СОДЕРЖАНИЕ

[ОПИСАНИЕ ПРЕДМЕТНОЙ ОБЛАСТИ. ПОСТАНОВКА ЗАДАЧИ 2](#_Toc40338413)

[ВЫБОР СРЕДСТВ И МЕТОДОЛОГИЙ ПРОЕКТИРОВАНИЯ. ВЫБОР СУБД 3](#_Toc40338414)

[ПОСТРОЕНИЕ КОНЦЕПТУАЛЬНОЙ МОДЕЛИ 5](#_Toc40338415)

[ПРОЕКТИРОВАНИЕ ЛОГИЧЕСКОЙ СТРУКТУРЫ БД 6](#_Toc40338416)

[ВЫЯВЛЕНИЕ ПОЛНОГО ПЕРЕЧНЯ ОГРАНИЧЕНИЙ ЦЕЛОСТНОСТИ 8](#_Toc40338417)

[ПРОЕКТИРОВАНИЕ ФИЗИЧЕСКОЙ СТРУКТУРЫ БД 9](#_Toc40338418)

[ОРГАНИЗАЦИЯ ВВОДА ДАННЫХ В БД 10](#_Toc40338419)

[ОРГАНИЗАЦИЯ КОРРЕКТИРОВКИ БД 11](#_Toc40338420)

[РЕАЛИЗАЦИЯ ЗАПРОСОВ, ПОЛУЧЕНИЕ ОТЧЁТОВ 12](#_Toc40338421)

[РАЗРАБОТКА ИНТЕРФЕЙСА 14](#_Toc40338422)

[РЕАЛИЗАЦИЯ ПРОЕКТА В СУБД 15](#_Toc40338423)

[ВЫВОД 16](#_Toc40338424)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 17](#_Toc40338425)

# ОПИСАНИЕ ПРЕДМЕТНОЙ ОБЛАСТИ. ПОСТАНОВКА ЗАДАЧИ

На сегодняшний день школы, институты и другие учебные учреждения не предоставляют ученикам и студентам достаточного экономического образования, и вследствие чего, большинство выходят во взрослую жизнь финансово безграмотными. [1]

Чтобы не оказаться на грани банкротства в начала карьеры или в ее разгаре, необходимо иметь кое-какие привычки. По мнению экспертов, одной из важнейших привычек — это вести ежедневный учет расходов. Ежедневный учет расходов также необходим каждому из нас, как и ежедневные процедуры ухода за лицом, телом и волосами.

Ежедневные расходы можно вести в обыкновенном блокноте, ежедневнике, однако в этом случае, придется составлять отчеты самостоятельно, а также блокнот можно потерять или забыть. Именно поэтому было принято решение, перенести ежедневный учет расходов в электронный формат.

Основной задачей курсового проекта является автоматизация ведения учета расходов и отчета по ним.

В качестве основных запросов в БД можно выделить следующие запросы:

* CRD операции с расходами
* Чтение выборки расходов за определённый срок (год, месяц, день)
* Отчёт по расходам – сумма расходов с одинаковой категорией за определённый срок

# ВЫБОР СРЕДСТВ И МЕТОДОЛОГИЙ ПРОЕКТИРОВАНИЯ. ВЫБОР СУБД

В качестве основной СУБД была выбрана SQLite. Легко встраивается в приложения база данных. Так как это система базируется на файлах, то она предоставляет довольно широкий набор инструментов для работы с ней, по сравнению с сетевыми СУБД. При работе с этой СУБД обращения происходят напрямую к файлам, вместо портов и сокетов в сетевых СУБД. Именно поэтому SQLite очень быстрая, а также мощная благодаря технологиям обслуживающих библиотек. [2]

Преимущества SQLite:

* Файловая структура — вся база данных состоит из одного файла, поэтому её очень легко переносить на разные машины
* Используемые стандарты — хотя, может показаться, что эта СУБД примитивная, но она использует SQL. Некоторые особенности опущены (RIGHT OUTER JOIN или FOR EACH STATEMENT), но основные все - таки поддерживаются
* Отличная при разработке и тестировании — в процессе разработки приложений часто появляется необходимость масштабирования. SQLite предлагает всё что необходимо для этих целей, так как состоит всего из одного файла и библиотеки, написанной на языке C.

Несомненно, данную СУБД стоит использовать во встроенных приложениях — если нет необходимости в простом переносе приложения и не важна масштабируемость. Например, однопользовательские приложения, мобильные приложения или игры, но для данного курсового проекта она подходит, так как нет необходимости вносить сотни тысяч записей одновременно с разных устройств. Однако при масштабируемости, возможен быстрый переход на PostgreSQL из - за использования Django ORM.

Django ORM позволяет связывать базы данных с концепциями объектно-ориентированных языков программирования, таких как Python, создавая «виртуальную объектную базу данных». [3]

# ПОСТРОЕНИЕ КОНЦЕПТУАЛЬНОЙ МОДЕЛИ

Для реализации задуманного проекта, необходимо построить концептуальную модель базы данных, так как качественно спроектированная база данных необходима приложению так же, как и прочный фундамент необходим дому.

Сперва необходимо построить модель, которая свяжет таблицу «Расходы» и таблицу «Категория». Они будут находиться в связи 1 ко многу, так, что один расход может иметь только одну категорию, а категория может иметь много расходов. Модель изображена на рисунке 1. [4]



Рисунок 1 – Модель «Расходы» - «Категория»

Для того, чтобы связать таблицы «Пользователь» и «Расходы», потребуется связь 1 ко многим, потому что у одного пользователя может быть много расходов. Модель изображена на рисунке 2.

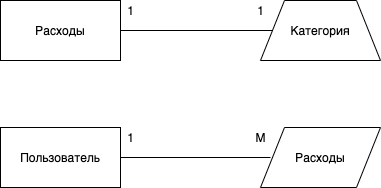


Рисунок 2 – Модель «Пользователь» - «Расходы»

# ПРОЕКТИРОВАНИЕ ЛОГИЧЕСКОЙ СТРУКТУРЫ БД

Для создания качественной базы данных, необходимо спроектировать ее логическую структуру.

Для начала определяются требуемые поля:

* Email
* Имя
* Пароль
* Баланс
* Сумма расхода
* Категория расхода
* Дата расхода
* Иконка расхода

Далее необходимо разделить поля на таблицы и определить типы и размеры полей [5]:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Таблица 1 – Пользователь | |  | Таблица 2 - Расходы | |  |
| Пользователь | |  | Расходы | |  |
| Email | varchar(254) |  | Сумма | integer |  |
| Имя | varchar(150) |  | Категория | Integer(FK) |  |
| Пароль | varchar(128) |  | Дата | date |  |
| Баланс | integer |  | Пользователь | Integer(FK) |  |
|  |  |  |  |  |  |
| Таблица 3 - категория | |  |  |  |  |
| Категория |  |  |  |  |  |
| Название | varchar(100) |  |  |  |  |
| Иконка | varchar(100) |  |  |  |  |

Следующим этапом будет построение базовой модели и реализация её в СУБД. Базовая модель представлена на рисунке 3. [6]



Рисунок 3 – Базовая модель в СУБД

# ВЫЯВЛЕНИЕ ПОЛНОГО ПЕРЕЧНЯ ОГРАНИЧЕНИЙ ЦЕЛОСТНОСТИ

Исходя из описания предметной области, выявлено одно ограничение:

* Расходы и баланс можно указывать только целыми числами.

Основываясь на выявленном ограничении, необходимо предусмотреть возможность контролировать данные, вводимые пользователем.

Для валидации данных, вводимых пользователем, более производительным вариантом будет валидация на стороне клиента, однако во избежание взломов клиентской стороны, валидация необходима и на стороне сервера.

Таким образом, пользователю разрешено вводить в поля расходов и баланса только целые числа, а на стороне сервера, все числа преобразовываются в целые, а также не целые числа не запишутся в БД, так как поле установлено целочисленное.

Проверка на целое число на стороне клиента производится путём навешивания обработчика на событие keypress, который пропускает только целые числа в поле ввода, используя регулярное выражение /[^0-9]/ig.

# ПРОЕКТИРОВАНИЕ ФИЗИЧЕСКОЙ СТРУКТУРЫ БД

Для начала необходимо заполнить таблицы категорий и добавить администратора.

Администратор создаётся командой createsuperuser или другим администратором через административную панель.

Данные в таблицу категорий заносятся администратором через административную панель. Данные таблицы категорий представлены в таблице 4.

|  |  |  |
| --- | --- | --- |
| Таблица 4 - категории | | |
| id | category | icon |
| 1 | Развлечения | fas fa-gamepad |
| 2 | Транспорт | fas fa-bus |
| 3 | Продукты | fas fa-utensils |
| 4 | Разное | fas fa-puzzle-piece |

# ОРГАНИЗАЦИЯ ВВОДА ДАННЫХ В БД

Ввод данных происходит через специальные формы на странице.

Сперва данные вводятся пользователем в форму, которая проверяет их валидность, затем данные отправляются на сервер в JSON формате, сервер принимает их и тоже валидирует и в последствии сохраняет в базу данных.

# ОРГАНИЗАЦИЯ КОРРЕКТИРОВКИ БД

Для того, чтобы в базу данных не попадали грязные данные, такие как HTML вёрстка, SQL инъекции и так далее, необходимо позаботиться о том, чтобы формы ввода данных не пропускали такие данные, а также, если они всё таки попали на сервер, то сервер должен их тоже провалидировать.

Достигается данная валидация путём использования различных методов таких как регулярные выражения и подобные. [7]

# РЕАЛИЗАЦИЯ ЗАПРОСОВ, ПОЛУЧЕНИЕ ОТЧЁТОВ

В таблице 5 представлены SQL запросы для выполнения необходимых операций.

|  |  |
| --- | --- |
| Таблица 5 - запросы |  |
| Назначение | Запрос |
| Добавление расхода | INSERT INTO "costs\_cost" ("owner\_id", "cost", "category\_id", "created\_at")  VALUES (1, 100, 1, '2020-05-02') |
| Удаление расхода | DELETE  FROM "costs\_cost"  WHERE "costs\_cost"."id" IN (2) |
| Выборка по дате | SELECT "costs\_cost"."id",  "costs\_cost"."owner\_id",  "costs\_cost"."cost",  "costs\_cost"."category\_id",  "costs\_cost"."created\_at"  FROM "costs\_cost"  WHERE ("costs\_cost"."owner\_id" = 1 AND "costs\_cost"."created\_at" LIKE '%2020-05-14%') |
| Отчёт за дату | SELECT "costs\_cost"."category\_id",  SUM("costs\_cost"."cost") AS "total\_cost"  FROM "costs\_cost"  WHERE ("costs\_cost"."owner\_id" = 1 AND "costs\_cost"."created\_at" LIKE '%2020-05-02%' ESCAPE '\')  GROUP BY "costs\_cost"."category\_id"  ORDER BY "total\_cost" DESC |
| Увеличение баланса | UPDATE "user\_extension\_user"  SET  "balance" = 10001  WHERE "user\_extension\_user"."id" = 1 |
| Получение расходов за текущий месяц | SELECT SUM("costs\_cost"."cost") AS "month\_cost"  FROM "costs\_cost"  WHERE ("costs\_cost"."owner\_id" = 1 AND "costs\_cost"."created\_at" LIKE '%2020-05%' ESCAPE '\') |
| Получение пользователя | SELECT "user\_extension\_user"."id",  "user\_extension\_user"."username",  "user\_extension\_user"."balance"  FROM "user\_extension\_user"  WHERE "user\_extension\_user"."id" = 1 |
| Регистрация пользователя | INSERT INTO "user\_extension\_user" ("password", "last\_login", "is\_superuser", "username", "first\_name", "last\_name", "email", "is\_staff", "is\_active", "date\_joined", "balance")  VALUES ('pbkdf2\_sha256$180000$8DU9LuhEv6aE$DEz64nkwVgpFzjbfNxBq+sMSLsfFkRFBXSx89L5ZEmE=', NULL, 0, login, '', '', email@gmail.com', 0, 1, '2020-05-02 06:43:20.912016', 0) |

# РАЗРАБОТКА ИНТЕРФЕЙСА

Для того, чтобы пользователь мог выполнять необходимые ему операции с БД, был создан пользовательский интерфейс, отражающий весь функционал системы.

На странице входа есть поля ввода логина и пароля для входа, а также ссылка на регистрацию, где предоставлены поля почты, логина и пароля для регистрации.

На главной странице в верхнем меню предоставлена возможность просмотра баланса и сумма расходов за текущий месяц, ссылка на отчёт, а также выход из системы. На левой карточке предоставлена возможность создать расход и выведен список расходов за сегодня, а при выборе даты на правой карточке с календарём, на левой появятся расходы на выбранную дату.

На странице отчёта, на левой карточке предоставлен отчёт, за выбранный период на правой карточке с календарём.

# РЕАЛИЗАЦИЯ ПРОЕКТА В СУБД

Итогом проделанной работы можно считать выполненный планируемый функционал создаваемой системы. Однако в будущем требуется уделить внимание улучшению пользовательского интерфейса, а именно график отчёта. В качестве дополнительных возможностей, можно предоставить пользователю создавать категории, необходимые именно ему.

# ВЫВОД

В результате курсового проекта было реализовано Single Page Application «Cost Control». Оно позволит пользователям начать следить за своей финансовой грамотностью и расходами, ведь в нём были реализованы:

* CRD операции с расходами
* Чтение выборки расходов за определённый срок (год, месяц, день)
* Отчёт по расходам – сумма расходов с одинаковой категорией за определённый срок

Приложение спроектировано таким образом, что возможно дальнейшая поддержка и расширение. Примерами расширения может быть такой функционал как: диаграммы и графики отчёта, изменение расхода, а также возможность создаваться собственные категории и так далее.

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Богатый Папа, Бедный Папа [электронный ресурс] - режим доступа: https://books.google.ru/books/about/%D0%91%D0%BE%D0%B3%D0%B0%D1%82%D1%8B%D0%B9\_%D0%BF%D0%B0%D0%BF%D0%B0\_%D0%B1%D0%B5%D0%B4%D0%BD%D1%8B%D0%B9\_%D0%BF.html?id=vy\_lCwAAQBAJ&printsec=frontcover&source=kp\_read\_button&redir\_esc=y#v=onepage&q&f=false
2. SQLite [электронный ресурс] - режим доступа: https://habr.com/ru/post/149356/
3. Docs Django [электронный ресурс] - режим доступа: https://docs.djangoproject.com/en/3.0/topics/db/queries/
4. ER-диаграммы [электронный ресурс] - режим доступа: lucidchart.com/pages/ru/erd-диаграмма
5. Model field reference [электронный ресурс] - режим доступа: <https://docs.djangoproject.com/en/3.0/ref/models/fields/>
6. Схема БД [электронный ресурс] - режим доступа: <https://eax.me/database-design/>
7. Безопасность веб-приложения Django [электронный ресурс] - режим доступа: https://developer.mozilla.org/ru/docs/Learn/Server-side/Django/web\_application\_security