МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«МОСКОВСКИЙ АВИАЦИОННЫЙ ИНСТИТУТ
(национальный исследовательский университет)»

Институт (Филиал) № 8 «Компьютерные науки и прикладная математика»

Кафедра 806

Группа М8О-407Б-19 **Направление подготовки** 01.02.03 «Прикладная математика и информатика»

Профиль Информатика

Квалификация: бакалавр

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА БАКАЛАВРА

на тему «Сервис поиска, классификации и обработки тематических изображений в интернете»

Автор ВКРБ: Волков Матвей Андреевич (_____)

Руководитель: Филимонов Николай Сергеевич (_____)

Консультант: (_____)

Консультант: (_____)

Рецензент: (_____)

К защите допустить

Заведующий кафедрой № 806 «Вычислительная математика и программирование» Крылов Сергей Сергеевич (_____)

____ мая 2023 года

Москва 2023

РЕФЕРАТ

Выпускная квалификационная работа бакалавра состоит из 18 страниц, 6 рисунков, 4 использованных источников.

КЛЮЧЕВЫЕ СЛОВА, КЛЮЧЕВЫЕ СЛОВА, КЛЮЧЕВЫЕ СЛОВА, КЛЮЧЕВЫЕ СЛОВА, КЛЮЧЕВЫЕ СЛОВА

С другой стороны рамки и место обучения кадров представляет собой интересный эксперимент проверки существенных финансовых и административных условий. Товарищи! сложившаяся структура организации позволяет выполнять важные задания по разработке существенных финансовых и административных условий. Таким образом укрепление и развитие структуры способствует подготовки и реализации дальнейших направлений развития. Таким образом укрепление и развитие структуры способствует подготовки и реализации соответствующий условий активизации.

Не следует, однако забывать, что постоянный количественный рост и сфера нашей активности обеспечивает широкому кругу (специалистов) участие в формировании позиций, занимаемых участниками в отношении поставленных задач. С другой стороны новая модель организационной деятельности требуют определения и уточнения систем массового участия. Не следует, однако забывать, что постоянное информационно-пропагандистское обеспечение нашей деятельности представляет собой интересный эксперимент проверки позиций, занимаемых участниками в отношении поставленных задач. Задача организации, в особенности же реализация намеченных плановых заданий требуют от нас анализа существенных финансовых и административных условий.

СОДЕРЖАНИЕ

ТЕРМИНЫ И ОПРЕДЕЛЕНИЯ

В настоящей выпускной квалификационной работе бакалавра применяют следующие термины с соответствующими определениями:

HTML — стандартизированный язык гипертекстовой разметки документов для просмотра веб-страниц в браузере.

HTTP — протокол передачи гипертекста, изначально в виде гипертекстовых документов в формате HTML, в настоящее время используется для передачи произвольных данных.

DOM-дерево — структурное представление HTML файла.

API — описание способов взаимодействия одной компьютерной программы с другими.

Метод HTTP — последовательность из любых символов, кроме управляющих и разделителей, указывающая на основную операцию над ресурсом. Обычно метод представляет собой короткое английское слово, записанное заглавными буквами.

HTTP запрос — сигнал серверу через http метод, всегда требующий какого-то ответа от сервара.

Хендлер — функция, которую выполняет веб-приложение, когда был сделан соответствующий http запрос.

База данных — набор информации, которая хранится упорядоченно в электронном виде.

Сервер — сервером в веб приложении называют программу, которая занимается обработкой запросов с клиентской части приложения. В основном на сервере происходит вся логика работы веб приложения. Обычно на серверной части запускается база данных.

Клиент — приложение, которое путем отправки запросов на сервер способно визуализировать полученную информацию. Обычно приложение клиент общается с пользователем. При этом не присутствует никакой логики, связанной с основным приложением. Например, браузер является клиентом.

Библиотека — сборник подпрограмм или объектов, используемых для разработки программного обеспечения.

Парсинг — это автоматизированный сбор и структурирование информации с сайтов при помощи программы или сервиса.

jQuery — набор функций JavaScript, фокусирующийся на взаимодействии

JavaScript и HTML. Библиотека jQuery помогает легко получать доступ к любому элементу DOM, обращаться к атрибутам и содержимому элементов DOM, манипулировать ими.

Конститентность — согласованность данных друг с другом, целостность данных, а также внутренняя непротиворечивость.

Миграция базы данных — переход от одной структуры базы данных к другой без потери косистентности.

Пайплай — конвеер данных, поступающий из одной логической программы другой.

Чистая архитектура — понятие при конструировании микросервиса, созданное для разделения различных концепций, путем написания кода на нескольких уровнях с четким правилом, которое позволяет создать тестируемый и поддерживаемый проект.

GO — язык программирования, созданный компанией гугл преимущественно для создания бекэнд сервисов.

Модуль — называется пакет в проекте языка GO, который может быть переиспользован в других приложениях или проектах, как внешняя библиотека. Был добавлен в версии 1.11.

Пакет — понятие в языке GO, обозначающее коллекцию исходного кода в одной дериктории скомпилируемых вместе. Пакет также является подключаемым модулем.

Реверс-инженеринг — исследование некоторого готового устройства или программы, а также документации на него с целью понять принцип его работы. Также известно, как «обратная разработка».

SQL-инъекция — один из распространённых способов взлома сайтов и программ, работающих с базами данных, основанный на внедрении в запрос произвольного SQL-кода.

ПЕРЕЧЕНЬ СОКРАЩЕНИЙ И ОБОЗНАЧЕНИЙ

В настоящей выпускной квалификационной работе бакалавра применяют следующие сокращения и обозначения:

SQL — Structured Query Language

API — Application Programming Interface

БД — база данных

HTTP — HyperText Markup Language

DOM — Document Object Model

ЯП — язык программирования

PDF — Portable Document Format

ID — Identification

JSON — Javascript object notation

RegExp — Regular Expression

CSS — Cascading Style Sheets

JS — JavaScript

ВВЕДЕНИЕ

В данном документе описываются основные моменты работы с шаблоном. Приведены примеры ко всему, что может понадобится при написании отчета, даны пояснения касаемо особенностей оформления. Вопросы содержания не рассматриваются, обращайтесь к шаблону, предоставленному институтом (ссылка в README).

1 ТЕОРЕТИЧЕСКАЯ ЧАСТЬ

1.1 Обоснование библиотек

В микросервисе были использованы внешние модули для упрощения частоиспользуемых действий в коде. Каждая из них представляет собой очень полезную функциональность, которая будет перечислена ниже с обоснованием использования.

1.1.1 pdfcpu

Pdfcpu [1] — пакет, написанный на ЯП GO, яляющийся мощным сборником pdf файлов. Представляет интферфейс для создания сложноструктурированных pdf файлов.

В данном пакете возможно создавать абсолютно любые pdf файлы с различными метаданными документа, начиная с содержания, заканчивая указыванием автора написания документа. Библиотека способна также использовать различные картинки различных разрешений, позволяя менять их размер, цвет, прозрачность и другие параметры.

Пакет был взят в использование потому, что предоставляется разработчиками удобный и понятный интферфейс для собирания большого количество картинок в pdf файл. Также модуль отличается своей быстротой исполнения поставленных задач. Рисунок 1 может продемонстрировать простоту использования пакета.

```
1  func CreatePDFFromImagesDir(imagesDirPath string, outputPath
       string) error {
2    imagesPath, err := GetImagesPathStr(imagesDirPath)
3    if err != nil {
4      return errors.Wrap(err, "something␣wrong␣with␣creating␣
       images␣path")
5    }
6    err = api.ImportImagesFile(imagesPath, outputPath, nil, nil)
7    if err != nil {
8      return errors.Wrap(err, "pdf␣creation␣filure")
9    }
10   return nil
11 }
```

Рисунок 1 – Демонстрация работы интерфейса pdfcpu

1.1.2 goquery

Goquery [2] — пакет, написанный на ЯП GO, который способен выполнять задачи библиотеки jQuery [3].

Данный пакет используется для того, чтобы парсить DOM дерево полученного HTML файла. С ним возможно быстро и удобно получить нужную информацию с любого сайта. После реверс-инженеринга, благодаря этому покету, быстро и просто достается информация о картинках, классифицируется на разделы и подразделы (например, группировка на главы).

```
1    func (mlb mangaLibController) getChapterID(doc
       *goquery.Document) (string, error) {
2      result, ok := doc.Find("#comments").Attr("data-post-id")
3      if !ok {
4        return "", errors.Wrap(customerrors.ErrEmptyAttr,
       "mangalib:␣data-post-id")
5      }
6      return result, nil
7    }
8
```

Рисунок 2 – Демонстрация работы пакета goquery

Как видно из рисунка 2, видно, что используются команды jQuery для получения информации о главе. Далее эта глава используется дальше, чтобы достать картинки этой главы.

1.1.3 jet

Go-jet [4] — пакет, написанный на ЯП GO, являющийся конструктором SQL запросов в БД.

С помощью этой библиотеки возможно собирать любые SQL запросы, избегая уязвимость — SQL-инъекцию. Этот пакет подключается к БД и генерирует код на языке GO, при помощи которого можно собирать различные SQL запросы. От инъекции помогает избавиться тот факт, что абсолютно весь запрос делается при помощи конструктора, в процессе сбора которого есть постоянные проверки на соответствие данных, поступающих в процесс создания запроса. Также стоит заметить, что сама возможность собирать запрос из различных кусочков, может очень сильно помочь при составлении денамического запроса, который так или иначе зависит от поступивших на вход параметров.

Именно по этой причине был выбран данный пакет. Как можно видеть из рисунка 3, собирается SQL запрос в зависимости от пришедших на вход параметров. Таким образом, условная секция запроса может варьироваться.

```
1    func (perC personController) GetEmailByID(ctx
     context.Context, person domain.PersonInfo) (string, error){
2       selectStmt := table.Persons.SELECT(table.Persons.Email)
3       var whereStmt postgres.BoolExpression
4
5       personID, _ := uuid.Parse(person.PersonID)
6       switch {
7       case personID != uuid.UUID{} && personID != uuid.Nil:
8         whereStmt = table.Persons.ID.EQ(postgres.UUID(personID))
9       case person.TelegramID > 0:
10        whereStmt =
     table.Persons.TelegramID.EQ(postgres.Int64(person.TelegramID))
11      default:
12        return "", customerrors.ErrEmailsNotFound
13      }
14      stmt, args := selectStmt.WHERE(whereStmt).Sql()
15
16      ...
17
18      return email, nil
19   }
20
```

Рисунок 3 – Демонстрация работы пакета jet

1.2 Устройство сайтов

Предметом исследования диплома был ограниченный список сайтов, участвующие в сборе картинок с последующей их сортировкой и группировкой в pdf файл для последующего их удобного просмотра или прочтения. Суть изучения состояло в том, чтобы произвести реверс-инженеринг сайта, понять как он работает. Затем использовать полученные знания в поиске нужной мне информации, путем парсинга страницы сайта, разбиения полученной структуры сайта на составляющие, с последующей структуризацией в нужный мне формат для правильной отдачи итогого результата.

Абсолютно любой сайт состоит из 3-х компонент:

а) HTML

б) CSS

в) JavaScript

1.2.1 HTML

В данной компоненте описывается различная информация о странице. Будь то текст статьи, картинки и прочее. Затем, браузер, применяя свои движки визуализации, преображает инфорацию, содержащуюся в компоненте, в понятную для человека визуализированное представление. HTML имеет свою структуру, называемой DOM деревом. Эта структура по сути своей является набором различных тэгов. Для идеального примера можно сказать, что HTML похож на LATEX. Таким образом, HTML — это то, откуда будет получаться нужная информация.

Как было сказано ранее, HTML состоит из тэгов. Есть следующие виды тэгов:

а) Двойные

б) Ординарные

Двойной тэг отличается от ординарного тем, что у двойного тэга есть закрывающий тэг. Между открывающим тэгом и закрывающим пишется текстовая информация. Например, на рисунке 4 продемонстрирован двойной тэг, обозначающий параграф. Обычно такой тэг используется в различных статьях. На рисунке 5 продемонстрирован пример ординарного тэга, обозначающий картинку. Когда браузер видит этот тэг, он понимает, что перед ним картинка. За кадром происходит запрос на сервер по ссылке, расположенной в мета информации, с последующим отображением изображения.

```
1         <p class="paragraf">Текст параграфа</p>
2
```

Рисунок 4 – Пример двойного тэга

```
1         <img src="picture.img" alt="Cup␣of␣Kitty">
2
```

Рисунок 5 – Пример ординарного тэга

На картинках 4 и 5 были продемонстрированы тэги, которые требуют

от браузера некоторых действий. Но, как было выше отмечено, HTML очень похож на LATEX. И похож он тем, что в основном страницы сайта нужно верстать. Поэтому люди придумали тэг, который не требует никаких действий со стороны браузера, но при этом будет как-то объединять информацию.

На рисунке 6 изображен тэг div. Этот тэг нужен исключительно для того, чтобы как-то объединять информацию, присваивая ей какую-то общую метаинформацию, например, класс.

```
1  <div class="pritty"></div>
2
```

Рисунок 6 – Тэг div

Из совокупности тэгов состоит страница сайта. А структура, которая в конечном итоге образуется, называется DOM деревом.

1.2.2 CSS

Эта компонента, в свою очередь, отвечает за то как будет выглядить информация, содержащаяся в HTML. По сути своей, CSS — оформление страницы. После появления данной компоненты, сайты в интернете стали выглядить так, как мы их сейчас видим.

Для формирования различных стилей, используется метаинформация тэгов. Удобнее всего для таких нужд оказалось использовать тэг, изображенный на рисунке 6, так как этот тэг как никто лучше служит для объединения информации, которую можно как-то по-особенному расположить и украсить.

Именно поэтому на большинстве сайтов можно увидеть именно этот тэг.

1.2.3 JavaScript

Эта компонента отвечает за логику на страницах. Логика может использоваться при анимировании сайтов и общения с сервером. Общение с сервером нужно для того, чтобы возможно было у него получить какую-то информацию.

1.3 Методика парсинга

Понимание сайта — важная часть абсолютно любого парсинга данных. Как понятно из определения, необходимо собирать информацию. Для того, чтобы собирать информацию, необходимо знать как она расположена. Для того, чтобы понимать как расположена информация, необходимо исселодовать пути появления этой информации перед пользователем и выяснить откуда она берется.

Плавно мы подбираемся к тому, что знание устройство сайта, клиентской части и сервера — необходимы для ранее сказанного парсинга данных. Другими словами, для парсинга данных с сайта необходимо полностью понимать как тот устроен. Если не получается полностью изучить, то нужно понять что за данные нужны, чтобы в конечном итоге построить то, что нужно. Нужно цепляться буквально за любую крупицу данных, которая может так или иначе помочь.

1.3.1 Взаимодействие с сервером

Для поиска подобной крупицы информации нужно понимать, что ничего не происходит из ниоткуда. Везде есть какие-то следы. Они могут быть зашифрованы, или спрятаны за тонной других запросов. Но информация откуда-то получается. Нужно сделать оговорку, потому что есть сайты, которые не используют сервер для получения какой-либо информации, а она уже сразу закодирована в верстке сайта. Такими сайтами называются «визиткой». Как понятно из навзвания, такие сайты нужны для того, чтобы красиво продемонстрировать род деятильности, прорекламировать компанию, которую представляет тот или иной сотрудник и указать там контакты для сотрудничества. Обычно такие сайты одностроничные, под собой не имеют никакой логики. Такие сайты очень удобно парсить. Но чаще всего человек сталкивается с сложными сайтами со сложной логикой, где есть большая база пользователей и контета. Такие сайты просто обязаны обращаться за какой-то логикой на сервер.

Можно заметить, что в разделе1.2.3 упоминается JS — логическая сторона сайта. Почему не пользоваться этой замечательной компонетой сайта для различной логики? Дело, конечно же, в безопасности. Куда более

надежно будет положить какую-то информацию в переменную, а потом ее в будущем отобразить на странице при помощи, например, jQuery. Никто не хочет, чтобы алгоритмы обработки личной информации лежали перед всеми на видном месте, чтобы их можно было запустить и все расшифровать. Или как еще по-другому можно хранить миллионы миллиардов информации?

Итак, так как предмет исследования как раз идет за нужной информацией на сервер, необходимо понять из каких крупиц информации создаются следующие более сложные запросы на сервер.

2 ИСПОЛЬЗОВАНИЕ ТИТУЛЬНОЙ СТРАНИЦЫ

Заполнение полей показано в файле `main.tex`, приведу ещё раз здесь. Желательно заполнить все поля по образцу, чтобы не было проблем с тем, что что-то может быть не определено.

ЗАКЛЮЧЕНИЕ

Товарищи! рамки и место обучения кадров играет важную роль в формировании форм развития. Таким образом дальнейшее развитие различных форм деятельности играет важную роль в формировании дальнейших направлений развития. Разнообразный и богатый опыт постоянное информационно-пропагандистское обеспечение нашей деятельности позволяет оценить значение систем массового участия. Повседневная практика показывает, что реализация намеченных плановых заданий представляет собой интересный эксперимент проверки систем массового участия.

Значимость этих проблем настолько очевидна, что консультация с широким активом позволяет выполнять важные задания по разработке системы обучения кадров, соответствует насущным потребностям. Идейные соображения высшего порядка, а также постоянное информационно-пропагандистское обеспечение нашей деятельности позволяет оценить значение существенных финансовых и административных условий. Задача организации, в особенности же сложившаяся структура организации обеспечивает широкому кругу (специалистов) участие в формировании систем массового участия. Разнообразный и богатый опыт начало повседневной работы по формированию позиции представляет собой интересный эксперимент проверки форм развития.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. *GitHub*. Pdfcpu. — 2023. — URL: https://github.com/pdfcpu/pdfcpu (дата обращения 28.04.2023).

2. *GitHub*. Goquery. — 2023. — URL: https://github.com/PuerkitoBio/goquery (дата обращения 28.04.2023).

3. *Wikipedia*. jQuery. — 2023. — URL: https://ru.wikipedia.org/wiki/JQuery (дата обращения 28.04.2023).

4. *GitHub*. Go-jet. — 2023. — URL: https://github.com/go-jet/jet (дата обращения 29.04.2023).