Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное
учреждение высшего образования
«Пермский государственный аграрно-технологический университет
имени академика Д. Н. Прянишникова»

Беляков А.Ю.

# ПРОГРАММНАЯ ИНЖЕНЕРИЯ

Методические рекомендации для выполнения курсового проекта

*Пермь*
*ФГБОУ ВО Пермский ГАТУ*
*2025*

**УДК 004.43**
**ББК 32.973-018.1**
**Б 448**

*Рецензенты:*

Е.А. Муратова, к.э.н., доцент, заведующий кафедры информационных технологий и программной инженерии (ФГБОУ ВО Пермский ГАТУ)

А.А. Зорин, к.т.н., доцент, доцент кафедры информационных технологий и программной инженерии (ФГБОУ ВО Пермский ГАТУ)

**Б 448 Беляков А.Ю.**

Программная инженерия: методические рекомендации для выполнения курсового проекта / авт. А.Ю. Беляков; М-во науки и высшего образования РФ, федеральное гос. бюджетное образов. учреждение высшего образования «Пермский гос. аграрно-технолог. университет им. акад. Д.Н. Прянишникова». – Пермь: ФГБОУ ВО Пермский ГАТУ, 2025. – 24 с.

В методических рекомендациях для выполнения курсового проекта представлена постановка задачи на проектирование информационной системы и сформулированы требования к технической реализации проекта.

Методические рекомендации для выполнения курсового проекта предназначены для обучающихся очной и заочной форм обучения по направлению подготовки 09.03.03 Прикладная информатика.

**УДК 004.43**
**ББК 32.937-018.1**

Рекомендованы к изданию методической комиссией факультета экономики и информационных технологий ФГБОУ ВО Пермский ГАТУ, протокол № __ от __ декабря 202__ г.

## Содержание

## Введение

Методические рекомендации предназначены для выполнения курсового проекта по направлению подготовки 09.03.03 Прикладная информатика, направленность (профиль) «Прикладная информатика в экономике» по дисциплине «Программная инженерия». Существенную роль в освоении компетенций по дисциплине «Программная инженерия» играет отработка умений самостоятельного выполнения технических проектов автоматизации бизнес-процессов, что вызывает потребность в разработке методических рекомендаций.

Целью данных методических рекомендаций для выполнения курсового проекта является получение опыта подготовки технического задания на проектирование информационной системы, развитие умений и навыков разработки и тестирования информационной системы обучающимися, полученных при контактной работе с преподавателем и при изучении учебной, методической, справочной литературы.

Основной задачей методических рекомендаций для выполнения курсового проекта является расширение навыков самостоятельной профессиональной деятельности обучающимися и проверка знаний по программированию сложных информационных систем.

В методических рекомендациях для выполнения курсового проекта представлены: постановка задачи на курсовое проектирование, особенности и этапы разработки информационной системы, включая описание используемых технологий, архитектуру приложения и интерфейс пользователя. Методические рекомендации содержат описание результатов выполнения курсового проектирования, перечень рекоменду-

емой основной и дополнительной литературы, базы данных и информационно-справочные и поисковые системы.

Практическая цель курсового проектирования состоит в развитии и закреплении различного рода комплексных умений и практических навыков, связанных с выполнением программной реализации информационных систем, включающих следующие ключевые аспекты:

- проектирование и разработку архитектуры приложения;
- проведение декомпозиции программного кода на архитектурные слои, логические части, классы и функции;
- организацию синхронной или асинхронной обработки данных;
- проведение рефакторинга программного кода;
- проведение анализа html-структуру документа;
- работу с библиотеками программного кода;
- анализ предметной области, структурирование данных и работа с массивами объектов.

## Постановка задачи на выполнение курсового проекта

Выполнение курсового проекта посвящено рассмотрению методики автоматизации бизнес-процессов по сбору, структурированию и обработке больших массивов данных. В частности, в рамках настоящего курсового проекта предстоит реализовать **приложение для сбора данных** со статических и/или динамических страниц сайта (*эти требования можно уточнить у преподавателя*).

Как правило, программы, осуществляющие сбор данных со страниц сайтов, называют парсерами данных, а сам процесс парсингом данных.

### Выбор сайта для сбора данных

**Выбор сайта** для сбора данных остаётся за обучающимся.

Есть некоторые **требования к сайту** (или страницам сайта):

- собираемых данных должно быть достаточно много – в результате парсинга должен получиться массив объектов (от нескольких десятков до нескольких сотен объектов),

- в каждом объекте должно быть несколько полей с разными типами данных, при этом желательно, чтобы данные были устроены иерархически, то есть чтобы некоторые поля объекта содержали в качестве значений объекты со своими полями (Листинг 1),

- желательно чтобы сайт предполагал паджинацию, то есть необходимость переключения между страницами сайта.

Паджинация предполагает, что сайт не выводит сразу всю информацию на одну страницу. Как правило, товар (любые другие объекты) сортируется по какому-то критерию (по цене, по популярности) и выводится на страницу сайта пор-

ционно (например, по 20 на страницу). От пользователя требуется переключаться между страницами сайта чтобы посмотреть весь товар. Таким образом, для программы-парсера требуется предусмотреть возможность перехода между страницами сайта чтобы получить данные с нескольких страниц. Чаще всего в этом не возникает затруднения при программировании, так как адрес каждой следующей страницы задаётся в параметре запроса page (или аналогичном) и от программиста требуется просто организовать цикл перебора страниц (http-запросов со сменой url-адресов с параметром page).

Листинг 1.

Пример json-структуры с разными типами данных.

```
{
  "coord": {
    "lon": 56.2855,
    "lat": 58.0174
  },
  "weather": [
    {
      "id": 804,
      "main": "Clouds",
      "description": "пасмурно",
      "icon": "04d"
    }
  ],
  "base": "stations",
  "main": {
    "temp": 4.7,
    "feels_like": 4.7,
    "temp_min": 4.7,
    "temp_max": 4.7,
    "pressure": 1017,
    "humidity": 93,
    "sea_level": 1017,
    "grnd_level": 998
  },
  "visibility": 10000,
  "wind": {
    "speed": 1,
    "deg": 0
  },
  "clouds": {
    "all": 100
  },
```

```
  "dt": 1760686023,
  "sys": {
    "type": 1,
    "id": 8984,
    "country": "RU",
    "sunrise": 1760669641,
    "sunset": 1760706366
  },
  "timezone": 18000,
  "id": 511196,
  "name": "Пермь",
  "cod": 200
}
```


Обычно перечисленным выше требованиям удовлетворяют страницы маркетплейсов. Для курсового проектирования не обязательно использовать именно сайт какого-либо маркетплейса, возможно подойдёт сайт с результатами спортивных мероприятий за несколько лет или какой-нибудь портал со статистическими данными с нескольких регионов страны. Если вы не в состоянии самостоятельно определиться с выбором сайта для парсинга, можете остановиться на сборе данных с сайта https://www.chitai-gorod.ru/ (работает без регистрации и авторизации). Как вариант, для проектирования можно использовать погодные сайты, то есть написать парсер для сбора статистики о погоде в разных населённых пунктах на какой-то диапазон дат.

Кроме сбора данных с помощью написанного и апробированного парсера, в рамках курсового проектирования, **требуется** также показать примеры обработки и сохранения данных, в частности добавьте в реализацию две функции:

1) функция для **сортировки** полученного массива данных по одному из доступных параметров, например, по количеству отзывов (по рейтингу, по цене, по автору, по весу);
2) функция для **фильтрации** данных, например, по бренду товара или по наличию отзывов.

Результаты каждой из функций следует сохранять в json-файл (для проверки разместить в Приложении В).

В рамках курсового проектирования требуется выполнить следующие **этапы разработки** информационной системы:

1) составление **Технического задания** (по ГОСТу, разместить в Приложении А);
2) разработка **приложения** для парсинга данных (разместить в Приложении Б, выбор языка программирования и сайта для сбора данных остаётся за студентом);
3) **апробация** парсера данных и сохранение данных, **анализ** эффективности и недостатков выполненного решения, доработка (при необходимости), описание интерфейса пользователя – разместить в тексте Отчёта в Разделе 3.

Таким образом, в рамках выполнения курсового проекта предстоит провести автоматизацию бизнес-процесса по сбору данных для организации.

## 2. Разработка приложения по сбору данных

В рамках выполнения курсового проекта в качестве программной реализации (прототипа) требуется разработать консольное приложение по сбору данных с сайта.

*Используемые технологии (можно поменять):*

– Node.js – платформа для разработки серверных приложений на JavaScript;

– json – способ представления иерархически организованных данных (*обязательно*);

– csv – способ представления плоских табличных данных (*при необходимости, не обязательно*);

– re (regular expressions) – формальный язык обработки текстовой информации, для поиска и проверки комбинаций символов в тексте, основанный на использовании метасимволов (*при необходимости, но, как правило, удобнее использовать специализированные библиотеки для поиска данных, например cheerio*);

– html – язык разметки для описания структуры web-страниц сайта;

– css – таблицы стилей для описания дизайна элементов web-сайта (*описание классов оформления, имя класса можно использовать для поиска данных на html-странице*);

– lodash – функциональная библиотека (*при необходимости, можно использовать для сортировки, фильтрации и преобразования коллекций данных*),

– fetch (или request, sync-request, axios) – для получения html-страницы сайта.

Предполагается, что будет создано **консольное приложение** (*это интерфейс пользователя*) на языке программирования JavaScript для платформы Node.js (кроссплатформенное - для Windows, Linux или Mac OS) с возможностью запуска и настройки параметров поиска вручную из терминала, а также с возможностью просмотра результатов сбора как в терминале, так и в сохранённых json-файлах.

**Последовательность парсинга** данных можно разбить на следующие **самостоятельные этапы**:

1) получение html-структуры документа;
2) перевод сырой структуры в иерархический объект, содержащий все элементы html-структуры (DOM);
3) поиск (фильтрация) необходимых элементов;
4) паджинация, получение данных с нескольких страниц сайта (*при необходимости*);
5) структурирование данных (изменение полей, работа с типами данных, добавление полей) и формирование массива объектов с необходимыми данными;
6) обработка (фильтрация, сортировка, изменение) массива объектов и сохранение в структурированном формате в файл (json, csv, xml, yaml) или в базу данных (этот этап рекомендуется оформить в виде отдельного программного модуля, к функционалу которого обращаться из основной программы).

При необходимости некоторые из обозначенных этапов можно совмещать. Однако разделение функционала позволяет не только по-отдельности его проектировать, но и организовать раздельное тестирование и, при необходимости, внесение изменений без изменения общего кода.

Далее опишем содержание обработки данных на каждом из этапов.

**Этап 1. Получение html-страницы.**

Этот этап включает формирование строки запроса, выполнение запроса, получение ответа от сервера, сохранение html-страницы в виде файла локально или в виде переменной в программе.

Рекомендуется разбивать формирование строки http-запроса к сайту на отдельные составные части, например, это может быть сочетание протокола, имени хостинга, маршрута, параметров запроса - `` `${host}search?${params}` `` (Листинг 2).

Листинг 2.

```
const host = "https://www.chitai-gorod.ru/";
let query = "Пушкин"; // запрос
let queryEncoded = encodeURIComponent(query); // кодируем кириллицу
для URL
const sortDirection = "priceDesc"; // направление сортировки
let pageNum = 2; // номер страницы - для паджинации
const params = `phrase=${queryEncoded}&sortPreset=${sortDirection}
&page=${pageNum}`;
const url = `${host}search?${params}`; // формируем URL с параметрами
запроса
```

Собственно, организовать получение содержимого html-страницы можно разными методами – fetch, request, sync-request, axios. Выбор методов загрузки содержимого может зависеть от особенностей (асинхронно, синхронно) приложения и остаётся на ваш выбор.

**Этап 2. Перевод в программный объект со структурой html-страницы.**

Этот этап может быть реализован с помощью одной из специализированных библиотек. Например, библиотека

`cheerio.js` позволяет это сделать достаточно просто (Листинг 3).

Листинг 3.

```
const $ = cheerio.load(html);
```

**Этап 3. Поиск необходимых элементов.**

Поиск элементов может быть осуществлён по названию тега, по атрибуту name, по id селектора, по css-классу, или просто по порядковому номеру в последовательности аналогичных тегов (например, необходимый тег может быть просто первым на html-странице) (Листинг 4).

Листинг 4.

```
const price = $('.product- card-price').first().text().trim();
```

**Этап 4. Паджинация.**

Сбор данных с нескольких страниц сайта может быть организован с помощью цикла со сменой нумерации страниц в строке запроса к сайту. Например, в Листинге 2 была показана переменная `pageNum`, которая может обеспечить паджинацию по страницам. Рекомендуется осуществлять паузу между последовательными запросами к страницам сайта (1-3 секунды). Для организации паузы можно использовать метод `setTimeout()`. В Листинге 5 приведён пример последовательной загрузки нескольких рисунков с сайта с паузой между загрузками.

Листинг 5.

```
const sleep = (ms) => new Promise(resolve => setTimeout(resolve, ms));

const downloadImages = async (links, delay = 2000) => {
    for (let link of links) {
        saveImg(link); // сохраняем очередной рисунок
        await sleep(delay); // задержка в миллисекундах
    }
};
```

```
let imgLinks = []; // массив ссылок на рисунки, полученный при парсинге

downloadImages(imgLinks); // пауза по умолчанию 2 сек
```

Обычный цикл for, организующий перебор объектов (страниц сайта, ссылок и т.п.), нельзя менять на функциональный метод `map`, так как он не поддерживает асинхронность.

**Этап 5. Структурирование данных.**

На этом этапе требуется полученные после парсинга объекты преобразовать в коллекции (структуры) по формату, который был установлен (как предполагается) требованиями бизнес-аналитика (в данном случае вы сами являетесь бизнес-аналитиком и сами формируете требования). Например, может потребоваться удалить ненужные или лишние поля объектов, изменить названия некоторых полей объекта на более читаемые, добавить поля, удалить поля, изменить типы данных полей. Если данные были получены с нескольких страниц сайта, то потребуется свести их к одному массиву объектов.

**Этап 6. Обработка и сохранение данных.**

После сбора и структурирования данных в Вашей реализации **требуется** показать примеры **обработки** данных:

6.1 провести **сортировку** полученного массива данных по одной из характеристик на Ваш выбор (по количеству отзывов, по рейтингу, по цене и т.п.);

6.2 провести **фильтрацию** данных по одной из характеристик, например, по бренду товара или наличию отзывов (на Ваш выбор).

Каждый из примеров обработки требуется **сохранить в отдельном файле** в формате `json`. Оба файла следует поместить в **приложение В «Данные парсинга»**, структурированно с отступами для удобства чтения в моноширном шрифте и размером шрифта не более 10 пт, с одинарным междустрочным абзацным интервалом (для плотности печати данных). Если файлы содержат большое количество объектов и занимают много страниц (более 5-ти), то можно каждый из массивов объектов сократить (оставив необходимое для контроля корректности сбора данных количество объектов), так чтобы снизить общее количество страниц.

**Важное уточнение**

Следует уточнить, что не все страницы html являются статическими. **Для динамически формируемых страниц** (например, с технологией Ajax) обычная загрузка html-страницы с помощью традиционных методов (sync-request или fetch) не приведёт к желаемому результату. После загрузки окажется, что в сырой строке будет только шаблон страницы без необходимых данных. В то время как при просмотре через браузер данные отображаются. В этом случае требуется использовать **альтернативный подход и загружать страницу, используя методы, имитирующие работу браузера**.

Для динамически формируемых страниц в Node.js обычно используют библиотеку `puppeteer`, которая позволяет в асинхронном режиме загрузить содержимое html-страницы со всеми данными, как и при просмотре страницы в обычном браузере. Затем методами самой библиотеки `puppeteer`, или, при желании, методами библиотеки `cheerio` преобразовать сырую страницу с данными в программный

объект и уже из него обычным образом (по имени класса, по id, по названию селектора) выбирать данные.

Для курсового проекта не обязательно искать сайт с динамически формируемыми страницами, достаточно показать умения парсинга статических страниц. Но, если вы нашли интересный сайт с подходящими для курсового проекта данными, и страницы в нем формируются динамически, то не возбраняется выбрать для реализации сайт с динамическими страницами (*это ненамного сложнее*).

## 3. Подготовка и защита курсового проекта

*Этапы выполнения курсового проекта:*

- получение задания на выполнение курсового проекта;
- обсуждение этапов проектирования и порядка разработки информационной системы;
- подготовка технического задания на проектирование информационной системы;
- разработка структур для хранения данных или схемы базы данных для информационной системы (при необходимости, не обязательно);
- проектирование модулей информационной системы (рекомендуется отдельные части бизнес-логики, например, функции сохранения, фильтрации и сортировки данных вынести в отдельный модуль);
- тестирование и доработка;
- составление отчёта о выполнении курсового проекта;
- защита курсового проекта (за Курсовой проект выставляется оценка, учитывается выполнение требований методического пособия к функционалу программной реализации, сложность парсинга, используемые технологии, стиль оформления кода и корректность программной реализации, оформление и ясность формулировок в Отчёте).

Рекомендуемый объём работы, **содержание Отчёта** по курсовому проекту и постраничное описание отражены в таблице 1.

Таблица 1.

| Название пункта и рекомендуемое содержание Отчёта по курсовому проекту | Рекомендуемое количество страниц |
|---|---|
| Титульный лист | 1 |
| Содержание | 1 |

| | |
|---|---|
| 1. Постановка задачи на проектирование информационной системы<br>*Кратко описать этапы автоматизированного процесса сбора данных и требования к разрабатываемому приложению (интерфейс, функционал приложения, ожидаемые результаты, форматы сохранения).* | 1 |
| 2. Анализ технологий проектирования<br>*– обзор технологий для хранения данных в структурированных файлах (csv, json, xml, yaml) или в базах данных (SQLite, MySQL, PostgreSQL);*<br>*– обзор языков программирования (например, C#, Python, Node.js);*<br>*– описание возможностей программных библиотек, используемых для реализации проекта;*<br>*– следует уделить внимание обоснованию выбора технологий для реализации проекта согласно задания на курсовое проектирование.* | 2-4 |
| 3. Реализация функционала информационной системы<br>*– описание этапов работы приложения с детализацией программной реализации с примерами программного кода;*<br>*– описание структур для хранения данных (json, csv, xml, yaml) или логической модели базы данных (схема и описание к ней);*<br>*– описание методов для обработки файлов (json, csv, xml, yaml) или описание SQL-запросов к базе данных;*<br>*– описание интерфейса пользователя и скриншоты программы.* | 5-6 |
| Заключение<br>*– описание что сделано, выводы об эффективности автоматизации и недостатках реализации, перспективы дальнейшей разработки.* | 1 |
| Список использованных источников | 1 |
| Приложение А. Техническое задание | 5 |
| Приложение Б. Программный код | 2-4 |
| Приложение В. Данные парсинга | 2-4 |

**Требования к оформлению** Отчёта по курсовому проекту:

- листы формата А4, поля примерно: слева 3 см, справа 1,5 см, сверху и снизу 2 см;
- шрифт для основного текста Times New Roman (или аналоги) размером 14 пт;
- основные абзацы текста: выравнивание по ширине;
- заголовки разделов: выравнивание по центру и полужирным шрифтом;
- программный код: обязательно в светлой теме, моноширным шрифтом и размером небольшим (10-12 пт);
- в тексте должны быть ссылки на все источники, упомянутые в «Списке использованных источников».

## Заключение

Работая над технической реализацией информационной системы в рамках выполнения курсового проекта обучающийся дополняет и систематизирует знания, полученные им в рамках контактной работы с преподавателем. Активное включение в изучение технологий реализации программных приложений позволит обучающемуся освоить компетенции, предусмотренные при изучении дисциплины «Программная инженерия».

В рамках организации выполнения курсового проекта должны реализовываться следующие принципы:

– принцип интерактивности обучения (обеспечение интерактивного диалога и обратной связи, которая позволяет осуществлять контроль и коррекцию действий студента);

– принцип развития интеллектуального потенциала (формирование алгоритмического, наглядно-образного, теоретического стилей мышления, умений принимать оптимальные или вариативные решения в сложной ситуации, умений обрабатывать информацию);

– принцип обеспечения целостности и непрерывности дидактического цикла обучения (предоставление возможности выполнения всех звеньев дидактического цикла в пределах темы).

В процессе выполнения курсового проекта обучающийся должен активно использовать электронные библиотечные системы, электронные поисковые системы и электронные периодические справочники. Кроме того, обучающийся должен регулярно использовать Интернет-ресурсы, находящиеся в свободном доступе.

## Перечень основной и дополнительной литературы

### Основная:

1. Прохоренок, Н. А. JavaScript и Node.js для веб-разработчиков / Н. А. Прохоренок, В. А. Дронов. – СПб: БХВ-Петербург, 2022. – 768 с.

2. Полуэктова, Н. Р. Разработка веб-приложений: учебное пособие для вузов / Н. Р. Полуэктова. – Москва: Издательство Юрайт, 2022. – 204 с. – (Высшее образование). – ISBN 978-5-534-13715-6. – Текст: электронный // Образовательная платформа Юрайт [сайт]. – URL: https://urait.ru/bcode/496682

3. Молинаро Э. SQL/ Сборник рецептов. – 2-ое изд.: пер. с англ. / Э. Молинаро, Р. де Граф. – СПб: БХВ-Петербург, 2022. – 592 с.

### Дополнительная:

1. Лаврищева, Е. М. Программная инженерия и технологии программирования сложных систем: учебник для вузов / Е. М. Лаврищева. – 2-е изд., испр. и доп. – Москва: Издательство Юрайт, 2022. – 432 с. – (Высшее образование). – ISBN 978-5-534-07604-2. – Текст : электронный // Образовательная платформа Юрайт [сайт]. – URL: https://urait.ru/bcode/491029

2. Гниденко, И. Г. Технологии и методы программирования: учебное пособие для вузов / И. Г. Гниденко, Ф. Ф. Павлов, Д. Ю. Федоров. – Москва: Издательство Юрайт, 2022. – 235 с. – (Высшее образование). – ISBN 978-5-534-02816-4. – Текст: электронный // Образовательная платформа Юрайт [сайт]. – URL: https://urait.ru/bcode/489920

3. Лаврищева, Е. М. Программная инженерия. Парадигмы, технологии и CASE-средства: учебник для вузов / Е. М. Лаврищева. – 2-е изд., испр. – Москва: Издательство Юрайт, 2022. – 280 с. – (Высшее образование). – ISBN 978-5-534-01056-5. – Текст: электронный // Образовательная платформа Юрайт [сайт]. – URL: https://urait.ru/bcode/491048

4. Малов, А. В. Концепции современного программирования: учебное пособие для вузов / А. В. Малов, С. В. Родионов. – Москва: Издательство Юрайт, 2022. – 96 с. – (Высшее образование). – ISBN 978-5-534-14911-1. – Текст : электронный // Образовательная платформа Юрайт [сайт]. – URL: https://urait.ru/bcode/485436

## Базы данных, информационно-справочные и поисковые системы

1. Электронный каталог библиотеки Пермского ГАТУ: базы данных, содержащие сведения обо всех видах литературы, поступающей в фонд Научной библиотеки Пермского ГАТУ. – URL: https://pgsha.ru/generalinfo/library/webirbis/.
2. Электронная библиотека / Пермский государственный аграрно-технологический университет имени академика Д. Н. Прянишникова. – URL: https://pgsha.ru/generalinfo/library/elib/.
3. ConsultantPlus (КонсультантПлюс) : компьютерная справочно-правовая система. – URL: https://www.consultant.ru/. – Режим доступа: для авторизированных пользователей. – Доступ из корпусов ПГАТУ.
4. eLIBRARY.RU : научная электронная библиотека. – URL: https://elibrary.ru/defaultx.asp. – Режим доступа: для зарегистрированных пользователей.
5. Polpred.com (Полпред.ком) : электронно-библиотечная система. – URL: https://polpred.com/news. – Режим доступа: для зарегистрированных пользователей.
6. IPRSMART : электронно-библиотечная система. – URL: https://www.iprbookshop.ru/. – Режим доступа: для зарегистрированных пользователей.
7. Гребенникон : электронная библиотека. – URL: https://grebennikon.ru/. – Режим доступа: для зарегистрированных пользователей.
8. Руконт : национальный цифровой ресурс : межотраслевая электронная библиотека. – URL: https://lib.rucont.ru/search. – Режим доступа: для зарегистрированных пользователей.

9. Лань : электронно-библиотечная система. – URL: https://e.lanbook.com/. – Режим доступа: для зарегистрированных пользователей.
10. Юрайт : электронно-библиотечная система. – URL: https://urait.ru/. – Режим доступа: для зарегистрированных пользователей.
11. Сетевая электронная библиотека (СЭБ). – URL: https://e.lanbook.com/. – Режим доступа: для зарегистрированных пользователей.
12. Электронные информационные ресурсы ФГБНУ ЦНСХБ. – Режим доступа: для авторизированных пользователей. – Доступ из интернет-зала главного корпуса.
13. **Перечень открытых интернет-ресурсов**:
Интуит - Открытый университет:
– Курс «Введение в стандарты WEB»:
https://intuit.ru/studies/professional_skill_improvements/1432/info
– Курс «Web-технологии»:
https://intuit.ru/studies/professional_skill_improvements/1252/info

Приложение 1

## Шаблон технического задания на разработку информационной системы

_____

наименование организации – разработчика ТЗ на АС

УТВЕРЖДАЮ

Руководитель _____

(должность, наименование предприятия – заказчика АС)

_____ _____

Личная подпись Расшифровка подписи

(печать)

Дата _____

УТВЕРЖДАЮ

Руководитель_____

(должность, наименование предприятия – разработчик АС)

_____ _____

Личная подпись Расшифровка подписи

(печать)

Дата _____

_____

наименование вида АС

_____

наименование объекта автоматизации

_____

сокращённое наименование АС

**ТЕХНИЧЕСКОЕ ЗАДАНИЕ**

На _____ листах

Действует с _____

202__

## 1. Общие сведения

### 1.1 Наименование системы

Полное наименование разрабатываемой системы – «Название Вашего приложения».

Краткое наименование – «Краткое название Вашего приложения».

### 1.2 Наименование заказчика и исполнителя

Организация: ФГБОУ ВО Пермский ГАТУ.

Адрес: ул. Петропавловская, 23.

Телефон: +7 (342) 217-90-66;

Исполнитель: Фамилия Имя Отчество.

### 1.3 Плановые сроки начала и окончания работ

Дата начала работ: __.__.20__.

Дата окончания работ: __.__.20__.

## 2. Назначение и цели создания системы

Назначение приложения по сбору данных «Название Вашего приложения»:

– автоматизация процессов выполнения задач сотрудниками организации.

Цели создания информационной системы:

– повышение производительности сбора и обработки данных, необходимых для осуществления бизнес-процессов организации;

– стандартизация информационных потоков организации;

– уменьшение количества рутинных операций;

– повышение достоверности собранных данных за счёт уменьшения операций, выполняемых вручную.

## 3. Характеристика объекта автоматизации

Объектом автоматизации является подсистема сбора и обработки структурированных данных организации. Данные собираются из открытых источников сети интернет (со статических или динамических страниц сайтов). Собранные данные впоследствии могут передаваться в реляционную базу данных, на страницы сайта организации, в файлы табличных процессоров, в структурированные фалы (json, csv, xml, yaml).

## 4. Требования к системе

Общие требования к приложению по сбору данных являются:

– работоспособность и надёжность (продолжение сбора данных даже при встрече с некоторым нарушении вёрстки html-страницы);

– интуитивно понятный интерфейс (или простота работы с консольным приложением);

– лицензионная чистота – применение средств в рамках общего лицензионного соглашения касательно корпоративного портала;

– соблюдение информационной безопасности и разграничение прав доступа к данным.

### 4.1 Требования к способам и средствам связи для информационного обмена между компонентами

Для обеспечения информационного обмена компоненты подсистемы должны формировать и обрабатывать поток данных в одном из стандартных структурированных форматов (json, csv, xml, yaml).

#### 4.1.1 Перспективы развития, модернизация системы

Дальнейшим развитием приложения по сбору данных может быть объединение сбора данных с различных сайтов и информационных порталов в одну информационную систему и добавление модуля для интеллектуальной обработки данных.

#### 4.1.2 Требования к квалификации персонала и режиму его работы

Для обеспечения максимальной работоспособности пользователей должны устанавливаться перерывы:

– через 2 часа после начала смены и через 1,5–2 часа продолжительностью 15 минут;

– через каждый час работы продолжительностью 10 минут

Для эксплуатации приложения по сбору данных определены следующие роли:

– системный администратор – должен быть квалифицированным специалистом с практическим опытом выполнения работ по администрированию программных и технических средств. В обязанности входит: установка, модернизация, настройка программного обеспечения, ведение учётных записей портала;

– пользователь приложения по сбору данных – должен иметь опыт работы с персональным компьютером и файловой подсистемой операционной системы (Linux или Windows) и свободно осуществлять базовые действия по настройке и запуску приложения, по работе с файлами данных.

#### 4.1.3 Требования к надёжности технических средств и программного обеспечения

Надёжность по отношению к техническим средствам должна обеспечиваться использованием в системе средств повышенной отка-

зоустойчивости и их резервированием, а также дублированием носителей информационных банков данных.

Надёжность программного комплекса обеспечивается использованием сертифицированных операционных систем, общесистемных программных средств и инструментальных программных систем, используемых при разработке программного обеспечения.

Разрабатываемое приложение должно обеспечивать защиту от некорректных действий пользователей и ошибочных исходных данных (корректная обработка исключительных ситуаций).

### 4.1.4 Требования к безопасности

Разрабатываемое приложение по сбору данных должно обеспечивать безопасный доступ к данным, предотвращая несанкционированный доступ или модифицирование данных.

Модуль аутентификации (при наличии) должен обеспечивать защищённый доступ ко всему программному интерфейсу приложения, за исключением статичной формы авторизации в системе предоставляя возможность пройти аутентификацию в корпоративном портале при помощи логина и пароля.

### 4.1.5 Требования по эргономике и технической эстетике

Модуль должен иметь удобный и интуитивно понятный графический пользовательский интерфейс. Диалоговый интерфейс должен соблюдать контекст подсистемы организационной коммуникации университета и управления в целом, тем самым действия конечного пользователя должны быть ясны и знакомы.

Пользовательский интерфейс модуля также должен аккомпанировать цветовой гамме и общему стилю корпоративного портала.

### 4.1.6 Требования к программному обеспечению

При проектировании приложения по сбору данных необходимо эффективно использовать веб-фреймворк Express.js, в качестве серверного окружения – программную платформу Node.js, а для хранения данных структурированный формат фалов (json, csv, xml, yaml – по выбору разработчика) или СУБД (PostgreSQL, MySQL, SQLite – по выбору разработчика).

### 4.1.7 Требования к техническому обеспечению

Техническое обеспечение системы должно максимально и наиболее эффективно использовать существующее в отделе автоматизации оборудование:

- процессор – 2x Intel Xeon 3.7 ГГц;
- оперативная память – 32 ГБ;

– дисковая система – 1ТБ;
– сетевой адаптер – 1 Гб/с.

## 5. Порядок контроля и приёмки системы

Приёмо-сдаточные испытания системы проводятся с привлечением сотрудников отдела автоматизации. По результатам опытной эксплуатации оформляется акт о приёме работ. Акт содержит заключение о соответствии системы техническому заданию.

### 5.1 Требования к составу и содержанию работ подготовки объекта автоматизации к вводу системы в действие

При подготовке к вводу в эксплуатацию приложения по сбору данных отдел управления информатизации должен обеспечить выполнение следующих работ:

– определить подразделение и ответственных должностных лиц для внедрения программного продукта;

– обеспечить пользователей интуитивно понятным интерфейсом или кратким руководством, которое поможет быстрее освоить внедрённую систему;

– провести опытную эксплуатацию приложения по сбору данных.

Приложение 2

Шаблон титульного листа отчёта по курсовому проекту

Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное
учреждение высшего образования
«Пермский государственный аграрно-технологический университет
имени академика Д. Н. Прянишникова»

Кафедра Информационных технологий
и программной инженерии

**ОТЧЁТ ПО КУРСОВОМУ ПРОЕКТУ**

на тему: «**Разработка приложения
по сбору данных с сайта магазина Читай-город**»

Выполнил:
студент группы ПИб-хххх
направления подготовки
09.03.03 Прикладная информатика
Иванов Иван Иванович

Проверил:
доцент кафедры ИТиПИ, к.т.н., доцент
Беляков Андрей Юрьевич

Пермь – 2025