**Министерство образования и науки Российской Федерации**

**Федеральное государственное бюджетное образовательное**

**учреждение высшего образования**

**«Московский государственный университет технологий и управления**

**имени К.Г. Разумовского (Первый казачий университет)»**

**Университетский колледж информационных технологий**

**ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА**

**(ДИПЛОМНЫЙ ПРОЕКТ)**

**тема: Разработка прикладного приложения «Консультант скейтбордиста»**

**Студент:** \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Строков С.А.

(подпись студента) (Ф.И.О. студента)

**Группа:** **П-403**

**Специальность: 09.02.03 «Программирование в компьютерных**

**системах»**

**Руководитель:** \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Маликов Е.В.

(подпись руководителя) (Ф.И.О.руководителя)

**Консультанты:**

**Экономическая часть** \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Блесткина О.В.

(подпись консультанта) (Ф.И.О.консультанта)

**Нормоконтроль** \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Гусева Е.Л.

(подпись консультанта) (Ф.И.О.консультанта)

Москва

2017

**СОДЕРЖАНИЕ**

Изм.

Лист

№ докум.

Подпись

Дата

Лист

2

ФГБОУ ВО МГУТУ им. К Г Разумовского (ПКУ) -09.02.03.26 2017 ДП

Разраб.

Строков С.А.

Проверил

Маликов Е.А.

Эконом.

Блесткина О.В.

Н.контр

Гусева Е.Л.

Разработка прикладного

Приложения «консультант скейтбордиста»

Лит.

Листов

79

*УниКИТ П-403*

[ВВЕДЕНИЕ 4](#_Toc485176081)

[1 ТЕОРИТИЧЕСКАЯ ЧАСТЬ 7](#_Toc485176082)

[1.1 Анализ предметной области 7](#_Toc485176083)

[1.2 Анализ существующих разработок 8](#_Toc485176084)

[1.3 Выбор инструментальных средств 18](#_Toc485176085)

[1.3.1 Этап проектирования 18](#_Toc485176086)

[1.3.2 Реализация базы данных 18](#_Toc485176087)

[1.3.3 Заполнение базы данных 19](#_Toc485176088)

[1.3.4 Разработка приложения 19](#_Toc485176089)

[1.3.5 Графическое оформление приложения 21](#_Toc485176090)

[2 ПРАКТИЧЕСКАЯ ЧАСТЬ 22](#_Toc485176091)

[2.1 Реализация ресурса 22](#_Toc485176092)

[2.1.1 Этап проектирования 22](#_Toc485176093)

[2.1.2 Реализация базы данных 23](#_Toc485176094)

[2.1.3 Заполнение базы данных 31](#_Toc485176095)

[2.1.4 Разработка приложения 32](#_Toc485176096)

[2.1.5 Интерфейс приложения 36](#_Toc485176097)

[2.2 Тестирование 39](#_Toc485176098)

[3 ЭКОНОМИЧЕСКАЯ ЧАСТЬ 42](#_Toc485176099)

[3.1 Расчет затрат на разработку программного продукта 42](#_Toc485176100)

[3.2 Расчет затрат на внедрение программного продукта 46](#_Toc485176101)

[3.3 Расчет текущих затрат по программному продукту 48](#_Toc485176102)



[3.4 Расчет экономической целесообразности разработки и внедрения информационных технологий 50](#_Toc485176103)

3

*Лист*

Изм.

Лист

№ докум.

Подпись

Дата

ФГБОУ ВО «МГУТУ им. К.Г. Разумовского (ПКУ)» 09.02.03.09 2017ДП

[4 ОХРАНА ТРУДА И ЖИЗНЕДЕЯТЕЛЬНОСТИ 54](#_Toc485176104)

[4.1 Общие требования безопасности 55](#_Toc485176105)

[4.2 Требования безопасности перед началом работы 56](#_Toc485176106)

[4.3 Требования безопасности во время работы 56](#_Toc485176107)

[4.4 Техника безопасности в аварийной ситуации 57](#_Toc485176108)

[ЗАКЛЮЧЕНИЕ 58](#_Toc485176110)

[СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ 60](#_Toc485176111)

[ПРИЛОЖЕНИЕ А Иллюстрации к таблице 1 62](#_Toc485176112)

[ПРИЛОЖЕНИЕ Б Листинг программы 65](#_Toc485176113)

**ВВЕДЕНИЕ**

В настоящее время получило широкое распространение такое течение, как «скейтбординг» и с каждым годом все больше и больше как подростков, так и взрослых встают в их ряды.

Скейтбординг — это спорт современных людей, чьей неотъемлемой частью жизни являются информационные технологии. Живя в век высоких технологий людям стали приходить на помощь специальные программы-помощники, которые являются высокотехнологичными системами, включающими в себя множество элементов.

Прогресс не стоит на месте и каждый год можно наблюдать все новые и новые виды досок, что вводит новичков в растерянность при выборе своего первого скейтборда. Технологии не обошли стороной и эту область – теперь люди идут выбирать скейты в специализированные интернет-магазины, обращаясь за помощью к онлайн-помощникам.

Актуальность выбранной темы обусловлена тем, что спрос на программы-помощники как на рынке информационных технологий, так и в повседневной жизни неуклонно растет. Так как большинство людей в наше время привыкли обращаться за помощью к различным компьютерным программам, то и необходимость в создании подобного рода продукта стоит остро, особенно связанного с таким прогрессивным и современным течением, как скейтбординг.

Актуальность создания данного приложения заключается в том, что на данный момент не существует специальных программ-помощников для выбора скейтборда, поэтому ставится цель – создать актуальное на сегодняшний день программное решение, предназначенное для людей, желающих стать скейтбордистами. Программа должна не только помогать с выбором, но и снабжать актуальной информацией о различных моделях. Так пользователи смогут получить достаточно информации для ориентирования в предметной области.

Хочется отметить, что данная тема актуальна и для специализированных магазинов, где продаются скейтборды. Порой покупатели в растерянности от разнообразия досок и не могут остановиться на чем-то конкретном, а консультантам трудно объяснить все тонкости каждой отдельной модели, не прибегая к терминологии. Если пользователь воспользуеться данным приложением прежде чем обратиться за помощью к консультанту, то это даст покупателю готовую подборку подходящих ему моделей и необходимую информационную базу, что сильно облегчит и сократит время покупки.

Целью выпускной квалификационной работы является разработка прикладного приложения «консультант скейтбордиста», изучение принципов работы систем поддержки принятия решений и их реализации, закрепление знаний работы с базами данных, освоение новых методов разработки систем.

В соответствии с поставленной целью необходимо решить следующие задачи:

* Изучить и составить описание параметров основных видов скейтбордов;
* Разработать базу данных для хранения информации о скейтбордах;
* Разработать ветку вопросов и ответов и внести все взаимоотношения в таблицы в базе данных;
* Разработать приложение, разграничивающееся на часть пользователя и администратора;
* Создать запросы на выборку для каждого вида скейтбордов;
* Разработать интуитивно понятный дизайн приложения;
* Проанализировать имеющиеся разработки в этой области.

Объектом исследования будут являться компьютерные автоматизированные системы, так как приложение напрямую связано с данным видом информационных систем.

Предметом исследования в свою очередь являются системы поддержки принятия решений.

Работа состоит из введения, четырех частей, заключения, списка использованной литературы, приложений, программы и базы данных.

1. **ТЕОРИТИЧЕСКАЯ ЧАСТЬ**
   1. **Анализ предметной области**

Данную программу можно выделить в класс программ-помощников. Программы такого рода служат для частичной замены экспертов-специалистов конкретной области и помогают пользователю со сложившейся проблемой.

Система поддержки решений СППР решает две основные задачи:

* Выбор наилучшего решения из множества возможных (оптимизация);
* Упорядочение возможных решений по предпочтительности (ранжирование).

Рассмотрим структуру СППР:

* Информационные хранилища данных;
* Средства и методы извлечения, обработки и загрузки данных;
* Многомерная база данных и средства анализа;
* Средства Data Mining.

По взаимодействию с пользователем выделяют три вида СППР:

* Пассивные помогают в процессе принятия решений, но не могут выдвинуть конкретного предложения;
* Активные непосредственно участвуют в разработке правильного решения;
* Кооперативные предполагают взаимодействие СППР с пользователем. Выдвинутое системой предложение пользователь может доработать, усовершенствовать, а затем отправить обратно в систему для проверки. После этого предложение вновь представляется пользователю, и так до тех пор, пока он не одобрит решение.

В рассматриваемой теме дипломной работы подобный помощник-консультант реализован в качестве прикладного приложения с серверной частью. В зависимости от выбираемого пользователем варианта отсылается запрос на серверную часть и по установленным правилам приходят результаты на клиентскую часть.

Приложение «консультант скейтбордиста», является прикладной СППР, так как служит для анализа небольших объемов данных, и предназначена для одновременной работы одного пользователя. Так же это активная СППР, так как результатом работы программы является выдача наилучших вариантов из всех существующих.

* 1. **Анализ существующих разработок**

На сегодняшний день это является уникальной разработкой, так как прямых аналогов не существует, но имеются интернет-магазины, обладающие частичным функционалом (таблица 1).

|  |  |  |  |
| --- | --- | --- | --- |
| Таблица 1 – Cравнительный анализ разработок. | | | |
| **Критерии** | **Allaboardshop** | **X3mboardshop** | **«Консультант скейтбордиста»** |
| Вид приложения | Интернет - магазин и школа  по сноубордам и лонгбордам | Спортивный интернет-магазин | Прикладное приложение |
| Целевая аудитория | Клиенты, желающие приобрести экипировку для катания на скейтбордах и сноубордах, сами доски и комплектующие и аксессуары, а так же желающие записаться на обучение с инструктором по данным дисциплинам. | Клиенты, которые подбирают скейтборды, лонгборды, сноуборды, лыжи, специализированную экипировку и аксессуары. Так же предоставляет спонсорскую поддержку молодым райдерам. | Частные пользователи или клиенты специализированных спортивных магазинов, которые желают подобрать себе скейтборд или лонгборд, а также узнать подробную информацию о каждом. |
| Способ оказания помощи в | На сайте реализован фильтр товаров | Так же реализован фильтр товаров | Алгоритм подбора подходящих вариантов, через |
| подборе скейтборда |  |  | систему помощи принятия решений |
| Продолжение таблицы 1 | | | |
| **Критерии** | **Allaboardshop** | **X3mboardshop** | **«Консультант скейтбордиста»** |
| Количество критериев, учитываемых при отборе | У фильтра стоит 6 критериев (рис. 1) | У фильтра 6 групп критериев, в каждой группе доступно несколько параметров с возможностью множественного выбора (рис. 4) | Пользователь отвечает на вопросы одним из трех предложенных вариантов, и в зависимости от варианта ответа ветка вопросов может либо дальше развиваться, либо предоставить готовый результат |
| Скорость выдачи результатов пользователю | Результаты выдаются сразу по окончанию заполнения пользователем фильтра | После выбора каждого нового критерия сортировки происходит автоматический подбор досок по параметрам, из-за чего сайт переходит в режим ожидания | Когда пользователь ответит на все предложенные вопросы – выдаются результаты |
| Способ выдачи результатов | Фильтр выдает ряд скейтбордов, заключенных в блоки (рис. 2) | Выводится отсортированный список скейтбордов, к тому же можно | Результаты выдаются списком по три элемента на страницу |
| Продолжение таблицы 1 | | | |
| **Критерии** | **Allaboardshop** | **X3mboardshop** | **«Консультант скейтбордиста»** |
|  |  | указывать сколько досок выводить сразу на странице и по каким критериям производить сортировку результатов (рис. 5) |  |
| Просмотр одного выбранного варианта | После нажатия правой кнопкой мыши на блок раскрывается подробная информация о выбранном варианте | Если кликнуть правой кнопкой по одному из предложенных вариантов, то откроется экран с доской | Чтобы рассмотреть один вариант – достаточно нажать на кнопку «подробнее», находящейся в нижней части блока |
| Описание  Одной модели скейтборда | В данном окне можно рассмотреть подробную информацию о скейтборде, включая описание, основные | Здесь очень подробно выведена вся необходимая информация о скейтборде и его параметрах. Это оптимальное количество | В появившемся окне выводится вся доступная в базе данных информация о конкретном скейтборде, его характеристики, описание и компоненты. Эти сведения являются достаточными для |
| Продолжение таблицы 1 | | | |
| **Критерии** | **Allaboardshop** | **X3mboardshop** | **«Консультант скейтбордиста»** |
|  | характеристики, стиль катания и отзывы, оставленные покупателями этой модели. Так же можно заметить те критерии, по которым проводилась фильтрация (рис. 3) | сведений, в котором новичку будет трудно запутаться, но и достаточное, чтобы определиться с выбором (рис. 6) | пользователя, чтобы появилось понимание основ скейтбординга и было проще определиться с выбором. |
| Различия способов фильтрации | Тут используется обычная выборка, по немногочисленным параметрам, используется всего один пункт, где можно выбрать сразу несколько вариантов | В данном случае используется сложная множественная выборка, здесь в каждом блоке доступны для пользователя выбор от одного до всех возможных вариантов. | Алгоритм на каждом этапе добавляет, либо наоборот отсекает множество вариантов из всех доступных вариантов, так что дойдя до последнего вопроса будет сделана точная выборка по пожеланиям пользователя |
|  |  |  |  |
| Продолжение таблицы 1 | | | |
| **Критерии** | **Allaboardshop** | **X3mboardshop** | **«Консультант скейтбордиста»** |
| Недостатки данного способа фильтрации | В данном случае разработчики отталкивались от комплекции покупателей, а не делали упор на комплектующие доски, это в итоге привело к тому, что | После выбора каждого нового критерия сортировки происходит автоматический подбор досок по параметрам, из-за чего сайт переходит в | Пользователь не знает точно по каким критериям идет фильтрация, а видит только уже результат. |
|  | единственный важный критерий сортировки – стиль катания, ибо по всем остальным критериям будет выдавать идентичные результаты | режим ожидания, это некомфортно пользователю, поэтому лучше дать сперва выбрать все критерии, а потом по нажатию на кнопку проводить фильтрацию. |  |
| Интуитивность интерфейса | Сам фильтр находится не на стартовой странице, чтобы перейти на него – необходимо | На данном сайте фильтр присутствует на каждой странице, где имеются скейтборды. Он | На начальном экране сразу предлагают начать и нажать на кнопку «старт», далее предлагается ответить на определенное количество |
| Продолжение таблицы 1 | | | |
| **Критерии** | **Allaboardshop** | **X3mboardshop** | **«Консультант скейтбордиста»** |
|  | кликнуть на закладку в верхней части панели. Сам фильтр реализован до банального понятно- пользователю лишь надо кликнуть по подходящему варианту (или нескольким), а в конце подтвердить, нажав на кнопку «Подобрать борд» | реализован на боковой панели, в виде выпадающих блоков-групп с возможностью выбора многих вариантов, после выбора одного критерия сразу выводятся отобранные варианты | вопросов, после чего выводятся результаты фильтрации, где пользователь может ознакомиться с каждым отдельно. |

Рассмотрим пункт «Вид приложения». Можно заметить, что все аналоги представлены в виде веб-приложений, а именно - интернет-магазинов. Это является лучшим способом помочь в выборе товара, имеющегося в магазине, но когда встает необходимость лишь разобраться в предметной области, то ценники и другая излишняя информация сбивают с толку. К тому же, если отсутствует подключение к сети Интернет, то доступ к ресурсам веб-приложений теряется, а прикладное приложение всегда можно запустить, единожды установив. Поэтому стоит ставить именное такой вид программ на терминалы в специализированных магазинах, чтобы клиенты всегда могли получить оперативную помощь в подборе скейтбордов.

Следующий пункт «Целевая аудитория». К первым двум ресурсам обращаются клиенты с целью что-то купить или получить услугу, предоставляемую организацией. Следовательно, упор делается именно на покупателей и во многом весь функционал несет цель продать товар или услугу. Приложение «консультант скейтбордиста», в свою очередь, является приложением-помощником, и пользователи получают оперативные советы по выбору скейтборда, именно то, в чем и заключался основной смысл программы.

«Способ оказания помощи в подборе скейтборда» в данном пункте рассматриваются программные решения в каждом приложении. В онлайн-магазинах реализованы фильтры товаров, что является стандартным и практичным решением для большинства подобных веб-приложений. При создании программы «консультант скейтбордиста» была разработана система вопросов-ответов, где после каждого выбранного варианта идет фильтрация из общего количества скейтбордов по запросам к серверу. Данный способ позволяет быстро и достаточно точно определить, что необходимо пользователю, ведь один вопрос может затрагивать сразу несколько критериев отбора.

Рассматривая пункт «количество критериев, учитываемых при отборе», обратим внимание, что пускай на обоих онлайн-магазинах и реализованы фильтры, но подход совсем разный – на Allaboardshop учитываются всего 6 критериев, которых явно недостаточно для точного подбора скейтборда. На X3mboardshop уже используется сложный фильтр, где все критерии разбиты по одной тематике на 6 групп, но в данном случае пользователю легко запутаться в большом количестве терминологии и потратить гораздо больше времени. Приложение «консультант скейтбордиста» свело подбор доски к автоматизму и не требует от пользователя широких познаний в предметной области, требуя от него только выбирать один из предложенных вариантов. Всего в базе данных предусмотрено 14 вопросов и 42 ответа, которые перебирают все возможные комбинации и отбирают то, что наиболее подходит пользователю.

«Скорость выдачи результатов пользователю» - в данном пункте идет речь о том, когда пользователь сможет увидеть результат фильтрации. После сравнения всех трех способов, можно сделать вывод, что на X3mboardshop выбранный способ является не удачным, ибо в среднем ожидание длится около пяти секунд, а, учитывая, что критериев 6 групп и в каждой около десяти вариантов, то ожидание становится раздражающим фактором. Поэтому в «консультанте скейтбордиста» алгоритм вывода результата активируется после прохождения всех вопросов.

Следующим пунктом является «Способ выдачи результатов». Здесь рассматривается в каком виде пользователю предоставляется выводимые результаты фильтрации. Основным отличаем в этом пункте приложения «консультант скейтбордиста» от веб-приложений является вывод группами по три варианта, а не целым списком. Плюсом данного способа является возможность более детально изучить каждый предложенный вариант, заставляя задержаться на каждом более внимательно, а не пролистывать список сразу десятками. Такой способ подходит именно такому роду приложений, где вариантов не столь много и можно просмотреть каждый.

Далее идут два пункта «Просмотр одного выбранного варианта» и «Описание одной модели скейтборда», тут расписано как пользователь может ознакомиться с одним предложенным вариантом. Особых различий во всех трех рассматриваемых приложений нет, за исключением Allaboardshop. В данном онлайн-магазине слишком много лишней информации, которая не является ключевой при выборе скейтборда и лишь затрудняет понимание пользователя.

В следующих пунктах «Различия способов фильтрации» и «Недостатки данного способа фильтрации» разбираются основные отличительные особенности каждого подхода. После анализа, можно сделать вывод, что в приложении «консультант скейтбордиста» реализован наиболее совершенный алгоритм, так как в сравнении с Allaboarshop совершает гораздо более сложную выборку по большему количеству критериев, а по сравнению с X3mboardshop скорость выборки и практичность несколько выше.

После анализа пункта «Интуитивность интерфейса» приходим к выводу, что прикладное приложение удобнее использовать, так как все элементы на форме легко отличимые и пользователю интуитивно понятно какую функцию выполняет тот или иной компонент. В веб-приложении страницы сильно перегружены и часто приходится «гулять» по ним, пока не найдется нужное.

1. **Выбор инструментальных средств**

Работа с базой данных состоит из нескольких основных этапов и для каждого используется свое инструментальное средство:

* + 1. **Этап проектирования**

На этапе проектирования выбор пал на Sybase PowerDesigner. Это мощный и современный комплекс с возможностью концептуального и физического проектирования баз данных с и редактированием UML диаграмм. Можно выделить особенности, отличающие его от аналогового ряда:

* Наличие единого репозитория, с возможностью работы сразу нескольких специалистов над одной моделью;
* Возможность устанавливать только необходимые модули программы, предназначенные для конкретной задачи;
* Улучшенное управление моделями, включая синхронизацию объектов, моделей и баз данных.

Для целей дипломного проекта использовалась Trial версия программы с установленными модулями :

* PhysicalArchitect (PDM) – обладает всей необходимой конфигурацией для создания физической модели базы данных, а так же генерирования Sql-кода.
* ObjectArchitect (PDM, CDM, OOM) – предназначено для визуального моделирования баз данных, разработчикам доступно 9 видов UMLдиаграмм.
  + 1. **Реализация базы данных**

После проектирования базы данных – необходимо составить скрипт создания на языке Transact-SQL и выполнить его в СУБД Microsoft Sql Server R8. Выбор данного программного продукта обусловлен широким спектром возможностей, которые необходимы для реализации выбранной темы. Удобный и интуитивно понятный графический интерфейс данной СУБД позволяет быстро переключаться между доступным функционалом и не затормаживать процесс разработки.

Между Sybase PowerDesigner и Microsoft Sql Server R8 просто установить связь и в случае, если понадобится оперативно изменить физическую модель базы данных, то такая возможность предусмотрена, без потери уже внесенных данных.

Очень удобная внутренняя файловая система, благодаря которой всегда можно быстро найти сохраненные хранимые процедуры, функции и запросы. Так же наличие встроенного динамического помощника очень упрощает написание запросов, посредством появляющегося списка объектов.

Еще одним большим плюсом является возможность изменения в структуре данных и создание резервных копий базы данных прямо во время работы сервера. А разграничение прав доступа, организованное на уровне пользователя, не даст ему совершить несанкционированные действия, поскольку есть возможность создать пользовательские привилегии.

* + 1. **Заполнение базы данных**

В ходе разработки системы необходимо было продумать дерево вопросов-ответов и его визуализировать. Самым удобным инструментальным средством, посредством чего создавались диаграммы является «Mind Manager».

Благодаря обширному функционалу данного продукта, не составляет большого труда составить необходимую схему, провести связи и сделать пометки. Имеется интеграция в более чем 700 других приложений, сохранение диаграмм во многих форматах, а также доступно множество инструментов для бизнес-планирования, анализа рисков и соблюдения крайних сроков.

В составленной диаграмме интуитивно понятная структура вопросов-ответов, благодаря чему не возникает труда заполнить саму базу данных.

* + 1. **Разработка приложения**

При разработке приложения выбор пал на среду разработки Lazarus - Delphi совместимый кроссплатформенной IDE. Языком программирования является Free Pascal, который используется специально для работы с синтаксисом Delphi, построенном на ООП. Lazarus является открытым программным продуктом (open source) и любой пользователь может создать свой модуль, а так же использовать уже имеющийся в свободном доступе. Особенностью Lazarus является совместимость продукта со многими платформами, например, создавая приложение под Windows, может возникнуть необходимость перенести приложение на Linux, но никаких трудностей не возникнет, ведь встроенное расширение упрощает данный процесс, не прибегая к написанию дополнительных строк кода.

Наличие большого количества компонентов позволяют создавать приложения, предназначенные для самых разнообразных областей. В целях дипломного проекта использовались следующие компоненты:

* Tpanel – это основной класс, предназначенный для хранения других компонентов в себе;
* Timage – компонент, предназначенный для вывода графического изображения на форму;
* TGroupbox – предназначен для группировки схожих компонентов;
* TLabel – компонент для вывода текста, который пользователь не может изменить в ходе работы программы;
* TEdit – компонент для ручного ввода данных пользователем;
* TDBtext – предназначен для вывода текста из присоединенной к программе базы данных;
* TSqlQuery – служит для работы с Sql запросами, выдает данные в виде таблицы;
* TSqlTransaction – предназначен для транзакции с базой данных;
* MSSqlConnection – компонент для присоединения к базе данных MsSql server и поддержание соединения в ходе работы приложения.

Lazarus оснащен очень удобным и простым осуществлением перехода с формы на форму, что сильно ускоряет процесс разработки.

При написании кода очень помогает встроенные функции рефакторинга и автоформатировния.

Основным критерием, повлиявшим на выбор в качестве среды разработки Lararus – стала кроссплатформенность, так как приложение «консультант скейтбордиста» может использоваться пользователями разных операционных систем.

* + 1. **Графическое оформление приложения**

Каждому пользователю приятно использовать приложение не только с интуитивным графическим интерфейсом, но и визуально проработанным. В качестве графического редактора использовался продукт компании Adobe Systems, а именно - Photoshop CS6. Это на данный момент самый популярный и совершенный инструмент для работы с растровыми изображениями.

При разработке интерфейса Photoshop CS6 использовался для создания изображения, подходящего под размеры формы программы. В ходе разработки использовались различные встроенные инструментальные средства, позволяющие как редактировать изображение, так и создавать что-то новое. Например, самым используемым инструментом была «кисть», благодаря которой создавались основные элементы интерфейса.

После того, как все необходимые элементы интерфейса были созданы – изображение сохраняется в формате Png, поддерживаемый средой разработки Lazarus, и уже непосредственно используется для оформления приложения.

# ПРАКТИЧЕСКАЯ ЧАСТЬ

* 1. **Реализация ресурса**

Рассмотрим процесс разработки приложения поэтапно:

**2.1.1 Этап проектирования**

На этапе проектирования была разработана структура базы данных. Итогом этого этапа стали две XML-схемы, описывающие основные таблицы и связи между ними. Физическая модель базы данных, на которой идет полное описание скейтборда, его характеристик и составляющих (рисунок 7).

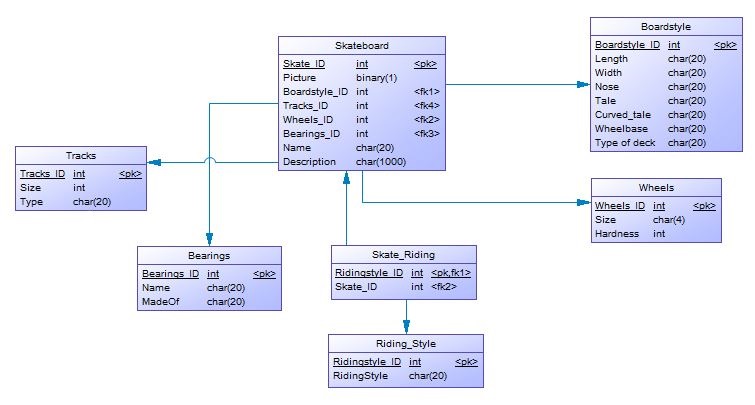


Рисунок 7 – Модель таблиц скейтборда

Вторая физическая модель описывает структуру, предназначенную для хранения вопросов и ответов, а также все взаимосвязи между ними (рисунок 8).

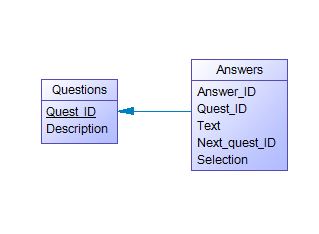


Рисунок 8 – Модель структуры вопросов-ответов

* + 1. **Реализация базы данных**

Рассмотрим каждую таблицу, ее столбцы и предназначение.

**Таблица Skateboard**

Данная таблица является составной и содержит в себе полную информацию об одной модели скейтборда.

|  |  |  |  |
| --- | --- | --- | --- |
| Таблица 2 – Описание таблицы Skateboard | | | |
| Название | Описание | Тип данных | Дополнительно |
| Skate\_ID | Уникальный идентификатор скейтборда. | int | Первичный ключ |
| Picture | Содержит в себе путь до изображения на диске | Nvarchar(max) |  |
| Boardstyle\_ID | Уникальный идентификатор вида деки. | Int | Внешний ключ |
| Tracks\_ID | Уникальный идентификатор подвесок. | Int | Внешний ключ |
|  | | | |
| Продолжение таблицы 2 | | | |
| Название | Описание | Тип данных | Дополнительно |
| Wheels\_ID | Уникальный идентификатор колес. | Int | Внешний ключ |
| Bearings\_ID | Уникальный идентификатор подшипников. | Int | Внешний ключ |
| Name | Название скейтборда | Char(20) |  |
| Description | Описание скейтборда | Char(1000) |  |

**Таблица Riding\_Style**

Данная таблица содержит все виды стилей катания на скейтборде.

|  |  |  |  |
| --- | --- | --- | --- |
| Таблица 3 – Описание таблицы Riding\_Style | | | |
| Название | Описание | Тип данных | Дополнительно |
| Ridingstyle\_ID | Уникальный идентификатор стиля катания. | Char(10) | Первичный ключ |
| RidingStyle | Название стиля катания | char(20) |  |

**Таблица Boardstyle**

Данная таблица содержит в себе характеристики деки.

|  |  |  |  |
| --- | --- | --- | --- |
| Таблица 4 – Описание таблицы Boardstyle | | | |
| Название | Описание | Тип данных | Дополнительно |
| Boardstyle\_ID | Уникальный идентификатор деки. | Int | Первичный ключ |
| Length | Параметр, отвечающий за длину. | Char(20) |  |
| Nose | Наличие у доски переднего конкейва. | Char(20) |  |
| Tale | Наличие у доски заднего конкейва. | Char(20) |  |
| Curved\_tale | Определяет загнут ли задний конкейв у деки. | Char(20) |  |
| Wheelbase | Расстояние между задними траками и передними | Char(20) |  |
| Type\_of\_deck | Тип деки | Char(20) |  |
| Description | Описание деки | varchar(max) |  |

**Таблица Bearings**

Данная таблица описывает виды подшипников.

|  |  |  |  |
| --- | --- | --- | --- |
| Таблица 5 – Описание таблицы Bearings | | | |
| Название | Описание | Тип данных | Дополнительно |
| Bearings\_ID | Уникальный идентификатор подшипников. | int | Первичный ключ |
| Name | Название подшипников | char(20) |  |
| MadeOf | Материал изготовления |  |  |

**Таблица Skate\_Riding**

Данная таблица служит для присвоения сразу нескольких стилей катания одной доске.

|  |  |  |  |
| --- | --- | --- | --- |
| Таблица 6 – Описание таблицы Skate\_Riding | | | |
| Название | Описание | Тип данных | Дополнительно |
| Ridingstyle\_ID | Уникальный идентификатор стиля катания. | char(20) | Внешний ключ |
| Skate\_ID | Уникальный идентификатор скейтборда. | int | Внешний ключ |

**Таблица Tracks**

Данная таблица служит для хранения информации о подвесках.

|  |  |  |  |
| --- | --- | --- | --- |
| Таблица 7 – Описание таблицы Tracks | | | |
| Название | Описание | Тип данных | Дополнительно |
| Tracks\_ID | Уникальный идентификатор подвесок. | int | Первичный ключ |
| Size | Параметр, определяющий размер подвесок | int |  |
| Type | Тип строения подвесок | Char(20) |  |
| Description | Описание модели подвесок | Varchar(max) |  |

**Таблица Wheels**

Данная таблица служит для хранения информации о колесах.

|  |  |  |  |
| --- | --- | --- | --- |
| Таблица 8 – Описание таблицы Wheels | | | |
| Название | Описание | Тип данных | Дополнительно |
| Wheels\_ID | Уникальный идентификатор колес. | int | Первичный ключ |
| Size | Параметр, определяющий размер колес | Char(4) |  |
| Hardness | Жесткость колес | int |  |
| Description | Описание модели колес | Varchar(max) |  |

**Таблица Questions**

Данная таблица служит для описания вопросов.

|  |  |  |  |
| --- | --- | --- | --- |
| Таблица 9 – Описание таблицы Questions | | | |
| Название | Описание | Тип данных | Дополнительно |
| Quest\_ID | Уникальный идентификатор вопроса. | char(10) | Первичный ключ |
| Description | Текст вопроса. | Char(1000) |  |

**Таблица Answers**

В данной таблице описываются ответы, а также указывается к какому вопросу они относятся. Также информация на какой следующий вопрос переходить в случае выбора и SQL-запрос на выборку скейтбордов.

|  |  |  |  |
| --- | --- | --- | --- |
| Таблица 10 – Описание таблицы Wheels | | | |
| Название | Описание | Тип данных | Дополнительно |
| Answer\_ID | Уникальный идентификатор вопроса. | Char(10) | Первичный ключ |
| Quest\_ID | Уникальный идентификатор ответа. | Char(10) | Внешний ключ |
| Text | Текст ответа. | Char(1000) |  |
| Next\_quest\_ID | Здесь указывается ID вопроса, на который надо переходить после выбора данного варианта ответа. | Char(10) |  |
|  |  |  |  |
|  |  |  |  |
| Продолжение таблицы 10 | | | |
| Название | Описание | Тип данных | Дополнительно |
| Selection | Sql запрос на выборку скейтборда. | nvarchar(max) |  |

**Хранимые процедуры и представления**

Хранимая процедура Skates. Функция заключается в возвращении всех параметров доски и компонентов по ее уникальному идентификатору (рисунок 9).

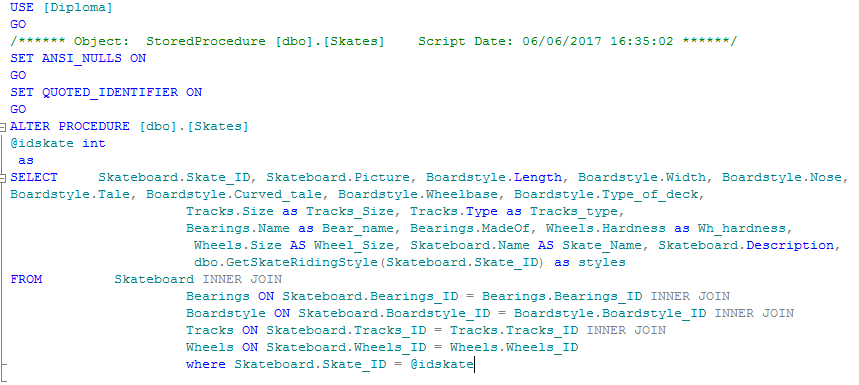


Рисунок 9 – Хранимая процедура Skates

Хранимая процедура AddRdStyles, которая используется при добавлении нового скейтборда и предназначена для присваивания ему стилей катания (рисунок 10).

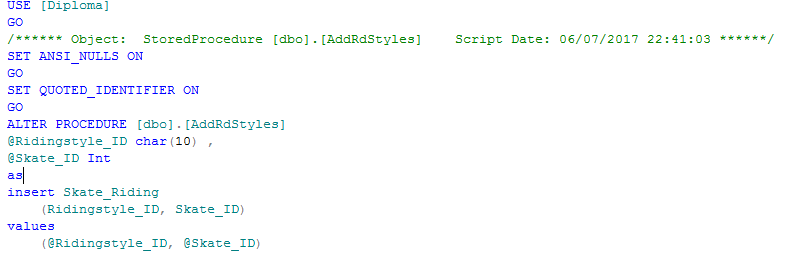


Рисунок 10 – Хранимая процедура AddRdStyles

Хранимая процедура AddSkate, которая активируется при добавлении новой доски из программы. Сама определяет максимальный id и добавляет всю внесенную информацию о скейтборде (рисунок 11).

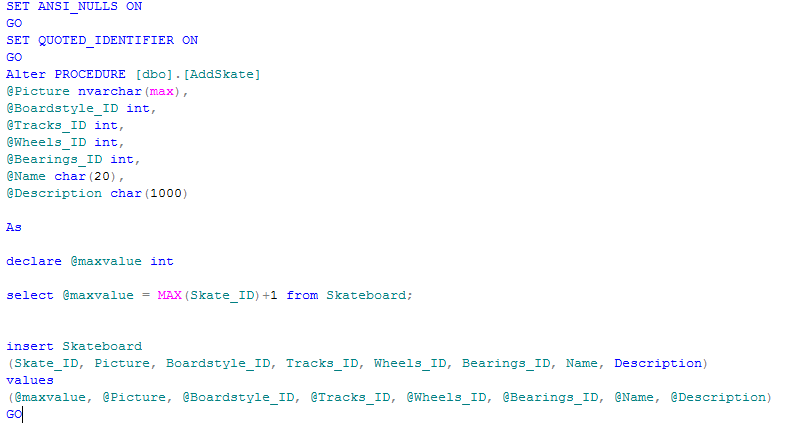


Рисунок 11 - Хранимая процедура AddSkate

* + 1. **Заполнение базы данных**

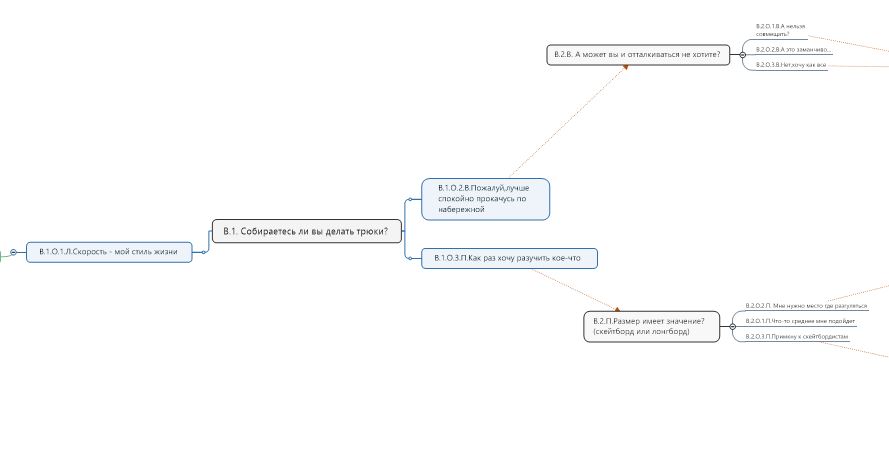
UML-диаграмма структуры вопросов-ответов (рисунок 12). Здесь представлен небольшой кусок дерева, где у каждого вопроса представлено по три варианта ответа, связанного со следующим вопросом, если он не является конечным.

Рисунок 12 – структура вопросов ответов

* + 1. **Разработка приложения**

Основные компоненты TSqlQuery, использовавшиеся в ходе разработки приложения, так как через них осуществляются логика программы и связь с базой данных (таблица 11).

|  |  |  |
| --- | --- | --- |
| Таблица 11 – описание компонентов TSqlQuery | | |
| Название | Описание | Код Sql (забитый в «свойствах» компонента) |
| Qryforid | Через данный компонент получают sql-запрос, предназначенный для дальнейшей выборки скейтбордов из общего количества. Он активируется после того как пользователь нажал на один из вариантов ответа и в запрос передался его id. | Select Selection from dbo.Answers where Answer\_id = |
| Qryquest | Предназначен для загрузки из базы данных вопросов. |  |
| Qrytestskate | Служит для загрузки ответов из базы данных, соответствующих текущему вопросу. |  |
| Qryinfo | В данном компоненте содержится основное условие выборки скейтбордов, к которому после выбора пользователем варианта ответа появляются | SELECT distinct s1.Name as Skate\_Name, s1.Skate\_ID, s1.Picture, s1.Tracks\_ID, s1.Bearings\_ID, s1.[Description], |
|  |  |  |
| Продолжение таблицы 11 | | |
| Название | Описание | Код Sql (забитый в «свойствах» компонента) |
|  | добавочные условия, посредством конструкции «where». Результатом является список досок, подходящих под составленный запрос. | s1.Name, s1.Wheels\_ID, s1.Boardstyle\_ID,  b1.Type\_of\_deck  FROM Skateboard s1  INNER JOIN Boardstyle b1 ON s1.Boardstyle\_ID = b1.Boardstyle\_ID  INNER JOIN Skate\_Riding sr1 ON s1.Skate\_ID = sr1.Skate\_ID  INNER JOIN  Riding\_Style rs1 ON rs1.Ridingstyle\_ID = sr1.Ridingstyle\_ID |
| qryonepage | Данный компонент служит для получения данных об одной доске, используя хранимую процедуру «Skates», описанную ранее. |  |

Рассмотрим блок-схемы нескольких алгоритмов работы основных процедур в программе. Принцип работы процедуры Choosevar, которая в зависимости от выбранного пользователем варианта ответа возвращает выборку скейтбордов (рисунок 13).

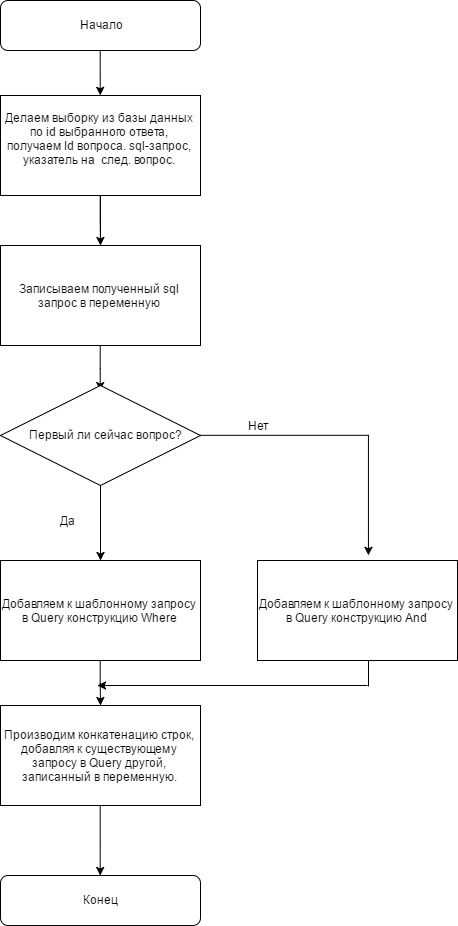


Рисунок 13 – Блок-схема работы процедуры Choosevar

Работа функции printnextquest, принцип которой заключается в выводе ответов, соответствующих текущему вопросу (рисунок 14).

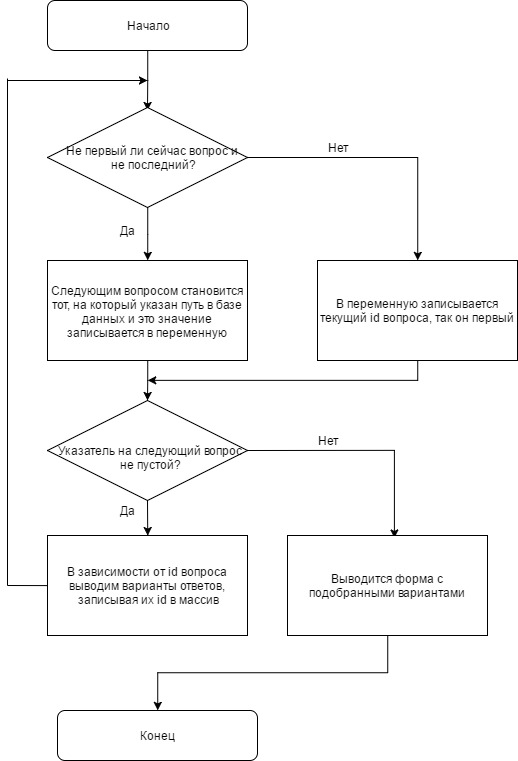


Рисунок 14 – Блок-схема работы функции printnextquest

* + 1. **Интерфейс приложения**

Основным критериями, при разработке интерфейса были интуитивность и визуальная привлекательность. Так как приложение ориентировано на тех, кто планирует приобрести скейтборд, то это в основном продвинутые современные пользователи, которых можно привлечь интересной графикой или дизайном, поэтому было решено создать, используя инструментальные средства, свое индивидуальное оформление, соответствующее заданным критериям.

Приложение «консультант скейтбордиста» состоит из четырех основных форм. Первая форма, которая открывается пользователю после запуска приложения. На ней находятся кнопки «Старт», «Wiki» и «Admin» (рисунок 15).



Рисунок 15 – Стартовая форма программы

При нажатии на кнопку «Старт» появляются вопросы и ответы (рисунок 16).

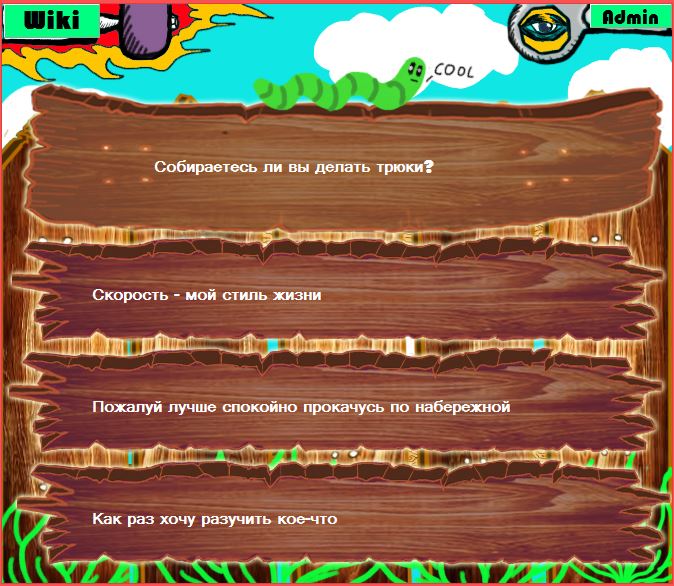


Рисунок 16 – Вывод вопросов и ответов

Ответив на последний вопрос пользователю откроется вторая форма, на которой будут выведены все результаты выборки. Так же эта форма доступна после нажатия на кнопку «Wiki» на стартовом экране (рисунок 17).



Рисунок 17 – Форма вывода результатов выборки

Если пользователя заинтересовала конкретная модель, то, нажав на кнопку «подробнее», есть возможность рассмотреть данную модель (рисунок 18).



Рисунок 18 – Форма вывода одной модели скейтборда

Так же в программе реализована не только пользовательская часть, но и администратора. Нажав на кнопку «Admin» на стартовом экране и правильно указав пару логин/пароль можно добавить новый скейтборд (рисунок 19).

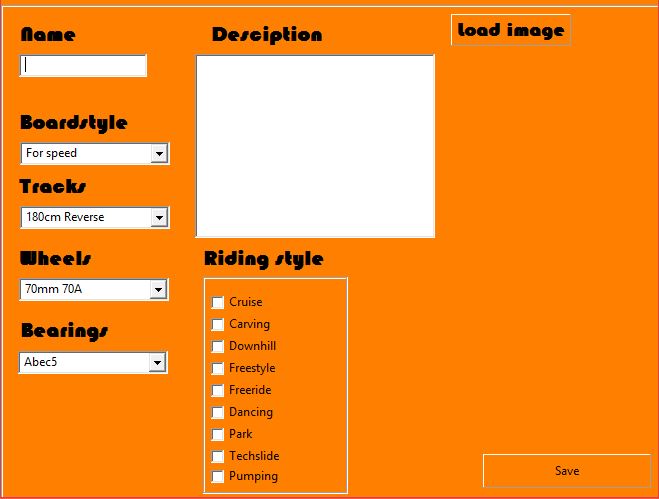


Рисунок 19 – Администраторская часть программы.

* 1. **Тестирование**

На этапе тестирования был произведен осмотр каждой функции программы и сравнение результата с ожидаемым (таблицы 12-14).

|  |  |  |
| --- | --- | --- |
| Таблица 12 – Тесты для первой формы | | |
| Действие | Ожидаемый результат | Фактический результат |
| Нажатие на кнопку «Старт» | Откроется следующее изображение, с выведенными на нем вопросами и ответами. | Ожидаемое верно. |
| Нажатие на кнопку «Admin» | Откроется панель с формой для ввода пары логин/пароль. | Ожидаемое верно. |
| Нажатие на кнопку «Wiki» | Откроется форма со всеми досками. | Ожидаемое верно. |

|  |  |  |
| --- | --- | --- |
| Таблица 13 – Тесты для формы вывода результатов | | |
| Действие | Ожидаемый результат | Фактический результат |
| Нажатие на кнопку «Next» | Форма обновится и выведет следующие три доски. После того, как дойдет до последнего скейтборда, снова покажет первые три. | Ожидаемое верно. |
| Нажатие на кнопку «Подробнее» | Откроется форма, отображающая информацию о конкретно выбранной доске. | Ожидаемое верно. |

|  |  |  |
| --- | --- | --- |
| Таблица 14 – Тесты для администраторской части | | |
| Действие | Ожидаемый результат | Фактический результат |
| Ввод в поле «Name» | Возможность ввести до 10 символов. | Ожидаемое верно. |
| Ввод в поле «Description» | Возможность ввести до 1000 символов. | Ожидаемое верно. |
| Нажатие на выпадающие списки | В компонентах Lookupcombobox выбор параметров доски, подгруженных из базы данных. | Ожидаемое верно. |
| Нажатие на кнопку «Load image» | Открывает окно «existing file», где необходимо выбрать изображение доски. В случае если | Ожидаемое верно. |
| Продолжение таблицы 14 | | |
| Действие | Ожидаемый результат | Фактический результат |
|  | выбран файл не того формата, то всплывает сообщение о неправильно выбранном формате файла. |  |
| Нажатие на кнопку «Save» | Делает Insert в базу данных, сохраняя всю информацию, введенную пользователем. Закрывает форму. | Ожидаемое верно. |

1. **ЭКОНОМИЧЕСКАЯ ЧАСТЬ**
   1. **Расчет затрат на разработку программного продукта**

Для того, чтобы рассчитать затраты на разработку программного продукта можно использовать следующую формулу (1):

, (1)

где *ЗФОТР* – общий фонд оплаты труда разработчиков ПП;

*ЗОВФ* – начисления на заработную плату разработчиков ПП во внебюджетные фонды;

*ЗЭВМ* – затраты, связанные с эксплуатацией техники;

*ЗСПП* – затраты на специальные программные продукты, необходимые для разработки ПП;

*ЗХОН* – затраты на хозяйственно-операционные нужды (бумага, литература, носители информации и т.п.);

*РН* – накладные расходы (*РН* = 30% от *ЗФОТР*).

Общее время, затраченное на разработку программного продукта, составило 3 месяца. Из него машинное время (непосредственная работа с вычислительной и оргтехникой) - 1 месяц.

Фонд оплаты труда за время работы над программным продуктом (2):

, (2)

где *ОРj* – оклад *j*-го разработчика. В разработке участвовал 1 человек, его оклад составляет 45000 руб.;

*ТРПРj* – общее время работы над ПР в месяцах, ;

 – коэффициент дополнительной зарплаты, = 0 (премий или надбавок нет);

 – районный коэффициент, = 0 (для Москвы повышенный коэффициент по законодательству РФ не предусмотрен).

Таким образом,

= 45 000 руб.

Отчисления во внебюджетные фонды складываются из страховых взносов (глава 34 части 2 НК РФ) и взносов на обязательное социальное страхование от несчастных случаев на производстве и профессиональных заболеваний.

Так как годовой фонд заработной платы разработчика не превышает 755 тыс. руб., то используются максимальные ставки страховых взносов, уплачиваемых на обязательное пенсионное страхование и на обязательное социальное страхование на случай временной нетрудоспособности. Ставка страхования от несчастных случаев в соответствии с классом профессионального риска составляет 0,2 %. Значения используемых ставок для 2017 года по видам взносов составляют (таблица 15):

Таблица 15 - Значения ставок страховых взносов и СНС

|  |  |  |
| --- | --- | --- |
| № | Наименование взноса | Размер ставок, % |
| 1. | Обязательное пенсионное страхование | 22 |
| 2. | Обязательное медицинского страхования | 5,1 |
| 3. | Обязательное социальное страхование | 2,9 |
| 4. | Страхование от несчастных случаев | 0,2 |
|  | ИТОГО | 30,2% |

Сумма начислений на заработную плату во внебюджетные фонды составляет:

,

= 13590 руб.

Затраты, связанные с использованием вычислительной и оргтехники (3):

, (3)

где – коэффициент готовности ЭВМ,  = = 0,96;

*n* – количество единиц техники, равно 1;

 – себестоимость машиночаса,  = 50руб.;

 – машинное время работы над программным продуктом, равно 1 месяцу.

Перевод рабочего времени в часы (4):

, (4)

где *Тчас* – рабочее время, ч;

*Тмес* – рабочее время, мес., (*Тмес* =1);

*ЧРД* – число рабочих дней, (*ЧРД* = 22);

*Тсм* – продолжительность рабочей смены, (*Тсм* = 8 ч);

*Ксм* – количество рабочих смен, (*Ксм* = 1).

Таким образом, время на разработку ПП с использованием ЭВМ составляет:

 часа,

 = 8448 руб.

Расчет затрат на специальные программные продукты, необходимые для разработки ПП (5):

, (5)

где *Цρ* – цена ρ-го специального программного продукта.

Перечень программных продуктов специального назначения приведен в таблице 16.

Таблица 16 - Программные продукты специального назначения

|  |  |  |
| --- | --- | --- |
| №п/п | Название ПП | Цена, руб. |
| 1 | MindManager (демо-версия) | 0 |
| 2 | Lazarus | 0 |
| 3 | Средства Microsoft SQL server | 0 |
| 4 | PowerDesigner (демо-версия) | 0 |
|  | Итого | 0 |

0 руб.

Затраты на хозяйственно-организационные нужды приведены в таблице 17 и вычисляются по формуле 6:

, (6)

где *Цτ* – цена τ-го товара, рублей.;

*Кτ* – количество τ-го товара.

Таблица 17 - Затраты на хозяйственно-организационные нужды

|  |  |  |  |
| --- | --- | --- | --- |
| Наименование | Цена за единицу (руб.) | Кол-во (шт.) | Всего (руб.) |
| Пачка бумаги | 500 | 1 | 500 |
| СD-диск | 50 | 2 | 100 |
| Картридж для принтера | 2500 | 1 | 2500 |
| Итого |  |  | 3100 |

 3100 руб.

Накладные расходы (7):

, (7)

*Рн*= руб.

В результате, затраты на разработку программного продукта составят:

45000 + 13590 + 8448 + 3100 +15000 = 85138 руб.

1. **Расчет затрат на внедрение программного продукта**

Затраты на внедрение программного продукта (*КВПР*) (8):

, (8)

где *ЗМ* – затраты на приобретение материалов, руб.;

*ЗКТС* – затраты на приобретение комплекса технических средств, руб.;

*ЗПО* – затраты на приобретение программного обеспечения (включают стоимость разработанного ПП, а также других существующих ПП, необходимых для функционирования системы), руб.;

*ЗФОТВ* – затраты на оплату труда работников, занятых внедрением проекта, руб.;

*ЗОВФ* – отчисления во внебюджетные фонды с заработной платы работников, занятых внедрением проекта, руб.;

*ЗЭВМ* – затраты, связанные с эксплуатацией ЭВМ при внедрении проектного решения, руб.;

*Рком* – командировочные расходы, руб.;

*РН* – накладные расходы, руб.;

*kТУН* – коэффициент транспортирования, установки и наладки комплекса технических средств, определяется действующими нормативами организации, а также спецификой конкретного проекта.

Для внедрения программного продукта требуются следующие расходные материалы: 1 CD-диск, стоимостью 50 рублей. Следовательно *ЗМ* = 50 рублей. Так же требуется дополнительное приобретение одного компьютера-терминала в зал, следовательно, *ЗКТС* = 20000 рублей.

Затраты на приобретение программного обеспечения в данном случае равны стоимости одной лицензионной копии Windows и составляют *ЗПО* = 7300 рублей.

Внедрением занят один системный инженер с окладом 45000 руб. Время внедрения – 1 рабочий день. По формуле рассчитываем затраты на оплату труда и отчисления во внебюджетные фонды.

*ЗФОТВ* = руб,

где 22 – это количество рабочих дней в марте 2017, так как в тот месяц проводилось внедрение ПП .

*ЗОВФ* = 617 руб.

Затраты, связанные с эксплуатацией ЭВМ при внедрении проектного решения составят:

 руб.

Командировочные расходы при внедрении программного продукта не планируется, следовательно, *Рком*= 0.

Так как коэффициент накладных расходовпо данным организации составляет *kНР* = 0,3, то величина накладных расходов равна 613 руб.

Суммарные затраты на внедрение составят:

 = 31 009 руб.

## Расчет текущих затрат по программному продукту

Чтобы рассчитать величину годовых затрат, которые нес магазин, чтобы получить результаты аналогичные тем, которые достигаются при внедрении разработанного программного продукта используем следующую формулу (9):

, (9)

где *ЗП1* – затраты на оплату труда сотрудника на выполнение функций до внедрения проектного решения,

*ОТвн1* – отчисления во внебюджетные фонды;

*ЗЭВМ1* – затраты, связанные с эксплуатацией ЭВМ, принимаются равными 0 рублей;

*Мз1* – годовые материальные затраты на сопровождение программного продукта составляют 0 рублей, так как аналогичного программного продукта на предприятии не существует;

*НР1* – накладные расходы.

Временные затраты работы сотрудника в месяцах (10):

, (10)

где Т1час – время, затрачиваемое сотрудником на работу, которую предлагается автоматизировать при помощи ПП, в месяцах и часах соответственно (Т1 час = часов);

 – принятое нами среднее число рабочих дней в месяц за год;

*Чрч* – число рабочих часов в день.

 = 11 месяцев.

Тогда затраты на оплату труда сотрудника (11):

, (11)

где Ос – оклад продавца-консультанта (составляет 25 000 руб.);

 руб.

Отчисления на социальные нужды вычисляются (12):

, (12)

 руб.

Подставив соответствующие значения в формулу, получим:

 руб.

Годовые затраты на эксплуатацию системы после внедрения программного продукта рассчитываются аналогично по формуле (13):

, (13)

где *ЗП2* – затраты на оплату труда сотрудника после внедрения;

*ОТвн2* – отчисления во внебюджетные фонды;

*ЗЭВМ2* – затраты, связанные с эксплуатацией ЭВМ после внедрения, принимаются равными 1500 рублей;

*Мз2 –* материальные затраты, годовые материальные затраты на сопровождение программного продукта составляют:

*-* тех. поддержка и подписка на обновления 1500 руб/мес, т.е. в год сумма = руб.

*НР2* – накладные расходы.

Временные затраты работы сотрудника в месяцах (14):

, (14)

где *Т2мес, Т2час* – время, затрачиваемое сотрудником на обработку результатов, в месяцах и часах, но так как программа полностью заменяет одного продавца консультанта, то *Т2час* = 0 часов;

*Чрд* – принятое нами среднее число рабочих дней в месяц за год;

*Чрч* – число рабочих часов в день.

 = 0,00625 месяцев.

Тогда затраты на оплату труда сотрудника (15):

, (15)

где Ос – оклад сотрудника (оклад составляет 25 000 руб.).

 руб.

Отчисления на социальные нужды (16):

, (16)

 руб.

Подставив соответствующие значения в формулу, получим:

 = 22001 руб.

Таким образом, текущие затраты магазина до внедрения разработанного программного продукта составляют 189915 рублей, после внедрения 22001 рубль.

1. **Расчет экономической целесообразности разработки и внедрения информационных технологий**

Для разрабатываемого проекта расчет экономической эффективности производится исходя из следующих условий:

* годовые текущие затраты до внедрения автоматизированной системы, учитывая, что эта сумма является годовой зарплатой продавца-консультанта,

*С1 =* 189915 руб.;

* годовые текущие затраты после внедрения системы, учитывая, что программа выполняет обязанности продавца-консультанта,

*С2 =* 22001 руб.;

* горизонт расчета принимается исходя из предполагаемого срока использования разработки, равен 4 годам, т.е. *Т=Тн=* 4;
* шаг расчета равен одному году, *t* = 1 ;
* капитальные вложения равны затратам на создание ПП,

*К* = 85138 руб.;

* норма дисконта равна норме дохода на капитал, принимаемая для расчета *Е* = 10%.

Ожидаемая условно-годовая экономия от внедрения системы (17):

, (17)

где *Эуг* – величина экономии, руб.;

*С*1 – годовые текущие затраты до внедрения автоматизированной системы, руб.;

*С*2 – годовые текущие затраты после внедрения системы, руб.;

*∑Эi* – ожидаемый дополнительный эффект от различных факторов, руб.

Так как основным фактором, по которому производится расчет экономической эффективности от внедрения программного продукта, является уменьшение затрат на оплату труда продавца-консультанта, то дополнительный эффект отсутствует, т.е. *∑Эi* =0.

Подставив вычисленные выше значения в формулу, получим:

Эуг = 189915 – 22001 + 0 = 167914 руб.

где *Эуг* – ожидаемая условно-годовая экономия, руб.

Величина ожидаемого годового экономического эффекта от внедрения ПП (18):

, (18)

где *Эг* – ожидаемый годовой экономический эффект, руб.;

*Эуг* – ожидаемая условно-годовая экономия, руб.;

*К* – капитальные вложения (равны затратам на создание ПП), руб.;

*Ен* – нормативный коэффициент экономической эффективности капитальных вложений .

Нормативный коэффициент экономической эффективности капитальных вложений (19):

, (19)

где *Тн* – нормативный срок окупаемости капитальных вложений, лет.

Предполагаемый срок окупаемости 4 года.

Подставив вычисленные выше значения в формулу (18), получим:

*Эг = =* 146629 руб*.*

Расчетный коэффициент экономической эффективности капитальных вложений составляет (20):

, (20)

где *Ер* – расчетный коэффициент экономической эффективности капитальных вложений;

*Эуг* – ожидаемая условно-годовая экономия, руб.;

*К* – капитальные вложения на создание системы, руб.

Подставив вычисленные выше значения в формулу, получим:

Расчетный срок окупаемости капитальных вложений составляет (21):

, (21)

где *Ер* – коэффициент экономической эффективности капитальных вложений.

Подставив вычисленные выше значения в формулу, получим:

 0,5 лет.

Срок окупаемости без дисконтирования =6 месяцев.

Показатели экономической целесообразности разработки и внедрения программного продукта сведены в результирующей таблице 18.

Таблица 18 - Показатели экономической целесообразности разработки и внедрения программного продукта.

|  |  |
| --- | --- |
| Наименование показателя | Значения |
| Затраты на разработку и внедрение ПП, руб. | 116147 |
| Затрат на эксплуатацию, руб/год : |  |
| - до внедрения | 189 915 |
| - после внедрения | 22001 |
| Ожидаемая экономия от внедрения ПП, руб.: |  |
| - в год | 167914 |
| - в месяц | 13993 |
| Срок окупаемости, месяц (затраты/экономия в мес.) | 6 |

Произведенные расчеты свидетельствуют, что внедрение, разработанного программного продукта, позволит заменить одного сотрудника (продавца-консультанта), что приведет к сокращению годовых текущих затрат на 167914 руб.

Опираясь на оценку экономической эффективности можно сделать вывод о том, что разработка и внедрение предлагаемого программного продукта является экономически обоснованной и целесообразной.

1. **ОХРАНА ТРУДА И ЖИЗНЕДЕЯТЕЛЬНОСТИ**

Основываясь на ГОСТ 12.1.033-81 введен 1982-07-01.

Соблюдение правил техники безопасности обеспечивает сохранение здоровья работника на рабочем месте.

Для обеспечения комфортной и правильной работы на ПЭВМ также нужно соблюдать санитарно-гигиенические нормы:

* обязательно заземление компьютера;
* нормы освещённости: экрана – 100-250 Лк, поверхность стола в зоне размещения рабочего документа – 300-500 Лк;
* чередование работы и перерывов в работе: 5-10 мин. после каждого часа работы или 15-20 минут после каждых двух часов работы;
* продолжительность непрерывной работы - не более 2 часов;
* обеденный перерыв не входит в суммарное время регламентированных перерывов; ночью (с 22 до 6 часов) суммарное время регламентированных перерывов увеличивается на 60 минут;
* при несоответствии гигиенических условий труда нормативам время перерывов увеличивается на 30 %;
* соблюдать оптимальное расстояние от глаз до экрана монитора (50-70 см); температура в помещении – 21-25 0С;
* относительная влажность – 40-60 %;
* скорость движения воздуха – 0,1 м/с;
* обязательна ежедневная влажная уборка;
* убедиться в отсутствии видимых повреждений аппаратуры и рабочего места;
* должно быть подъёмно поворотное кресло, регулируемое по высоте и углам наклона сидения и спинки; ширина сидения – 400 мм;
* стол должен быть с выдвижной доской для клавиатуры (высота – 725 мм, ширина - 800-1000 мм, длина – 800-1200 мм);
* высота сидения стула должна быть отрегулирована в соответствии с ростом ученика, чтобы линия взора приходилась на центр экрана монитора;
* монитор должен быть установлен так, чтобы не было бликов;
* в кабинете должны быть регулируемые жалюзи;
* расстояние между мониторами – 2 метра (если они друг за другом), 1,5 метра (рядом);
* минимальные размеры рабочего места одного пользователя: площадь – 6 м2, объём – 20 м3;
* беременным и кормящим матерям компьютер вреден;
* нельзя устанавливать компьютер возле источников тепла;
* следует избегать сгибов и защемлений питающего кабеля и проводов;
* работать только чистыми руками, плавно нажимая на клавиши, не допуская резких ударов, нельзя класть книги, дискеты и тетради на монитор, системный блок и клавиатуру;
* нельзя работать при плохом самочувствии, с влажными руками, во влажной одежде;
* допустимый уровень шума – 50 - 75 дБ.

1. **Общие требования безопасности**

К занятиям допускаются работники, прошедшие инструктаж по охране труда с росписью в журнале.

Лица, работающие на ЭВМ, не допускаются к работе в состоянии наркотического, токсического и алкогольного опьянения.

Осторожно и бережно обращаться с дорогостоящей и сложной аппаратурой, установленной в компьютерной мастерской. Соблюдать правила пожарной и электробезопасности. Соблюдать порядок включения и выключения ПЭВМ. Следить за исправностью аппаратуры и немедленно прекратить работу при появлении необычного звука, запаха или самопроизвольного отключения аппаратуры. Нельзя прикасаться к корпусу ПЭВМ, питающим проводам и предметам, связанным с землёй (батареи отопления, водопровод). Нельзя прикасаться к экрану и тыльной стороне монитора, клавиатуры. При работе с принтером нельзя перемещать бумагу вручную, не печатать с открытой крышкой.

1. Требования безопасности перед началом работы

Перед началом работы лица, работающие на ЭВМ, обязаны подготовить свое рабочее место: расположить монитор, клавиатуру и «мышь» таким образом, чтобы работать было удобно.

Перед выполнением работы, внимательно изучить её содержание и порядок выполнения.

Не приступать к выполнению работы без разрешения руководителя.

Электрический щиток в кабинете включать работникам запрещается.

1. Требования безопасности во время работы

Во время работы быть внимательным и аккуратным, не отвлекаться на посторонние разговоры, точно выполнять указания руководителя.

Включать и выключать компьютер разрешается только с разрешения руководителя.

Запрещается передвигать работающий компьютер, трогать соединительные провода, прикасаться к тыльной стороне всех устройств, класть вещи на аппаратуру, самим устранять любые неисправности в работе аппаратуры.

Во время выполнения работы не прикасаться и не наклоняться (особенно с неубранными волосами) к вращающимся частям компьютера.

Не держать на рабочем месте предметы, не требующиеся для выполнения задачи.

Соблюдать расстояние до экрана монитора не менее 50-70 см.

Запрещается работать за компьютером без очков работникам, имеющим плохое зрение.

После 30 минут работы за компьютером необходимо выполнить комплекс упражнений для глаз (в течение 2-3 минут).

1. Техника безопасности в аварийной ситуации

Техника безопасности в аварийной ситуации:

Обнаружив неисправность в работе компьютера, необходимо немедленно сообщить руководителю и отключить компьютер от сети.

В случае возникновения пожара принять меры к ликвидации очага возгорания, вызвать пожарную команду по телефону 01, сообщить о случившемся руководителю.

Во всех случаях поражения электротоком, необходимо вызвать врача независимо от состояния пострадавшего.

В случае внезапного заболевания или обморочного состояния сообщить об этом руководителю и действовать по его указанию.

Требования безопасности по окончании занятий:

После выполненной работы, подготовить компьютер к выключению.

Привести в порядок свое рабочее место.

**Вывод по охране труда и технике безопасности**

Если не нарушать изложенные выше правила по технике безопасности можно полностью избежать внештатных ситуаций, аварий.

# ЗАКЛЮЧЕНИЕ

В процессе выполнения дипломной работы было разработано приложение «консультант скейтбордиста». Результатом выполненной работы является готовая система, состоящая из приложения пользователя и базы данных.

Благодаря разработанной системе пользователи имеют возможность получить интерактивную помощь при подборе скейтборда, и узнать необходимые сведения о каждой модели. Предпочтения каждого пользователя точно учитываются, и ответив на последний вопрос выдается ряд скейтбордов, по подходящим ему критериям. Так же, администратор может добавлять новые скейтборды в базу, посредством специальной части программы.

Данное приложение может быть использовано в специализированных спортивных магазинах и позволит либо заменить одного продавца-консультанта в зале, либо упростить его работу. Приложение в интерактивной форме поможет разобраться новичкам в устройстве и классификации скейтбордов, не прибегая к сложной терминологии, и сэкономив время людей-консультантов.

По итогам проделанной работы были изучены:

* новые способы разработки систем и приложений;
* устройство и организация систем поддержки принятия решений;
* основы работы с новыми инструментальными средствами;

Так же были закреплены уже имеющиеся навыки программирования на Object Pascal и Transact SQL, а в ходе разработки освоено несколько новых, ранее не изученных особенностей языков.

В дальнейшем система будет развиваться и возможными расширениями будут:

* Перенос приложения на Web;
* Расширение дерева вопросов;
* Переработка алгоритма;
* Изменение интерфейса под сенсорный терминал.

Из всего вышеперечисленного можно сделать вывод, что приложение «консультант скейтбордиста» является уникальной и полезной разработкой, которая может быть полезна как пользователям, так и предприятию, внедрившему эту систему.

# СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

1. МЕТОДИЧЕСКИЕ УКАЗАНИЯ ПО ОФОРМЛЕНИЮ ВЫПУСКНОЙ КВАЛИФИКАЦИОННОЙ РАБОТЫ (ДИПЛОМНЫЙ ПРОЕКТ, ДИПЛОМНАЯ РАБОТА). – М. Министерство образования и науки Российской Федерации ФГБОУ ВО «МГУТУ им. К.Г.Разумовского (ПКУ)» Университетский колледж информационных технологий, 2017
2. Справочник MSDN SQL SERVER [Электронный ресурс] – URL: <https://msdn.microsoft.com/en-us/library/mt590198(v=sql.1).aspx>
3. Lazarus Wiki [Электронный ресурс] – URL: <http://wiki.freepascal.org/>
4. "Типовая инструкция по охране труда при работе на персональном компьютере. ТОИ Р-45-084-01" (утв. Приказом Минсвязи РФ от 02.07.2001 N 162)
5. Методические указания к выполнению экономической части выпускной квалификационной работы для студентов специальности 080801 «Прикладная информатика (по областям)» [Электронный ресурс] – URL: <http://pandia.ru/text/79/122/14105-4.php>
6. ГОСТ 2.105-95 – Единая система конструкторской документации. Общие требования к текстовым документам;
7. ГОСТ 2.111-2013 – Единая система конструкторской документации. Нормоконтроль;
8. ГОСТ 7.1-2003 – Система стандартов по информации, библиотечному и издательскому делу. Библиографическая ссылка. Общие требования и правила составления;
9. ГОСТ 7.32-2001 – Отчет о научно-исследовательской работе;
10. ГОСТ 7.82-2001 – Система стандартов по информации, библиотечному и издательскому делу. Библиографическая ссылка. Общие требования и правила составления;
11. ГОСТ 19.201-78 – Техническое задание. Требования к содержанию и оформлению;
12. Уринцов А.И., Дик В.В. «Системы поддержки принятия решений» [Текст]/ Уринцов А.И., Дик В.В. М.: МЭСИ, 2008. (дата обращения 01.05.2017)
13. Графические возможности Delphi.Canvas. / Учебник по Delphi [Электронный ресурс] – URL: http://shiva16.narod.ru/Unit24.html (дата обращения 12.05.2017);
14. Системы поддержки принятия решений : учебник и практикум для бакалавриата и магистратуры / В. Г. Халин [и др.] ; под ред. В. Г. Халина, Г. В. Черновой. — М. : Издательство Юрайт, 2015. — 494 с. (дата обращения 01.05.2017)
15. Грофф Джеймс Р., Вайнберг Пол Н., Оппель Эндрю Дж. : «SQL. Полное руководство» Издательство: Вильямс, 2016 г. (дата обращения 28.05.2017)

# ПРИЛОЖЕНИЕ А

**Иллюстрации к таблице 1**

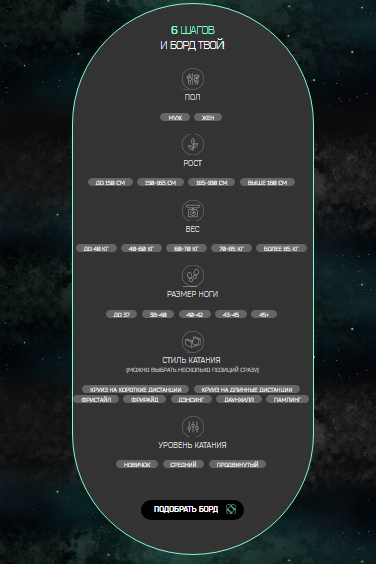


Рисунок 1 – Фильтр товаров на Allaboardshop.

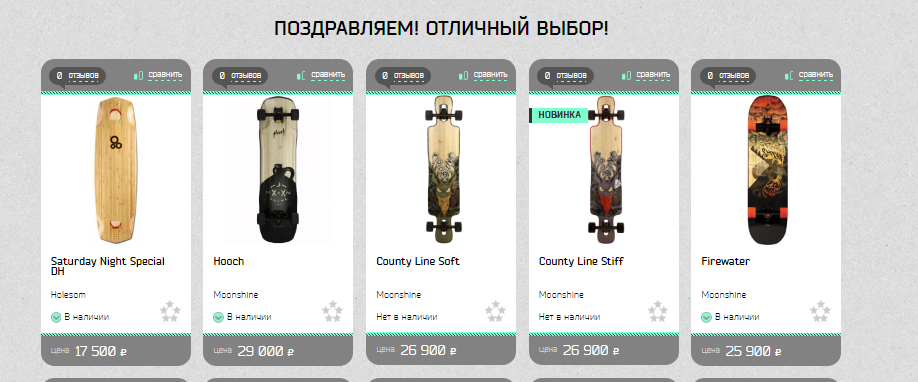


Рисунок 2 – Вывод скейтбордов после фильтрации на Allaboardshop.

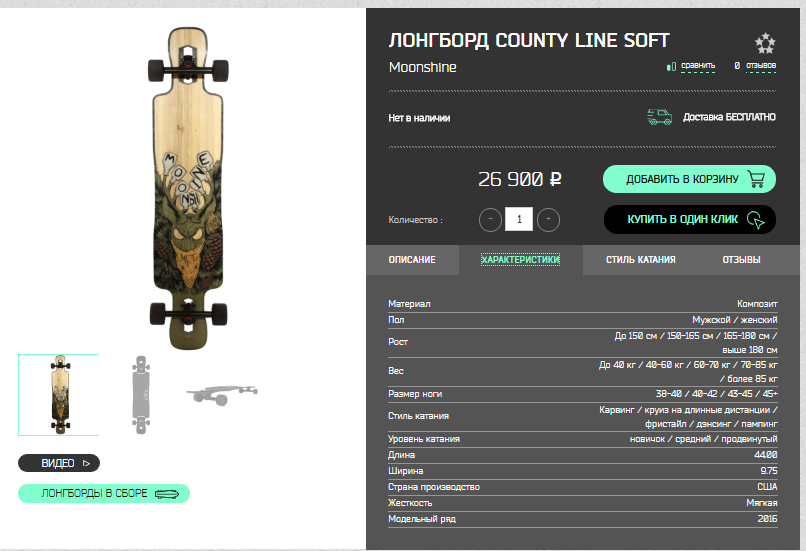


Рисунок 3 – Выбранный скейтборд на Allaboardshop.

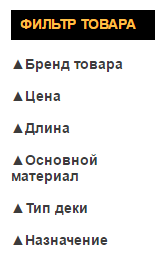
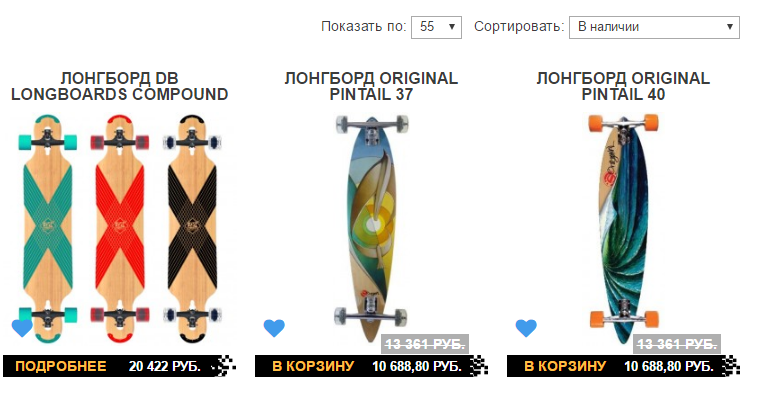


Рисунок 4 – Фильтр товаров на X3mboardshop.

Рисунок 5 – Результат фильтрации на сайте X3mboardshop.

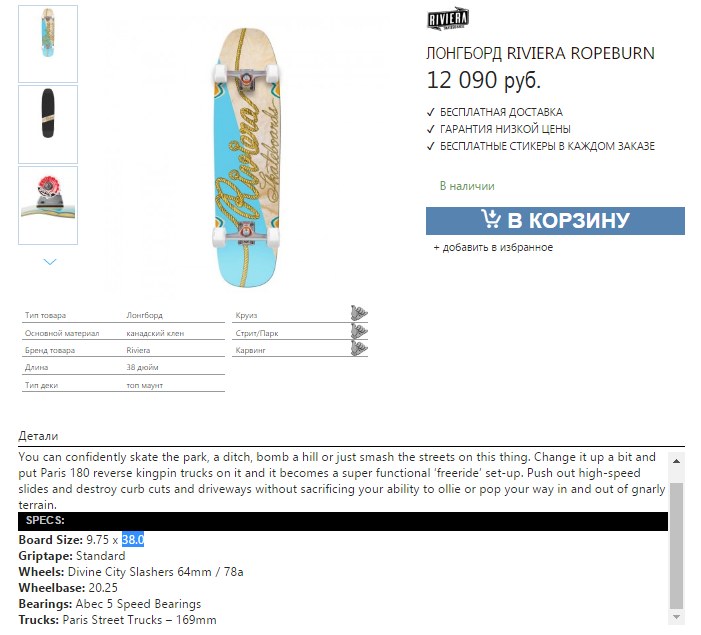


Рисунок 6 – Экран отображения скейтборда на X3mboardshop.

# ПРИЛОЖЕНИЕ Б

**Листинг программы**

unit Unit1;

{$mode objfpc}{$H+}

interface

uses

Classes, SysUtils, mssqlconn, sqldb, DB, FileUtil, Forms, Controls, Graphics,

Dialogs, StdCtrls, ExtCtrls, Grids, DBCtrls, DBGrids, maskedit, Unit2, Unit3;

type

{ TForm1 }

TForm1 = class(TForm)

DataSource1: TDataSource;

DBGrid1: TDBGrid;

Password: TEdit;

Memo1: TMemo;

Rbvar1: TLabel;

QueryQuest: TSQLQuery;

Qryforid: TSQLQuery;

MSSQLConnection1: TMSSQLConnection;

qryTestSkate: TSQLQuery;

Rbvar2: TLabel;

Rbvar3: TLabel;

SQLTransaction1: TSQLTransaction;

DSquestion: TDataSource;

DBtxtquest: TDBText;

Image1: TImage;

Startimg: TImage;

OkPanel: TPanel;

Login: TEdit;

Forpassword: TLabel;

GroupBox1: TGroupBox;

foradm: TGroupBox;

Forlogin: TLabel;

Admbutt: TPanel;

Wiki: TPanel;

procedure AdmbuttClick(Sender: TObject);

procedure DBNavigator1Click(Sender: TObject; Button: TDBNavButtonType);

procedure FormCreate(Sender: TObject);

procedure Label1Click(Sender: TObject);

procedure Label2Click(Sender: TObject);

procedure Label3Click(Sender: TObject);

procedure MouseEnter(vars: Tlabel);

procedure MouseLeave(vars: Tlabel);

procedure bbMouseEnter(BB:TPanel);

procedure bbMouseLeave(BB:TPanel);

procedure StartIMGClick(Sender: TObject);

procedure OkPanelClick(Sender: TObject);

procedure printnextquest(isfirst: boolean = False);

procedure WikiClick(Sender: TObject);

procedure Choosevar(var\_id: string);

private

{ private declarations }

public

{ public declarations }

end;

var

Form1: TForm1;

mass\_label\_id: array [1..3] of string;

first\_quest: boolean;

is\_clicked: boolean;

implementation

{$R \*.lfm}

{ TForm1 }

procedure Tform1.printnextquest(isfirst: boolean = False);

var

iLabel: integer;

next\_quest\_id: string;

begin

if (not isfirst) and (not QueryQuest.EOF) then

begin

Qryforid.Active := True;

next\_quest\_id := Qryforid.FieldByName('Next\_quest\_id').AsString;

end

else if (isfirst) then

begin

next\_quest\_id := QueryQuest.FieldByName('quest\_id').Value;

end;

if (next\_quest\_id <> '') then

begin

qryTestSkate.Filtered := False;

qryTestSkate.Filter := 'quest\_id="' + next\_quest\_id + '"';

qryTestSkate.Filtered := True;

qryTestSkate.First;

iLabel := 1;

while (not qryTestSkate.EOF) do

begin

case iLabel of

1 : Rbvar1.Caption := qryTestSkate.FieldByName('Text').Value;

2 : Rbvar2.Caption := qryTestSkate.FieldByName('Text').Value;

3 : Rbvar3.Caption := qryTestSkate.FieldByName('Text').Value;

end;

mass\_label\_id[iLabel] := qryTestSkate.FieldByName('Answer\_Id').AsString;

Inc(iLabel);

qryTestSkate.Next;

end;

end

else

begin

Rbvar3.Visible := False;

Rbvar2.Visible := False;

Rbvar1.Visible := False;

Startimg.Visible := True;

QueryQuest.Active := False;

qryTestSkate.Active := False;

Image1.Visible := False;

FORM3.Qryinfo.First;

form3.ShowModal;

//ShowMessage('Last question');

end;

end;

procedure TForm1.WikiClick(Sender: TObject);

begin

Form3.ShowModal;

end;

procedure TForm1.Label1Click(Sender: TObject);

begin

if (not is\_clicked) then

begin

is\_clicked := True;

Choosevar(mass\_label\_id[(Sender as tlabel).Tag]);

printnextquest;

DBtxtquest.Refresh;

is\_clicked := False;

end;

end;

procedure TForm1.Label2Click(Sender: TObject);

begin

if (not is\_clicked) then

begin

is\_clicked := True;

Choosevar(mass\_label\_id[(Sender as tlabel).Tag]);

printnextquest;

DBtxtquest.Refresh;

is\_clicked := False;

end;

end;

procedure TForm1.Label3Click(Sender: TObject);

begin

if (not is\_clicked) then

begin

is\_clicked := True;

Choosevar(mass\_label\_id[(Sender as tlabel).Tag]);

printnextquest;

DBtxtquest.Refresh;

is\_clicked := False;

end;

end;

procedure TForm1.MouseEnter(vars: Tlabel);

begin

Vars.Font.Color := $AACD66;

end;

procedure TForm1.MouseLeave(vars: Tlabel);

begin

Vars.Font.Color := Clwhite;

end;

procedure TForm1.StartIMGClick(Sender: TObject);

begin

form3.Qryinfo.SQL.clear;

form3.Qryinfo.SQL.Text:=Memo1.Text;

is\_clicked := False;

first\_quest := True;

Rbvar3.Visible := True;

Rbvar2.Visible := True;

Rbvar1.Visible := True;

Startimg.Visible := False;

Wiki.Enabled := True;

QueryQuest.Active := False;

QueryQuest.SQL.Clear;

QueryQuest.SQL.Add('select quest\_id,description from dbo.questions where quest\_id=''В.1.''');

QueryQuest.Active := True;

qryTestSkate.Active := False;

qryTestSkate.SQL.Clear;

qryTestSkate.SQL.Add('select \* from dbo.test\_quest\_skate');

qryTestSkate.Active := True;

DBtxtquest.DataField := 'Description';

printnextquest(True);

Image1.Visible := True;

end;

procedure Tform1.Choosevar(var\_id: string);

var

cond: string;

sql\_text: string;

sql\_text\_cond: string;

begin

sql\_text := 'select Quest\_id,Selection,Next\_quest\_id from Answers where Answer\_id=''' +

var\_id + '''';

Qryforid.sql.Text := sql\_text;

Qryforid.Active := True;

cond := Qryforid.FieldByName('Selection').AsString;

Qryforid.Active := False;

if cond <> '' then

begin

sql\_text\_cond := form3.qryinfo.SQL.Text;

Form3.Qryinfo.Active := False;

if (not first\_quest) then

begin

sql\_text\_cond := sql\_text\_cond + ' And ';

end

else

begin

first\_quest := False;

sql\_text\_cond := sql\_text\_cond + ' Where ';

end;

sql\_text\_cond := sql\_text\_cond + cond;

Form3.Qryinfo.SQL.Text := sql\_text\_cond;

Form3.Qryinfo.Active := True;

end

else

ShowMessage('1');

end;

procedure TForm1.bbMouseEnter(BB:TPanel);

begin

BB.Color:= $32CD9A ;

end;

procedure TForm1.bbMouseLeave(BB:TPanel);

begin

bb.Color:=$0080FF00;

end;

procedure TForm1.OkPanelClick(Sender: TObject);

begin

if (login.Text <> MSSQLConnection1.UserName) and (Password.text <> MSSQLConnection1.Password)

then

begin

ShowMessage('Incorrect password or username!');

Login.Text:='';

Password.Text:='';

foradm.Visible:=false;

end

else

begin

Login.Text:='';

Password.Text:='';

foradm.Visible:=false;

Unit2.tr1:=Self.SQLTransaction1;

Form2.ShowModal;

end;

end;

procedure TForm1.FormCreate(Sender: TObject);

begin

Wiki.Enabled := False;

end;

procedure TForm1.AdmbuttClick(Sender: TObject);

begin

if Foradm.Visible = False then

Foradm.Visible := True

else

Foradm.Visible := False;

end;

procedure TForm1.DBNavigator1Click(Sender: TObject; Button: TDBNavButtonType);

begin

if (button = nbNext) then

begin

printnextquest;

end;

DBtxtquest.Refresh;

end;

end.

unit Unit2;

{$mode objfpc}{$H+}

interface

uses

Classes, SysUtils, sqldb, DB, FileUtil, Forms, Controls, Graphics,

Dialogs, StdCtrls, ExtCtrls, DBCtrls;

type

{ TForm2 }

TForm2 = class(TForm)

Bearings: TLabel;

Boardstyle: TLabel;

dsBoardStyle: TDataSource;

dsTracks: TDataSource;

dsWheels: TDataSource;

dsBear: TDataSource;

Lblpath: TLabel;

lcmbBoardStyle: TDBLookupComboBox;

dsAdminSkate: TDataSource;

Fordescr: TEdit;

Lblname: TLabel;

chkRidingStyles: TCheckGroup;

Forname: TEdit;

GroupBox2: TGroupBox;

Image1: TImage;

Imgload: TPanel;

Lblname1: TLabel;

lcmbTracks: TDBLookupComboBox;

lcmbWheels: TDBLookupComboBox;

lcmbBearings: TDBLookupComboBox;

//OpenDialog1: TOpenDialog;

OpenDialog2: TOpenDialog;

Panel1: TPanel;

qryaddrdstl: TSQLQuery;

qryTracks: TSQLQuery;

qryWheels: TSQLQuery;

qryBear: TSQLQuery;

Ridingstyle: TLabel;

qryAdminSkate: TSQLQuery;

qryBoardStyle: TSQLQuery;

qryadd: TSQLQuery;

Tracks: TLabel;

Wheels: TLabel;

procedure FormCreate(Sender: TObject);

procedure FormShow(Sender: TObject);

procedure ImgloadClick(Sender: TObject);

procedure Panel1Click(Sender: TObject);

private

{ private declarations }

public

{ public declarations }

end;

var

Form2: TForm2;

strRidingStyles: TStringList;

tr1:TSQLTransaction;

implementation

{$R \*.lfm}

{ TForm2 }

procedure TForm2.ImgloadClick(Sender: TObject);

begin

if OpenDialog2.Execute then

try

Image1.Picture.LoadFromFile(OpenDialog2.FileName);

lblpath.Caption:=OpenDialog2.FileName;

except

on EInvalidGraphic do

ShowMessage('Ошибка загрузки')

else

ShowMessage('Прочие ошибки: ' + #13 + Exception(ExceptObject).Message);

end;

end;

procedure TForm2.Panel1Click(Sender: TObject);

var

i,id\_board:integer;

name\_txt ,Descr\_txt,Tr\_id ,brd\_id,wh\_id,bear\_id,path,rd\_id:string;

begin

name\_txt:= forname.Text;

Descr\_txt:= Fordescr.Text;

tr\_id:=lcmbTracks.KeyValue;

bear\_id:=lcmbBearings.KeyValue;

wh\_id:=lcmbWheels.KeyValue;

brd\_id:=lcmbBoardStyle.KeyValue;

path:=Lblpath.Caption;

qryadd.SQL.Text:='exec dbo.AddSkate @Picture=:path, @Boardstyle\_ID=:brd\_id, @Tracks\_ID=:tr\_id, @Wheels\_ID=:wh\_id, @Bearings\_ID=:bear\_id, @Name=:name\_txt, @Description=:Descr\_txt';

qryadd.ParamByName('path').AsString := path;

qryadd.ParamByName('brd\_id').AsInteger := StrToInt(brd\_id) ;

qryadd.ParamByName('tr\_id').AsInteger := StrToInt(tr\_id);

qryadd.ParamByName('wh\_id').AsInteger := StrToInt(wh\_id);

qryadd.ParamByName('bear\_id').AsInteger := StrToInt(bear\_id);

qryadd.ParamByName('name\_txt').AsString := name\_txt;

qryadd.ParamByName('descr\_txt').AsString := Descr\_txt;

qryadd.ExecSQL;

qryaddrdstl.Sql.Add('Select MAX(Skate\_ID) from Skateboard');

qryaddrdstl.Active:=True;

id\_board:=StrToInt(qryaddrdstl.Fields[0].AsString);

qryaddrdstl.Active:=False;

qryaddrdstl.SQL.Clear;

for i:=0 to (chkRidingStyles.items.Count)-1 do begin

if (chkRidingStyles.Checked[i]) then

begin

rd\_id:=strRidingStyles[i];

qryaddrdstl.SQL.Text:='exec dbo.AddRdStyles @Ridingstyle\_ID=:rd\_id, @Skate\_id=:id\_board';

qryaddrdstl.ParamByName('rd\_id').AsString :=rd\_id;

qryaddrdstl.ParamByName('id\_board').AsInteger := id\_board;

qryaddrdstl.ExecSQL;

tr1.Commit;

end;

end;

end;

procedure TForm2.FormShow(Sender: TObject);

var

txtSql: string;

txtElement: string;

begin

qryAdminSkate.Active := False;

//загрузка Riding\_style

txtSql := 'select Ridingstyle\_id,ridingstyle from Riding\_Style';

qryAdminSkate.SQL.Text := txtSql;

qryAdminSkate.Active := True;

strRidingStyles.Clear;

chkRidingStyles.Items.Clear;

while (not qryAdminSkate.EOF) do

begin

txtElement := qryAdminSkate.FieldByName('Ridingstyle\_id').AsString;

strRidingStyles.Add(txtElement);

txtElement := qryAdminSkate.FieldByName('Ridingstyle').AsString;

chkRidingStyles.Items.Add(txtElement);

qryAdminSkate.Next;

end;

qryAdminSkate.Active := False;

qryBoardStyle.Active := True;

qryWheels.Active := True;

qryBear.Active := True;

qryTracks.Active := True;

end;

procedure TForm2.FormCreate(Sender: TObject);

begin

strRidingStyles := TStringList.Create;

end;

end.

unit Unit3;

{$mode objfpc}{$H+}

{$GOTO ON}

interface

uses

Classes, SysUtils, sqldb, DB, mssqlconn, FileUtil, Forms,

Controls, Graphics, Dialogs, StdCtrls, ExtCtrls, Unit4;

type

{ TForm3 }

TForm3 = class(TForm)

Brdname2: TLabel;

Brdname3: TLabel;

DataSource1: TDataSource;

GroupBox1: TGroupBox;

grpskate1: TGroupBox;

grpskate2: TGroupBox;

grpbox: TGroupBox;

Image1: TImage;

Brdname1: TLabel;

Image2: TImage;

Image3: TImage;

Image4: TImage;

Image5: TImage;

Image6: TImage;

Image7: TImage;

lbldskr1: TLabel;

lbldskr2: TLabel;

lbldskr3: TLabel;

Lblabout: TLabel;

Lblabout1: TLabel;

Lblabout2: TLabel;

MSSQLConnection1: TMSSQLConnection;

PnlNext: TPanel;

Qryinfo: TSQLQuery;

qryonepage: TSQLQuery;

SQLTransaction1: TSQLTransaction;

procedure FormCreate(Sender: TObject);

procedure FormShow(Sender: TObject);

procedure Formopen(curlabel: Tlabel);

procedure PnlNextClick(Sender: TObject);

procedure LoadPicture(img: TImage);

procedure MouseEnter(vars: Tlabel);

procedure MouseLeave(vars: Tlabel);

procedure PnlNextMouseEnter(Sender: TObject);

procedure PnlNextMouseLeave(Sender: TObject);

private

{ private declarations }

public

{ public declarations }

procedure LoadInfo;

end;

var

Form3: TForm3;

recstatus: integer;

mass\_skate\_id: array [1..3] of integer;

roll\_direction: boolean;

implementation

{$R \*.lfm}

procedure Tform3.LoadPicture(img: TImage);

var

image\_path: string;

begin

image\_path := qryinfo.FieldByName('Picture').AsString;

if (image\_path <> '') then

img.Picture.LoadFromFile(image\_path);

end;

procedure Tform3.LoadInfo;

var

counter: integer;

sql\_text: string;

begin

sql\_text := qryinfo.SQL.Text;

counter := recstatus;

case recstatus of

1 :

begin

lbldskr1.Caption := qryinfo.FieldByName('Description').AsString;

Brdname1.Caption := qryinfo.FieldByName('Skate\_Name').AsString;

LoadPicture(Image1);

Image1.Stretch:=true;

Image1.Proportional:=false;

mass\_skate\_id[counter] := qryinfo.FieldByName('Skate\_id').AsInteger;

Lblabout.Tag := mass\_skate\_id[counter];

Inc(counter);

end;

2 :

begin

lbldskr2.Caption := qryinfo.FieldByName('Description').AsString;

Brdname2.Caption := qryinfo.FieldByName('Skate\_Name').AsString;

LoadPicture(Image2);

Image2.Stretch:=true;

Image2.Proportional:=false;

mass\_skate\_id[counter] := qryinfo.FieldByName('Skate\_id').AsInteger;

Lblabout1.Tag := mass\_skate\_id[counter];

Inc(counter);

end;

3 :

begin

lbldskr3.Caption := qryinfo.FieldByName('description').AsString;

Brdname3.Caption := qryinfo.FieldByName('Skate\_Name').AsString;

LoadPicture(Image3);

Image3.Stretch:=true;

Image3.Proportional:=false;

mass\_skate\_id[counter] := qryinfo.FieldByName('Skate\_id').AsInteger;

Lblabout2.Tag := mass\_skate\_id[counter];

end;

end;

end;

procedure TForm3.MouseEnter(vars: Tlabel);

begin

Vars.Font.Color := clRed;

end;

procedure TForm3.MouseLeave(vars: Tlabel);

begin

Vars.Font.Color := clAqua;

end;

procedure TForm3.PnlNextMouseEnter(Sender: TObject);

begin

PnlNext.Color:=$D355BA

end;

procedure TForm3.PnlNextMouseLeave(Sender: TObject);

begin

PnlNext.Color:=$00A333A6;

end;

procedure TForm3.PnlNextClick(Sender: TObject);

begin

if (not qryinfo.Active) then

qryinfo.Active := True;

if (not qryinfo.EOF) then

for recstatus := 1 to 3 do

begin

qryinfo.Next;

if (qryinfo.EOF) then

begin

qryinfo.First;

end;

loadinfo;

end;

end;

procedure TForm3.FormCreate(Sender: TObject);

begin

recstatus := 0;

roll\_direction := True;

if (recstatus = 0) then

begin

qryinfo.Active := False;

end;

end;

procedure TForm3.Formopen(curlabel: Tlabel);

var

lbl, i: integer;

img: string;

str: string;

s1: string;

begin

lbl := curlabel.Tag;

qryonepage.Active := False;

qryonepage.SQL.Clear;

qryonepage.SQL.Add('Exec dbo.Skates @idskate=:lbl');

qryonepage.ParamByName('lbl').AsInteger := lbl;

qryonepage.Active := True;

Form4.Name\_of\_deck.Caption := qryonepage.FieldByName('Skate\_name').AsString;

img := qryonepage.FieldByName('Picture').AsString;

Form4.Image1.Picture.LoadFromFile(img);

Form4.Image1.Stretch:=true;

Form4.Image1.Proportional:=false;

Form4.LblDescription.Caption := qryonepage.FieldByName('Description').AsString;

Form4.Lblstyles.Caption := qryonepage.FieldByName('Styles').AsString;

Form4.Lblnose.Caption :=

Form4.Lblnose.Caption + ' - ' + qryonepage.FieldByName('Nose').AsString;

Form4.Lbltail.Caption :=

Form4.Lbltail.Caption + ' - ' + qryonepage.FieldByName('Tale').AsString;

Form4.Lblwidth.Caption := ' - ' + qryonepage.FieldByName('Width').AsString;

Form4.Lbllength.Caption := ' - ' + qryonepage.FieldByName('Length').AsString;

Form4.Lbldcktp.Caption :=

Form4.Lbldcktp.Caption + ' - ' + qryonepage.FieldByName('Type\_of\_deck').AsString;

Form4.Lbltrsize.Caption :=

' - ' + qryonepage.FieldByName('Tracks\_Size').AsString;

Form4.Lbltrtype.Caption :=

Form4.Lbltrtype.Caption + ' - ' + qryonepage.FieldByName('Tracks\_Type').AsString;

Form4.Lblwhhrd.Caption :=

' - ' + qryonepage.FieldByName('Wh\_Hardness').AsString;

Form4.Lblwhsize.Caption := ' - ' + qryonepage.FieldByName('Wheel\_Size').AsString;

Form4.Lblbrname.Caption :=

Form4.Lblbrname.Caption + ' - ' + qryonepage.FieldByName('Bear\_name').AsString;

Form4.Lblmdof.Caption :=

Form4.Lblmdof.Caption + ' - ' + qryonepage.FieldByName('Madeof').AsString;

Form4.Lblwhlbs.Caption :=

' - ' + qryonepage.FieldByName('Wheelbase').AsString;

form4.showmodal;

end;

procedure TForm3.FormShow(Sender: TObject);

begin

PnlNextClick(pnlnext);

end;

end.

unit Unit4;

{$mode objfpc}{$H+}

interface

uses

Classes, SysUtils, sqldb, FileUtil, Forms, Controls, Graphics,

Dialogs, StdCtrls, ExtCtrls;

type

{ TForm4 }

TForm4 = class(TForm)

Grpparts: TGroupBox;

Image2: TImage;

Lbllength1: TLabel;

Lblnose1: TLabel;

Lblstyles: TLabel;

Lbltail1: TLabel;

Lbltrcks: TLabel;

Lbltrsize1: TLabel;

Lblwhhrd1: TLabel;

Lblwhlbs1: TLabel;

Lblwhsize: TLabel;

Lblbrname: TLabel;

Lblmdof: TLabel;

Lblwhhrd: TLabel;

LblWhls: TLabel;

Lbltrsize: TLabel;

Lbltrtype: TLabel;

Lblbr: TLabel;

Lblwhsize1: TLabel;

Lblwidth1: TLabel;

LblDescription: TMemo;

Parts: TLabel;

Grparam: TGroupBox;

grride: TGroupBox;

Lblwidth: TLabel;

Lbllength: TLabel;

Lblwhlbs: TLabel;

Lblnose: TLabel;

Lbltail: TLabel;

Lbldcktp: TLabel;

Riding\_styles: TLabel;

Grpdskr: TGroupBox;

Grpdeck: TGroupBox;

Image1: TImage;

Description: TLabel;

Name\_of\_deck: TLabel;

Params: TLabel;

private

{ private declarations }

public

{ public declarations }

end;

var

Form4: TForm4;

implementation

{$R \*.lfm}

end.