ГУАП

КАФЕДРА № 43

ОТЧЕТ   
ЗАЩИЩЕН С ОЦЕНКОЙ

ПРЕПОДАВАТЕЛЬ­­

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ст. преп. |  |  |  | Н.А. Соловьева |
| должность, уч. степень, звание |  | подпись, дата |  | инициалы, фамилия |

|  |
| --- |
| ОТЧЕТ О ЛАБОРАТОРНОЙ РАБОТЕ №3 |
| ТЕХНОЛОГИЯ СОЕДИНЕНИЯ С УДАЛЕННЫМ ИСТОЧНИКОМ ДАННЫХ. ОСНОВНЫЕ ОБЪЕКТЫ ПРОВАЙДЕРА ДАННЫХ: CONNECTION, COMMAND. |
| по дисциплине: КОНСТРУИРОВАНИЕ ИНФОРМАЦИОННЫХ СИСТЕМ |
|  |
|  |

РАБОТУ ВЫПОЛНИЛ

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| СТУДЕНТ ГР. № | 4631 |  |  |  | С.А. Гришин |
|  |  |  | подпись, дата |  | инициалы, фамилия |

Санкт-Петербург 2018

**Введение**

Для хранения информации, была выбрана была выбрана реляционная база данных SQlite у неё есть несколько особенностей: движок SQLite компонуется напрямую с программой, и становится её составной частью, протоколом обмена являются вызовы функций (API) библиотеки SQLite. Такой подход уменьшает накладные расходы, время отклика и упрощает программу.   
В качестве языка разработки был выбран Golang.  Go — компилируемый многопоточный язык программирования, разработанный внутри компании Google.

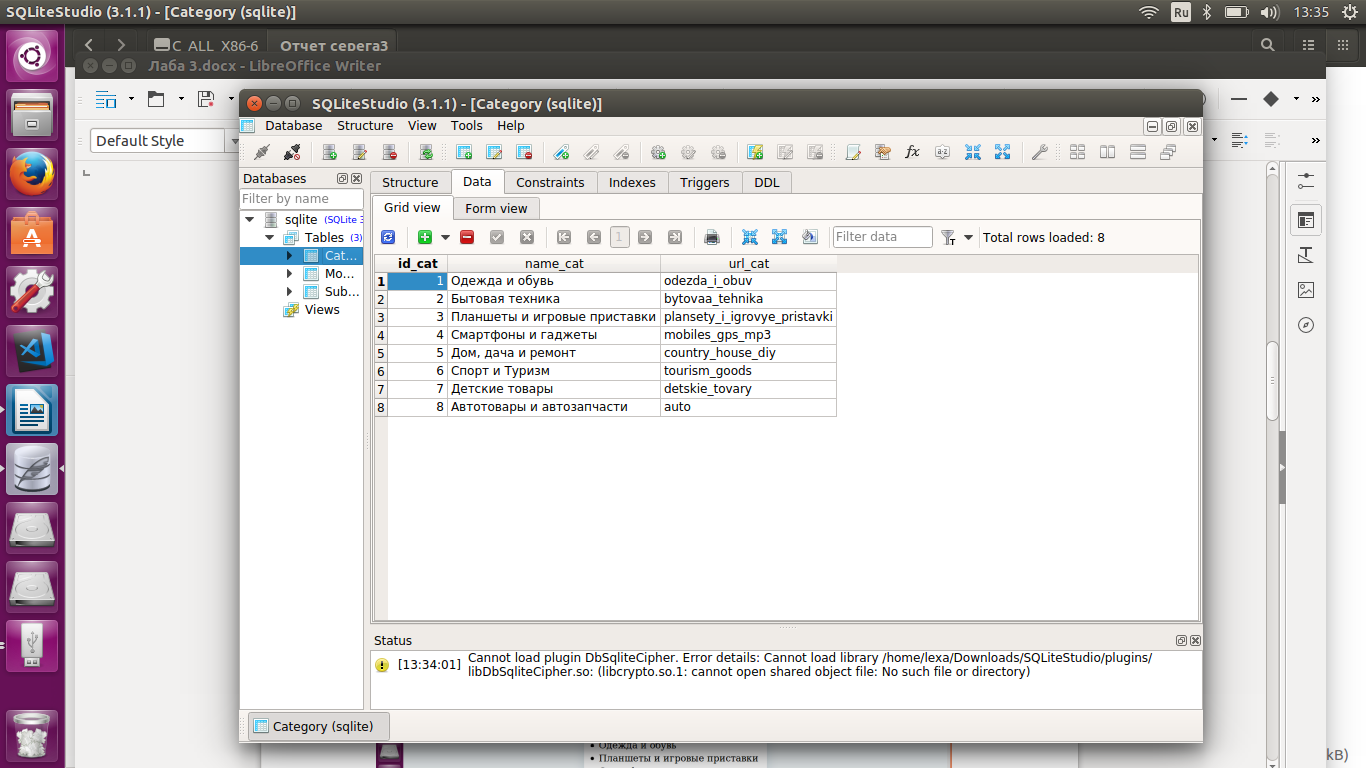
Язык Go разрабатывался как язык программирования для создания высокоэффективных программ, работающих на современных распределённых системах и многоядерных процессорах. Он может рассматриваться как попытка создать замену языкам Си и C++. 

Этот язык является относительно новым в 2007 была начата его разработка а первый релиз состоялся в 2009. Многое он вобрал в себя из уже существующих языков, и при определении вектора развития во многом опирается на опыт "старших (Java, C++) ".

Но при этом его определяющей идеей является "простота и читаемость" кода.   
Больше всего он подходит для написания утилит и backend`ов. Это достигается за счет следующих пунктов:

* Go — язык со строгой статической типизацией.
* Строковый тип со встроенной поддержкой юникода.
* Использование динамических массивов, хэш-таблиц, срезов (слайсов), вариант цикла для обхода коллекции
* Автоматическое управление памятью со сборщиком мусора.
* Средства параллельного программирования: встроенные в язык потоки (go routines), взаимодействие потоков через каналы и другие средства организации многопоточных программ.

Это делает его подходящим для нашей текущей задачи.

**Слепок содержимого Базы данных**

**Листинг программы**

package main

import (

    "database/sql" // Интерфейс для работы со SQL-like БД

    "html/template" // Шаблоны для выдачи html страниц

    "log" // Вывод информации в консоль

    "net/http" // Для запуска HTTP сервера

    \_ "github.com/mattn/go-sqlite3" // Драйвер для работы со SQLite3

)

// DB указатель на соединение с базой данных

var DB \*sql.DB

// Реализация обработчика запроса

func handler(w http.ResponseWriter, r \*http.Request) {

    // Выполнение запроса к базе данных

    rows, err := DB.Query(`SELECT name\_cat FROM Category`)

    if err != nil {

        log.Fatal(err)

    }

    // Чтение шаблона из файла

    tmpl, \_ := template.ParseFiles("tmpl/index.html")

    // Создание массива для снятия слепка с базы данных,

    names := []string{}

    // Итерируемся по всем строкам, который вернул запрос SQLite

    for rows.Next() {

        var temp string

        rows.Scan(&temp)

        // Записывает возвращенные данные в слепок

        names = append(names, temp)

    }

    // Вписываем данные в шаблон HTML страницы, дл отдачи пользователю

    tmpl.Execute(w, names)

}

func main() {

    var err error

    // Открытие соединения с БД SQLite3.db

    DB, err = sql.Open("sqlite3", "./sqlite.db")

    // Проверка установки соединения

    if err != nil {

        log.Fatal(err)

    }

    // Закрытие соединения с БД по выходу из функции main

    defer DB.Close()

    // CREATE TABLE Category (

    //  id\_cat INTEGER PRIMARY KEY AUTOINCREMENT,

    //  name\_cat TEXT NOT NULL

    //                   UNIQUE,

    //  url\_cat TEXT NOT NULL

    //                   UNIQUE

    // );

    // CREATE TABLE Subcategory (

    //  id\_subc INTEGER PRIMARY KEY AUTOINCREMENT,

    //  id\_cat REFERENCES Category (id\_cat),

    //  name\_subc TEXT NOT NULL

    //                   UNIQUE,

    //  url\_subc TEXT NOT NULL

    //                   UNIQUE

    // );

    // CREATE TABLE Model (

    //  id\_mod INTEGER PRIMARY KEY AUTOINCREMENT,

    //  name\_mod TEXT NOT NULL

    //                      UNIQUE ON CONFLICT ROLLBACK,

    //  price INTEGER,

    //  country TEXT,

    //  manufacture TEXT,

    //  weight INTEGER

    // );

    // Установка обработчика запроса по данному запросу

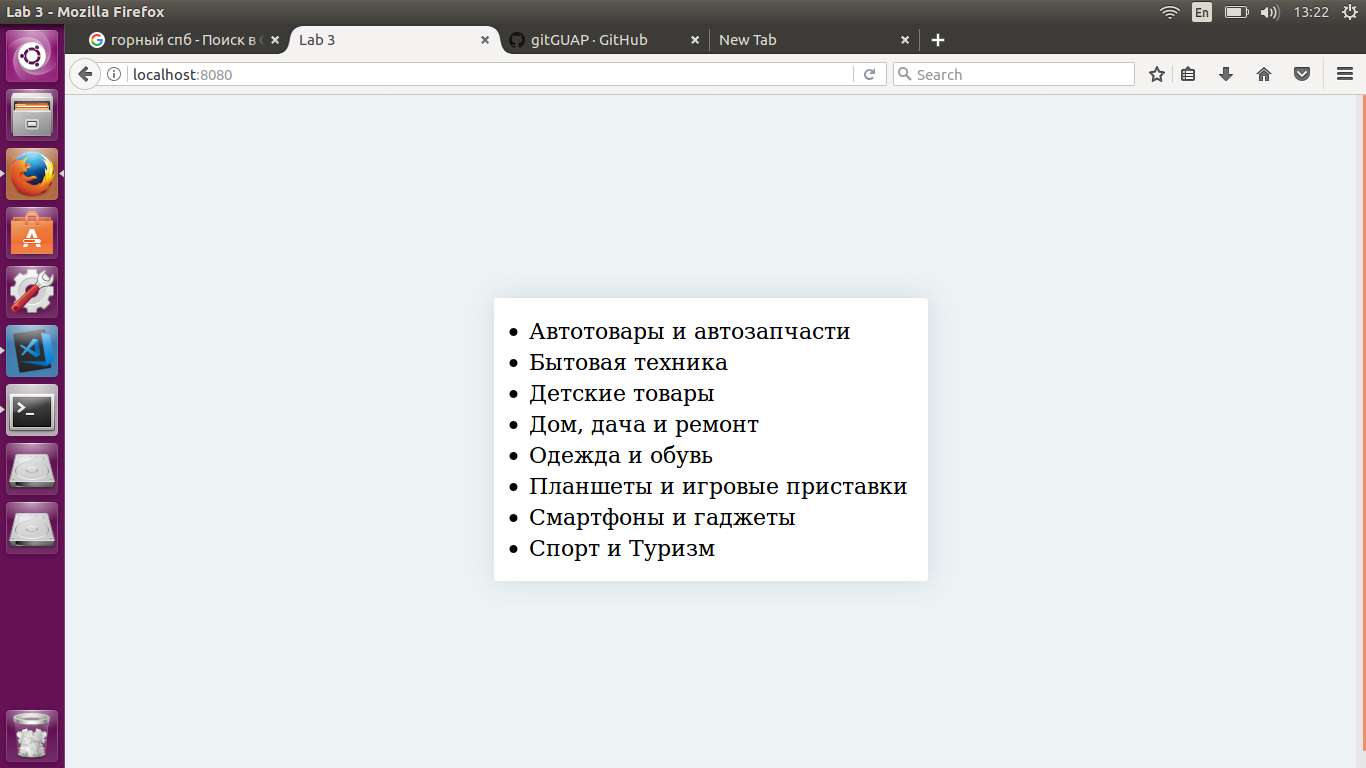
    http.HandleFunc("/", handler)

    log.Println("Listening...")

    // Запуск локального сервека на 8080 порту

    log.Fatal(http.ListenAndServe(":8080", nil))

}

**Пример выполнения**