраняться в оперативной памяти, что позволит повысить быстродействие системы.

Для отправки письма можно использовать отдельный файл-шаблон, что позволит добиться гибкой настройки стилей письма и информации,

содержащейся в нем без необходимости перезапуска сервера и изменения исходного кода сервера.

2.7.5 Разработка функций IDE для форм ввода

Для реализации функций простейшей IDE с нумерацией строк, выделением ключевых слов может быть использована javascript библиотека. Одной из самых популярных js библиотек для реализации требуемых функций является библиотека ACE. Среди основных преимуществ данной библиотеки можно выделить следующие:

- подсветка синтаксиса более чем 45 языков программирования, в том числе SQL,
- более 20 тем оформления,
- опциональная командная строка,
- возможность загрузки больших документов (до 4 млн строк),
- возможности поиска по тексту с помощью регулярных выражений,
- возможность использования нескольких курсоров в одном окне редактирования кода.

Все эти функции очень полезны для задач, когда требуется писать код на веб-странице. Пример работы данной библиотеки с официального сайта представлен на рисунке 58.

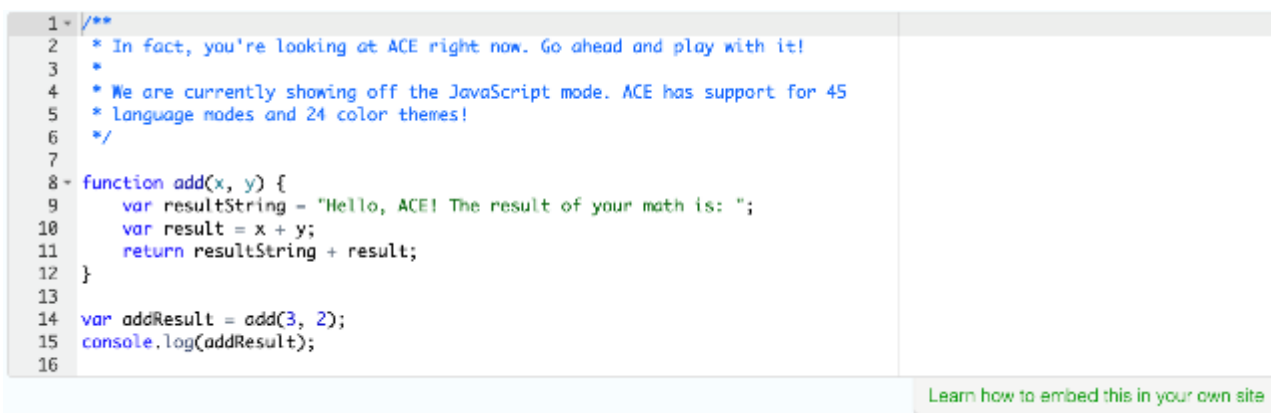
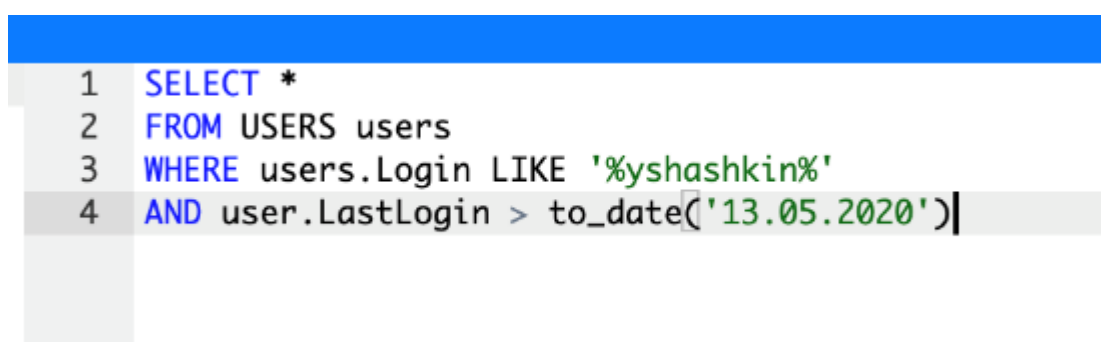


Рисунок 58 – Пример использования библиотеки ACE

Подключение библиотеки происходит путем загрузки кода библиотеки на страницу, а затем его запуска с помощью js скрипта. Данная библиотека позволяет производить гибкую настройку поля ввода для различных целей. Например, поле ввода для SQL скрипта должно подсвечивать синтаксис именно этого языка, что делается при помощи установки параметра Mode = "ace/mode/sql". Пример работы с таким параметром представлен на рисунке 59. Для проведения тестирования будет полезно ограничить ввод информации в поле с текстом задания, чтобы пользователь случайно не стер часть важной информации.



```
1 SELECT *
2 FROM USERS users
3 WHERE users.Login LIKE '%yshashkin%'
4 AND user.LastLogin > to_date('13.05.2020')
```

Рисунок 59 – Пример подсветки SQL синтаксиса

Данная библиотека не обладает функциональностью выделения ключевых слов процедурных расширений языка SQL (PL\SQL и T-SQL). Однако, это не является серьезной проблемой, так как для визуального форматирования кода необходимы именно ключевые слова стандартного языка, выделение которых нормально работает в данной библиотеке.

3 Тестирование и перспективы развития системы

3.1 Тестирование

Тестирование системы является важной составляющей в жизненном цикле любого продукта, который будут использовать реальные пользователи. Существуют общие подходы к тестированию, однако для всестороннего полного тестирования разрабатываемого продукта следует учитывать специфику продукта, чтобы понимать наиболее критичные места, отказ работы которых может повлечь потерю данных, остановку системы и финансовые издержки. Самыми эффективными общими подходами к тестированию являются методы «черного ящика» и «белого ящика».

Тестирование по методу «белого ящика» подразумевает проверку всех возможных условий и маршрутов программы. Так как система не имеет особых разветвлений в самых сложных алгоритмах (алгоритмы создания объектов базы данных и сохранения задания), то тестирование по принципу «белого ящика» может быть заменено автоматическим тестированием с написанием тестов, которые позволят проверить ограничения, связанные с ролями пользователей, навигацией по страницам в зависимости от входных данных и обработкой входных данных.

Тестирование по методу «черного ящика» предполагает, что тестировщик не знает, что происходит в системе, а знает лишь то, что система должна выдавать на выходе при получении определенных данных на вход. В случае СКЗБД целесообразно провести тестирование безопасности, чтобы выявить вероятность потери данных от неожиданных действий пользователя; тестирование интерфейса пользователя для оценки удобства использования системы в целом и проверки обработки сообщений об ошибках при вводе некорректных данных; тестирования компонентов SQL для выявления, какие функциональные особенности языка SQL (например, оконные функции, подзапросы, общие табличные выражения) корректно работают при выполнении задания в системе, а какие нет.

3.1.1 Автоматическое тестирование

Автоматическое тестирование позволит проверить, что программный код системы соответствует требованиям, ролевая модель пользователей системы работает корректно. Кроме того, авто-тесты позволят поддерживать разработанный функционал в рабочем состоянии при внесении изменений и доработке системы путем запуска всех автоматических тестов при сборке проекта. Если хоть один тест не получит ожидаемый результат, то разработчику будет выведено сообщение об ошибке и сборка не будет завершена успешно.

В ходе разработки автоматических тестов было написано 38 интеграционных тестов, которые проверяют, что неавторизованные пользователи не имеют доступа к страницам, предназначенным для менеджеров и ссылки, генерируемые для отправки комплектов заданий кандидату по электронной почте, доступны кандидату без дополнительной авторизации.

Для тестирования функциональности было написано 52 юнит-теста, среди которых находятся проверки того, что все объекты модели создаются в базе данных и сохраняются при соответствующем запросе; контроллеры системы выдают ожидаемый результат при вводе корректных данных и ожидаемые ошибки при вводе некорректных.

3.1.2 Тестирование безопасности

В тестировании безопасности СКЗБД можно выделить 2 основных этапа: проверка корректности работы ролевой модели и проверка невозможности SQL-инъекций (внедрение в данные, передаваемые через GET, POST запросы или значения Cookie, произвольного SQL кода), так как именно эти этапы позволят не допустить потерю данных и нарушение их целостности при некорректных действиях пользователей. Тестирование ролевой модели было проведено с помощью автоматических тестов, поэтому в данном пункте рассмотрим невозможность SQL-инъекций. В качестве кода для SQL-инъекции будем использовать код «DROP TABLE USERS». В таблице 3 представлены результаты тестирования безопасности совместно с описанием самого тестирования.

Таблица 3 – тестирование безопасности

Тест	Ожидаемый результат	Полученный результат	Оценка
Ввод кода в поле ввода ответа на форме решения задания	Ошибка	Ошибка	Успешно
Ввод кода в поле ввода ответа на форме создания задания	Ошибка	Ошибка	Успешно
Ввод кода в поле ввода задания на форме создания задания	Ошибка	Ошибка	Успешно
Ввод кода в поле ввода ограничения времени на форме создания задания	Ошибка	Ошибка	Успешно
Ввод кода в файле для создания схемы БД	Ошибок нет, команда выполнена, данные в основной базе не повреждены	Ошибок нет, команда выполнена, данные в основной базе не повреждены	Успешно
Ввод кода в поле названия комплекта на форме создания комплекта	Ошибка	Ошибка	Успешно
Ввод кода в поле электронный почты кандидата	Ошибка	Ошибка	Успешно

Из приведенной выше таблицы можно сделать вывод, что система не подвержена SQL-инъекциям. Это связано в первую очередь с несколькими удачными архитектурными решениями, заложенными на этапе проектировании системы, а именно:

- использование ORM;
- использовании другой базы данных для тестовых заданий;
- использование пользователей без прав на выполнение DDL (Data Definition Language, DDL командами являются CREATE, ALTER, и DROP) команд для выполнения тестовых заданий.

3.1.3 Тестирование интерфейса пользователя

Тестирование интерфейса пользователя позволит выявить основные направления действий пользователей, понять какие ошибки были допущены при проектировании интерфейса и корректно ли обрабатываются действия пользователей, если они не соответствуют изначальному бизнес-процессу. Для проведения такого тестирования было выбрано 3 респондента, которым сначала было предложено выполнить тестирование в качестве кандидата, а затем создать комплект заданий и отправить его кандидату в качестве менеджера. Контрольная группа была составлена из людей, не имеющих отношения к системе и не знакомой с изначальными требованиями к ней.

На основе результатов опроса респондентов была составлена таблица 4.

В результате ответов респондентов можно сделать следующие выводы:

- интерфейс кандидата доработок не требует, имеет смысл поработать надо нотификацией, отправляемой системой, чтобы она соответствовала концепции системы.
- Интерфейс менеджера нуждается в дополнительных кнопках, поясняющих функционал. Проблема на первое время может быть решена руководством пользователя и встречей для демонстрации функционала системы менеджерам, так как их количество невелико.

- Требуется добавить проверку на корректность введенного email.

Таблица 4 – результаты тестирования интерфейса пользователя

Респондент	Комментарий по интерфейсу кандидата	Комментарий по интерфейсу менеджера	Обработка некорректных действий
1	Все разъяснения о порядке выполнения тестирования получены на начальном экране тестирования, проблем не возникло	Имеет смысл добавить дополнительные инструкции для менеджера о взаимодействии с полями. Например, как это реализовано в codewars. В остальном, все хорошо: механика меню продумана отлично, комплекты и задания работают, цветовая гамма выбрана удачно	Сломать систему не получилось

Продолжение таблицы 4

2	Все тестирование предельно понятно. Возможно, стоит подумать над оформлением письма, которое приходит на почту, чтобы оно было стилизовано под интерфейс системы	Интерфейс отлично продуман, интуитивно понятен, проблем с использованием на персональном компьютере не возникло	Нужно добавить проверку корректности введенного email для отправки кандидату. Остальные поля корректно реагируют на неверные действия пользователя
3	Проблем не обнаружено, тестирование пройдено	Интерфейс понятен, вопросов по использованию системы нет.	Ошибок не выявлено

3.1.4 Тестирование компонентов SQL

Тестирование компонентов SQL заключается в проверке корректности обработки запросов, содержащих особые компоненты языка. Такое тестирование целесообразно проводить для каждой СУБД по-отдельности, так как они имеют свои специфические функции и процедурные расширения.

Далее представлено тестирование компонентов на базе СУБД MS SQL Server 2016.

Таблица 5 – тестирование компонентов T-SQL

Номер	Запрос содержит компонент	Результат
1	TOP(X)	Успешно
2	LEFT JOIN	Успешно
3	INNER JOIN	Успешно
4	FULL JOIN	Успешно
5	EXCEPT	Успешно
6	INTERSECT	Успешно
7	COALESCE	Успешно
8	Агрегирующие функции (SUM, MAX)	Успешно
9	GROUP BY	Успешно
10	ORDER BY	Успешно
11	HAVING	Успешно
12	PIVOT/UNPIVOT	Успешно
13	Общие табличные выражения (CTE)	Успешно
14	Оконные функции (OVER)	Успешно
15	RANK	Успешно
16	Использование переменных (DECLARE/SET)	Ошибка
17	APPLY	Успешно
18	WHERE	Успешно

Продолжение таблицы 5

19	FOR XML	Успешно
20	FOR JSON	Успешно
21	BEGIN...END	Ошибка
22	WHILE	Ошибка
23	BREAK/CONTINUE	Ошибка
24	CURSOR	Успешно

По результатам тестирования можно сделать вывод, что СКЗБД хорошо работает с запросами без процедурных расширений, однако вызывает ошибки при выполнении запросов с циклами/переменными и иными процедурными расширениями языка. Это не является критичной проблемой, так как разработка сложных запросов с использованием процедурных расширений очень сложна в условиях тестовой системы без полноценного подключения к СУБД. Данная проблема может быть решена с помощью изменения алгоритма проверки тестового задания – сохранять запрос пользователя в качестве хранимой процедуры и выполнять эталонный запрос и запрос кандидата через процедуры.

Далее представлено тестирование компонентов на базе СУБД Oracle 11G.

Таблица 6 – тестирование компонентов PL/SQL

Номер	Запрос содержит компонент	Результат
1	LEFT JOIN	Успешно
2	INNER JOIN	Успешно

Продолжение таблицы 6

3	FULL JOIN	Успешно
4	MINUS	Успешно
5	COALESCE	Успешно
6	Агрегирующие функции (SUM, MAX)	Успешно
7	GROUP BY	Успешно
8	ORDER BY	Успешно
9	HAVING	Успешно
10	To_date	Успешно
11	Общие табличные выражения (CTE)	Успешно
12	Оконные функции (OVER)	Успешно
13	RANK	Успешно
14	Использование переменных (DECLARE/SET)	Ошибка
15	APPLY (CROSS/OUTER)	Успешно
16	WHERE	Успешно
17	BEGIN...END	Ошибка
18	WHILE	Ошибка
19	BREAK/CONTINUE	Ошибка
20	CURSOR	Успешно

По результатам тестирования можно сделать вывод, аналогичный выводу из тестирования компонентов T-SQL. Процедурные расширения языка работают

некорректно, проблема может быть решена с помощью использования хранимой процедуры. Решение данной ошибки не является критически важным, поэтому оно останется на следующие циклы разработки.

3.2 Перспективы развития

СКЗБД, будучи системой, позволяющей отслеживать знания пользователей в области языка запросов к базе данных, имеет широкие перспективы развития в различных направлениях:

- добавление функционала обучения;
- развитие тестирования других языков программирования;
- развитие других видов тестирования (тесты, задания с текстовым развернутым ответом);
- интеграция с Moodle для учебных заведений.

Первые два пункта возможны в реализации, однако это требуют огромных затрат на аналитику, разработку и тестирование. Кроме того, система позиционируется в качестве системы контроля знаний, поэтому вводить в нее функционал обучения является нелогичным.

Рассмотрим последние два пункта более подробно.

3.2.1 Развитие других видов тестирования

Под развитием других видов тестирования подразумевается создание средств для составления заданий с выбором ответа из списка и заданий с развернутым ответом.

Задания с выбором ответа из списка не имеют особого смысла в ситуации удаленного прохождения тестирования, так как ответы на них обычно легко найти в интернете, а такие вопросы нацелены именно на знание материала тестируемым. При добавлении такого функционала в систему СКЗБД можно будет проводить тестирование на базе предприятия в таком же виде, в каком проводятся экзамены Microsoft и Oracle. Для примера рассмотрим, экзамен 70-

461 от Microsoft по написанию запросов на языке T-SQL. Данный экзамен содержит около 70 вопросов, около 50 из которых – тестовые вопросы с выбором ответа (с возможностью выбора нескольких правильных вариантов ответа). Так как такой серьезный экзамен содержит вопросы с выбором ответа, то они действительно могут хорошо оценить знания тестируемого.

Задания с развернутым ответом позволят включить в техническое собеседование какие-либо логические задачи, вопросы на понимание систем. В перспективе можно будет использовать вопросы такого вида для архитектурных вопросов (например, как бы вы организовали систему с такими-то модулями). Такие задания позволят повысить вариативность тестирования и дают возможность менеджеру выбирать, как и какие знания он хочет проверять у тестируемого.

3.2.2 Интеграция с Moodle

Moodle - самая популярная система управления обучением (LMS) с открытым кодом в мире. Интеграция СКЗБД с Moodle позволит значительно увеличить аудиторию пользователей системы, так как контроль знаний языка запросов к базам данных является одной из наиважнейших частей обучения представителей технических профессий.

Существует множество руководств по настройке интеграций сторонних приложений с Moodle. Такая интеграция может быть совершена 3 основными способами [13]:

- файловая интеграция – наиболее ручной из всех методов. Процесс очень прост. Создается файл, обычно в формате .csv, содержащий информацию, которую вы хотите загрузить в Moodle. Существуют разные требования к разметке файлов в зависимости от того, что вы загружаете. Есть способы автоматизировать этот процесс немного больше. Файлы могут быть загружены в пункт назначения через SFTP, например. Затем можно запланировать задание на сервере Moodle, чтобы считать новый файл,

который прибыл в пункт назначения, а затем обработать его автоматически.

- Внешняя база данных – подключение внешней БД и выгрузка данных оттуда в БД Moodle.
- Web API – программный интерфейс для взаимодействия компонентов программного обеспечения. Именно этот вариант предоставляет широкий спектр возможностей добавления функционала.

Вариант Web API является наиболее предпочтительным, так как не требует большого количества доработок со стороны Moodle. СКЗБД будет получать запросы от Moodle и возвращать необходимые данные в зависимости от действий пользователя. Проверка тестовых заданий, их хранение, работа с комплектами заданий будет происходить через REST API СКЗБД. Разработка данного функционала не потребует большого количества трудозатрат, так как вся обработка уже реализована, изменяется лишь средство представления данных.

ЗАКЛЮЧЕНИЕ

В соответствии с техническим заданием была разработана система контроля знаний языка запросов к базе данных в качестве выпускной квалификационной работы бакалавра.

Система уменьшает трудозатраты менеджеров ИТ-компаний на проведение технических собеседований на должности, связанные с работой с базами данных. Анализ существующих аналогов позволил выявить основные технические характеристики будущей системы, которые необходимы для успешного выполнения поставленной задачи.

На основе данных, полученных на этапе анализа, было выполнено проектирование компонентов системы, произведен выбор языка и средств разработки, были разработаны модули системы.

Была разработана технология тестирования программной системы и проведены автоматическое тестирование, тестирование безопасности, тестирование пользовательского интерфейса и тестирование компонентов системы. В результате тестирования были сделаны выводы, что система удовлетворяет поставленной задаче. Кроме того, были рассмотрены различные перспективы развития программного продукта. Самыми перспективными были выбраны развитие других видов тестирования и интеграция с системой электронного обучения Moodle.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. GeekBrains – образовательный портал [Электронный ресурс] <https://geekbrains.ru/> (дата обращения 23.11.2019).
2. Codewars [Электронный ресурс] <https://www.codewars.com/> (дата обращения 23.11.2019).
3. Codewars.com wiki [Электронный ресурс] <https://www.codewars.com/> (дата обращения 23.11.2019).
4. Teach coding with games: a review of Codewars and CodeCombat [Электронный ресурс] <https://opensource.com/education/15/7/codewars-codecombat-review> (дата обращения 23.11.2019).
5. Why Codewars is the best way to learn a new programming language! [Электронный ресурс] <https://opensource.com/education/15/7/codewars-codecombat-review> (дата обращения 23.11.2019).
6. Code Wars: Creating a Problem / Kata [Электронный ресурс] https://www.youtube.com/watch?v=Ytl4WPJ_zgA (дата обращения 23.11.2019).
7. How to Create a Kata in Codewars [Электронный ресурс] <https://www.youtube.com/watch?v=TZTnP3bZZSU> (дата обращения 23.11.2019).
8. Этапы разработки интерфейса [Электронный ресурс] <https://designpub.ru/%D1%8D%D1%82%D0%B0%D0%BF%D1%8B-%D1%80%D0%B0%D0%B7%D1%80%D0%B0%D0%B1%D0%BE%D1%82%D0%BA%D0%B8-%D0%B8%D0%BD%D1%82%D0%B5%D1%80%D1%84%D0%B5%D0%B9%D1%81%D0%B0-baf666d6ad8f> (дата обращения 25.05.2020).
9. Model-View-Controller [Электронный ресурс] <https://ru.wikipedia.org/wiki/Model-View-Controller> (дата обращения 25.05.2020).
10. Hibernate Tutorial [Электронный ресурс] <https://www.javatpoint.com/hibernate-tutorial> (дата обращения 25.05.2020).

- 11.Что такое ADFS (службы федерации Active Directory)? [Электронный ресурс] <https://qastack.ru/server/708669/what-is-adfs-active-directory-federation-services> (дата обращения 25.05.2020).
- 12.Иванова Г.С. Основы программирования: Учебник для втузов. – 4-е изд. М.: Изд-во МГТУ им. Н.Э. Баумана, 2007.
- 13.Moodle Integrations: Let Me Count the Ways [Электронный ресурс] <https://ethinkeducation.com/blog/moodle-integrations-ways/> (дата обращения 25.05.2020).

Приложение А
Техническое задание
Листов 9

Приложение Б

Руководство пользователя для менеджера

Листов 8

Приложение В

Фрагменты исходного кода программы

Листов 10

Приложение Г

Копии листов графической части

Листов 6