МИНОБРНАУКИ РОССИИ

Федеральное государственное образовательное учреждение

высшего образования

**«Московский технологический университет»**

**МИРЭА**

**Колледж приборостроения и информационных технологий**

ДИПЛОМНАЯ РАБОТА

СПЕЦИАЛЬНОСТЬ 09.02.03

**«Программирование в компьютерных системах»**

**На тему:**

«Разработка автоматизированной системы интерактивной проверки знаний (тестирования) в среде Visual Studio»

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Выполнил студент | | | | |
| группы |  | ЩПКО-01-13 (ПКС-41) | | |
|  | | |  | Гаганова В.С. |
| подпись | | |  | ФИО студента |
| Руководитель | | |  |  |
|  | | |  | Кузнецова Ю.В. |
| подпись | | |  | ФИО руководителя |
| Нормоконтроль | | |  |  |
|  | | |  | Власова И.М. |
| подпись | | |  | ФИО контролёра |

**ЗАДАНИЕ**

на дипломную работу

студенту 4 курса группы ЩПКО-01-13(ПКС-41)

по специальности 09.02.03

"Программирование в компьютерных системах"

Гагановой Валерии Сергеевне

ТЕМА ЗАДАНИЯ: " Разработка автоматизированной системы интерактивной проверки знаний (тестирования) в среде Visual Studio "

**Дипломная работа выполняется студентом колледжа в следующем объеме:**

Техническое задание.

**I Пояснительная записка**

*Введение*

*Анализ технического задания.*

**1. Исследование существующих решений**

*1.1. Исследование существующих решений в области систем тестирования.*

*1.2. Анализ входных и выходных данных программы.*

**2. Проектирование системы тестирования**

*2.1. Выбор методологии*

*2.2. Выбор инструментального средства проектирования*

*2.3. Проектирование системной архитектуры*

*2.4. Структура базы данных*

**3. Разработка автоматизированной системы тестирования**

*3.1. Выбор средств реализации системы*

*3.2. Разработка пользовательского интерфейса*

*3.3. Контрольный пример*

**Дата выдачи** 06.04.2017

**Срок окончания** 14.06.2017

|  |  |  |
| --- | --- | --- |
| **Председатель ПЦК Компьютерные системы** |  | А.В. Беседин |
| **Руководитель дипломной работы** |  | Ю.В. Кузнецова |

# СОДЕРЖАНИЕ

ВВЕДЕНИЕ 2

АНАЛИЗ ТЕХНИЧЕСКОГО ЗАДАНИЯ 5

1. ИССЛЕДОВАНИЕ ПРЕДМЕТНОЙ ОБЛАСТИ 7
   1. Автоматизированные системы тестирования 7
   2. Постановка задачи и определение основных требований к

разрабатываемой системе 9

* 1. Анализ существующих разработок 10

1. **ПРОЕКТИРОВАНИЕ СИСТЕМЫ ТЕСТИРОВАНИЯ** 11
   1. Выбор методологии 11
   2. Выбор инструментального средства проектирования 13
   3. Проектирование системной архитектуры 16
   4. Структура базы данных 23
2. **РАЗРАБОТКА АВТОМАТИЗИРОВАННОЙ СИСТЕМЫ ИНТЕРАКТИВНОЙ ПРОВЕРКИ ЗНАНИЙ СТУДЕНТОВ** 25
   1. Выбор средств реализации системы 25
   2. Разработка пользовательского интерфейса 29
   3. Контрольный пример 33

ЗАКЛЮЧЕНИЕ 35

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 37

ПРИЛОЖЕНИЕ 1. Код программы 40

ПРИЛОЖЕНИЕ 2. Руководство пользователя 55

# Техническое задание

Разработать автоматизированную систему тестирования для студентов с возможностью создавать и редактировать тесты преподавателями.

1. Назначение системы
2. Создание и хранение информации о пользователях и тестах
3. Прохождение пользователями тестов
4. Обработка и анализ пройденных тестов
5. Основные цели создания системы

Основной целью АС «Тестер» является автоматизация прохождения тестирования и дальнейшего анализа с целью упрощения процесса оценивания знаний.

1. Требования к структуре и функционированию системы

Пользователю потребуется компьютер на базе ОС семейства Windows NT не ниже версии 6.1, не менее 2 Гб оперативной памяти. Для корректной работы необходимо дополнительно программное обеспечение в виде сервера MySQL Community server и ПО для администрирования MySQL Work Bench, так же .NET Framework не ниже 4.5.1 для поддержки графической оболочки программы.

1. Требования к персоналу системы

Минимальные требования к администратору системы – далее Администратор:

- Умение поддерживать систему в работоспособном состоянии и устанавливать АС «Тестер» на ОС, подходящую под условия функционирования.

Минимальные требования к продвинутому пользователю системы – далее Преподаватель:

- Умение грамотно создавать тесты в рамках АС «Тестер».

Минимальные требования к рядовому пользователю системы – далее Студент

- Умение пользоваться интерфейсом в рамках прохождения тестов.

# Введение

В наше время, время информационных технологий и автоматизации всё сложнее делать много бумажной работы, особенно преподавателям, ведь всё чаще мы используем компьютерные системы и различное программное обеспечение, которое нацелено на упрощение множества повседневных и трудоемких задач.

Недостатком программированного контроля в его до компьютерном виде — высокая трудоемкость создания программированных карт, которые требовались на каждое занятие, сложность их последующей обработки. С появлением компьютерных технологий у преподавателей появилась возможность резко снизить трудоемкость и подготовки контроля, и обработки результатов.

Что спрашивать? Вопрос далеко не праздный. Дело в том, что в попытках привести все предметы к передовой технологии преподаватели зачастую не отдают себе отчет, что программированный контроль может и должен использоваться только в тех предметных областях, которые поддаются формализации. Внедрение программированного контроля в таких дисциплинах, как философия, литература, риторика, большей частью не только бесполезны, но и крайне вредны, так как формализуют попытки мышления там, где им надо давать полную свободу.

Программированный контроль знаний идеален там, где на каждый вопрос существует один или несколько конкретных правильных ответов, где вопрос не является двусмысленным, многозначным, неопределенным или зависящим от конкретной научной школы. Последний момент, кстати, гораздо шире, чем может показаться на первый взгляд. Практика показывает, что даже в пределах одной учебной кафедры преподаватели способны использовать — и спрашивать со студентов — совершенно различную терминологию.

Одной из главных проблем разработки контролирующих, педагогических, да и вообще специализированных программ являются их разработчики. С самого начала «компьютеризации» возникла следующая ситуация, которая не исправилась и по сей день: специалисты, обладающие знаниями в конкретной предметной области (в нашем случае — преподаватели какого-то предмета), существенно боятся компьютера, не понимают и сами всячески критикуют компьютерные технологии. В тех редких случаях, когда преподаватель все-таки садится за компьютер, все его попытки что-либо создать завершаются, как правило, рождением очередного контролирующего теста, концентрирующего в себе все возможные ошибки, как программирования, так и педагогики.

# ИССЛЕДОВАНИЕ ПРЕДМЕТНОЙ ОБЛАСТИ

* 1. Автоматизированные системы тестирования

В данное время существует множество автоматизированных систем, предназначенных для тестирования знаний студентов. Рассмотрению подлежат программы, ориентированные на специфику обучения в отечественных образовательных заведениях, также, будет рассмотрена отдельно взятая зарубежная программа тестирования для лучшего понимания и сравнения. Рассматриваться будут следующие параметры:

1. Возможности системы
2. Платформа
3. Разделение на ролевые модули
4. Достоинства и недостатки системы

Система Интерактивного Тестирования Знаний *«*СИнТеЗ*»*

1. Система имеет возможность разбивать тесты на отдельные темы, указывать уровень сложности и перемешивать темы, вопросы и варианты ответов, составлять сценарии тестирования, вывод статистики, создание отчетов.
2. Платформа: связка PHP и MySQL – устанавливается на один мощный компьютер-сервер и доступна для всех компьютеров, включенных в сеть.
3. Разделена на три ролевых модуля – завуч, учитель, ученик.

Завуч выступает в роли администратора и выполняет всю подготовительную работу – наделяет пользователей правами в ролевом модуле, создает и редактирует записи о предметах, учителях, учениках, классах.

Учитель создает и редактирует тесты, публикует их и проверяет результаты тестирования.

Ученик сдает тесты.

1. Среди достоинств системы можно отметить создание отчетов по классу и ученику, где можно увидеть среднюю успеваемость учеников, средний балл каждого ученика, окончательную отметку, которую учитель может изменить на своё усмотрение.

Система тестирования INDIGO

1. Система имеет возможность создавать тесты для проверки знаний учащихся, а также для определения профессионального уровня сотрудников, проведения опросов, олимпиад, психологического тестирования.
2. Платформа:
3. Разделение на ролевые модули. Разделена на модули администратор и пользователь, где администратор создает и редактирует тесты, управляет базой тестов, пользователей и базой результатов, строит отчеты и анализ статистики.

Пользователь, в свою очередь, может регистрироваться, авторизоваться, проходить тесты и смотреть результаты тестирования

1. Достоинства у системы в целом следующие: имеет многоязычный веб-интерфейс и поддерживает символы всех языков. Также, центр тестирования можно развернуть на любом компьютере или на интернет-серверах Indigo. Недостатком может являться не совсем понятный интуитивно интерфейс, перед прохождением тестов, ученикам нужно объяснить - как, что и куда нажимать по окончании теста, а это занимает драгоценное время от учебных занятий.
   1. Постановка задачи и определение основных требований к разрабатываемой системе.
   2. Анализ существующих решений в данной области

Подводя итоги анализа существующих решений можно выделить главные достоинства систем в целом: это возможность простого скачивания в открытом доступе и автономность самой системы, поскольку без этого фактора будет достаточно затруднительно сделать систему простой и независимой в прохождении тестов. Еще одним плюсом будет являться небольшой размер программы и минимальная документация, что позволит устанавливать АС очень быстро, а пользоваться ею просто.

# ПРОЕКТИРОВАНИЕ СИСТЕМЫ ТЕСТИРОВАНИЯ

* 1. Выбор методологии
  2. Выбор инструментальных средств проектирования

Платформа .NET Framework

Платформа .NET Framework – это технология, поддерживающая создание и выполнение нового поколения приложения и веб-служб. Платформа состоит из общеязыковой среды выполнения (среды CLR или Common Language Runtime) и библиотеки классов .NET Framework.

Основа платформы .NET Framework – среда CLR. Ее можно считать агентом, управляющим кодом во время выполнения и представляющим основные службы, такие как управление памятью, потоками и удаленное взаимодействие.

Основная задача среды – это управление кодом. Код, обращающийся к среде выполнения, называют управляемым кодом, а код, который не обращается к среде выполнения, называют неуправляемым кодом. Библиотека классов является комплексной объектно-ориентированной коллекцией типов, применяемых для разработки приложений – начиная с консольных приложений, запускаемых из командной строки, и заканчивая приложениями, использующими последние технологические возможности ASP.NET, например Web.Forms и веб-службы XML. Также, с помощью .NET можно создавать и обычные Windows-приложения с интерфейсом пользователя (GUI).

Существуют библиотеки классов .NET Framework. Библиотека классов платформы представляет собой коллекцию типов, которые тесно интегрируются со средой CLR. Библиотека является объектно-ориентированной. Она представляет типы, из которых управляемый код пользователя может наследовать функции. Это упрощает работу с типами .NET, но и уменьшает время, затрачиваемое на изучение новых средств платформы .NET. В коллекциях .NET реализуется набор интерфейсов, которые можно использовать для разработки пользовательских классов коллекций, которые могут объединяться с классами .NET.

Основные функции .NET следующие:

- Богатая функциональность. Платформа .NET Framework предоставляет богатый набор функционала «из коробки». Она содержит сотни классов, которые предоставляют функциональность, готовую к использованию в приложениях. Это означает, что разработчику не нужно вникать в низкоуровневые детали различных операций, таких как ввод-вывод, сетевое взаимодействие и т.д.

- Простая разработка веб-приложений. ASP.NET – это технология, доступная на платформе .NET для разработки динамических веб-приложений. ASP.NET предоставляет управляемую событиями модель программирования. ASP.NET предоставляет различные элементы пользовательского интерфейса (таблицы, сетки, календари и т.д.), что существенно упрощает задачу программиста.

- Поддержка ООП. Платформа .NET предоставляет полностью объектно-ориентированное окружение. Даже примитивные типы вроде целых чисел и символов теперь считаются объектами.

- Поддержка многоязычности. Как правило, в больших компаниях есть программисты, пишущие на разных языках. Платформа .NET позволяет человеку писать на том языке, к которому он привык.

- Безопасность. Windows-платформа всегда подвергалась критике за плохую безопасность. Microsoft приложила огромные усилия и сделали платформу .NET безопасной для корпоративных приложений. Безопасность типов, безопасность доступа к коду, аутентификация на основе ролей – все это делает приложения надежными и безопасными.

Что представляет собой .NET

.net Framework представляет собой программную платформу для создания приложений на базе семейства операционных систем Windows, а также многочисленных операционных систем разработки не Microsoft, таких как Mac OS X и различные дистрибутивы Unix и Linux.

Основные функциональные возможности .NET:

- Поддержка разных языков программирования (C#, Visual Basic, F# и т.д.)

- Наличие общего исполняемого механизма, который используется всеми языками .NET. Благодаря этому есть хорошо определенный набор типов, которые способен понимать каждый поддерживающий .NET язык.

- Тотальная интеграция языков. Поддерживается межъязыковые наследование, обработка исключений, отладка кода.

- Огромная библиотека базовых классов. Она позволяет упростить прямые вызовы к API-интерфейсу и предлагает согласованную объектную модель, которую могут использовать все языки .NET.

Основные компоненты .NET. Три кита: CLR, CTS, CLS

С точки зрения программиста, .NET представляет собой исполняющую среду и обширную библиотеку базовых классов. Уровень исполняющей среду называется общеязыковой исполняющей средой (Common Language Runtime) или средой CLR.

Основная задача CLR – автоматическое обнаружение, загрузка и управление типами .NET. Типами управляет .NET, а не программист. Также, среда заботится о низкоуровневых деталях – управление памятью, обработка потоков, выполнение разных проверок, связанных с безопасностью.

Другой компонент .NET – общая система типов (Common Type System) или система CTS. Предоставляет полное описание всех возможных типов данных и программных конструкций, которые поддерживаются исполняющей средой, а также способов, как все эти сущности могут взаимодействовать друг с другом. Любая возможность CTS может не поддерживаться в отдельно взятом языке, совместимом с .NET.

Именно поэтому существует третий компонент – CLS (Common Language Specification) или спецификация CLS. В ней описано лишь то подмножество общих типов и программных конструкций, каковое способны воспринимать все .NET языки. Следовательно, если использовать типы .NET только с функциональными возможностями, предусмотренными в CLS, то и все .NET совместимые языки могут их использовать. Отдельно взятому компилятору C# можно указать, чтобы он проверял весь код на предмет совместимости с CLS.

Библиотека базовых классов.

Кроме трёх китов, в составе платформы .NET существует библиотека базовых классов. Она доступна для всех языков, поддерживающих .NET.В этой библиотеке содержатся определения примитивов (потоки, системы графической визуализации, механизмы для взаимодействия с разными внешними устройствами), предоставляет поддержку целого ряда служб, которые нужны в большинстве реальных приложений.

Язык C#

Поскольку платформа .NET радикально отличается от предыдущих технологий Microsoft, корпорация разработала специально для неё новый язык программирования – C# (Си Шарп). Синтаксис этого языка похож на Java. Поскольку этот язык считается гибридным, он взял лучшее из нескольких языков программирования и является простым, мощным и гибким языком.

Главные достоинства языка

- C# спроектирован быть кроссплатформенным, однако его развитие не пошло в этом направлении. Поэтому под Windows образовалась достаточно полная .NET инфраструктура, на других же платформах равноценной инфраструктуры не появилось.

- Говоря о совокупности «простоты разработки», «красоты кода» и производительности, то используя C# очень просто написать код, удовлетворяющий этим критериям одновременно.

- Огромное количество библиотек с .NET идет в базе, плюс к ним множество свободно доступных библиотек, это покрывает практически все первостепенные задачи разработки под Windows. Наличие большого количества стандартных типов почти избавляет от библиотек, где базовые типы переопределены. И в силу того, что библиотеки С# сравнительно молодые,- интерфейсы библиотек, как правило, лучше вписываются в те или иные шаблоны проектирования, что часто упрощает их изучение.

- Код на С#, как правило, выглядит проще и лаконичнее, чем код, например С++. Используя С#, меньше шансов допустить ошибку в сложном коде и больше шансов написать чистый код, обладая теми же ресурсами. Это может быть полезно при решении сложных задач.

Интегрированная среда разработки

Microsoft Visual Studio - линейка продуктов компании Microsoft, включающих интегрированную среду разработки программного обеспечения и ряд других инструментальных средств. Данные продукты позволяют разрабатывать как консольные приложения, так и приложения с графическим интерфейсом, в том числе с поддержкой технологии Windows Forms, а также веб-сайты, веб-приложения, веб-службы как в родном, так и в управляемом кодах для всех платформ, поддерживаемых Windows, Windows Mobile, Windows CE, .NET Framework, Xbox, Windows Phone .NET Compact Framework и Silverlight. По умолчанию Visual Studio обеспечивает поддержку C#, C и C++, JavaScript, F# и Visual Basic. Приложения Windows можно создавать с помощью любого языка программирования, совместимого с .NET Framework (Visual Basic, C#, C++ и многие другие) и средств отладки .NET Framework.

Система управления базами данных (СУБД)

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

Главные достоинства СУБД MySQL

1. Быстродействие. Благодаря внутреннему механизму многопоточности, MySQL имеет очень высокое быстродействие.
2. Безопасность. Довольно высокий уровень безопасности обеспечивается благодаря базе данных mysql, создающейся при установке пакета и содержащей пять таблиц. При помощи этих таблиц можно описать, какой пользователь из какого домена с какой таблицей может работать и какие команды он может применять. Пароли, хранящиеся в базе данных, можно зашифровать при помощи встроенной в MySQL функции password().
3. Открытость кода. Благодаря этому пользователь может сам добавлять в пакет нужные функции, расширяя его функциональность так, как ему требуется.
4. Ресурсы. Для того, чтобы использовать эту СУБД, не понадобится очень мощный компьютер.
5. Кроссплатформенность. В настоящее время существуют версии программы для большинства распространенных компьютерных платформ. Пользователь может сам выбрать, с чем работать, например с Linux или Windows, в случае замены ОС он не потеряет свои данные и ему не придется устанавливать дополнительные инструменты для их переноса.
   1. Проектирование системной архитектуры
   2. Структура базы данных

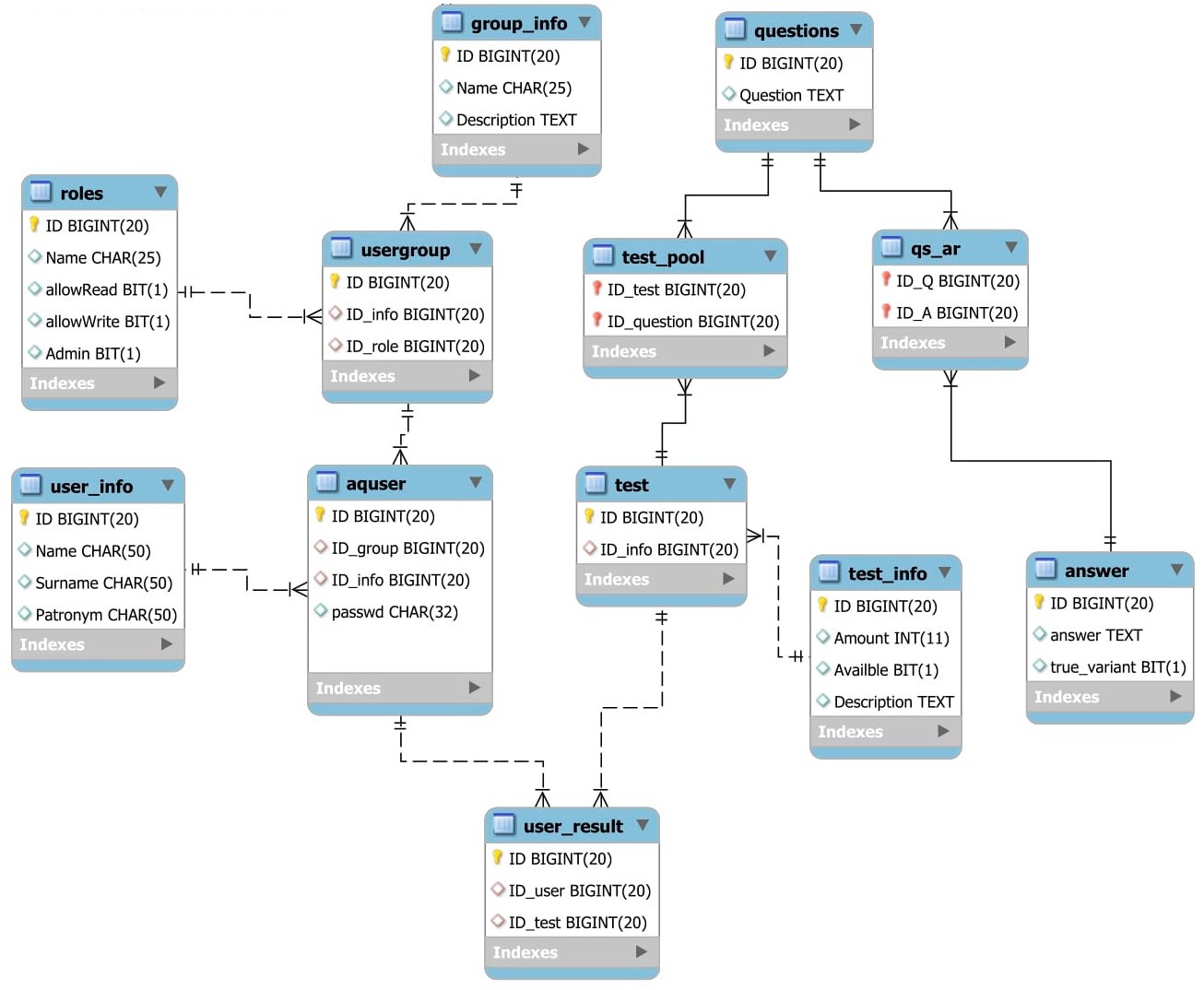


Рис.1. ER-диаграмма базы данных

Поле ID в каждой таблице является системным полем, уникальным для каждой записи в пределах одной таблицы.

Таблица **group\_info** содержит поля имени и описания группы пользователей.

Таблица **roles** состоит из полей: *Name* – имя уровня доступа, *allowRead* – разрешение на чтение, *allowWrite* – разрешение на запись и *Admin* – полный доступ к чтению и записи.

Таблица **usergroup** имеет два внешних ключа на **group\_info** и **roles**, она необходима для хранения указателей на роли. Группа этих трёх таблиц представляет собой ролевую модель.

Таблица **user\_info** содержит поля *Name*, *Surname*, *Patronym* – фамилию, имя и отчество соответственно. Она нужна для хранения информации.

Таблица **aquser** имеет два внешних ключа на **user\_info** и **usergroup**, также содержит поле пароля.

Таблица **questions** содержит поле *Question*, где хранится текст вопроса, который вводит преподаватель во время создания теста.

Таблица **answer** содержит поле ответа и бита, который указывает на то, правильный ли ответ. 0 – неправильный, 1 – правильный.

Таблица **qs\_ar** является таблицей связей вопросов и ответов.

Таблица **test** хранит внешний ключ на информацию о тесте в таблице **test\_info**.

Таблица **test\_pool** является таблицей связей вопросов и тестов. (Несколько тестов к нескольким вопросам и несколько вопросов в разных тестах)

Таблица **test\_info** содержит информацию о количестве вопросов в тесте, о доступности теста для данного пользователя и описание теста.

# РАЗРАБОТКА АВТОМАТИЗИРОВАННОЙ СИСТЕМЫ ТЕСТИРОВАНИЯ

* 1. Выбор средств реализации системы
  2. Разработка пользовательского интерфейса
  3. Контрольный пример