

# **Отчет по лабораторной работе № 3**

Бондарь Татьяна Владимировна

# Содержание

<b>1</b>	<b>Цель работы</b>	<b>5</b>
<b>2</b>	<b>Задание</b>	<b>6</b>
<b>3</b>	<b>Теоретическое введение</b>	<b>7</b>
<b>4</b>	<b>Выполнение лабораторной работы</b>	<b>9</b>
4.1	Оформление отчета к лабораторной работе № 2 . . . . .	9
<b>5</b>	<b>Выводы</b>	<b>11</b>
	<b>Список литературы</b>	<b>12</b>

## Список иллюстраций

4.1	Начало заполнения отчета . . . . .	9
4.2	Цель, задание, теоретическое введение . . . . .	9
4.3	Заполнение части “Выполнение лабораторной работы” . . . . .	10
4.4	Заполнение части “Выводы” и “Контрольные вопросы” . . . . .	10
4.5	Команда make . . . . .	10

## **Список таблиц**

# 1 Цель работы

Научиться оформлять отчёты с помощью легковесного языка разметки Markdown.

## 2 Задание

1. Сделайте отчёт по предыдущей лабораторной работе в формате Markdown.
2. В качестве отчёта просьба предоставить отчёты в 3 форматах: pdf, docx и md (в архиве, поскольку он должен содержать скриншоты, Makefile и т.д.)

## 3 Теоретическое введение

Чтобы создать заголовок, используйте знак ( # ). Чтобы задать для текста полужирное начертание, заключите его в двойные звездочки. Чтобы задать для текста курсивное начертание, заключите его в одинарные звездочки. Чтобы задать для текста полужирное и курсивное начертание, заключите его в тройные звездочки. Блоки цитирования создаются с помощью символа >. Неупорядоченный (маркированный) список можно отформатировать с помощью звездочек или тире. Чтобы вложить один список в другой, добавьте отступ для элементов дочернего списка. Упорядоченный список можно отформатировать с помощью соответствующих цифр. Чтобы вложить один список в другой, добавьте отступ для элементов дочернего списка. Синтаксис Markdown для встроенной ссылки состоит из части [link text] , представляющей текст гиперссылки, и части (file-name.md) – URL-адреса или имени файла, на который дается ссылка. Markdown поддерживает как встраивание фрагментов кода в предложение, так и их размещение между предложениями в виде отдельных огражденных блоков. Огражденные блоки кода – это простой способ выделить синтаксис для фрагментов кода. Внутритекстовые формулы делаются аналогично формулам LaTeX. Для обработки файлов в формате Markdown будем использовать Pandoc. Конкретно, нам понадобится программа pandoc , pandoc-citeproc <https://github.com/jgm/pandoc/releases>, pandoc-crossref <https://github.com/lierdakil/pandoc-crossref/releases>. Преобразовать файл README.md можно следующим образом: 1 pandoc README.md -o README.pdf или так 1 pandoc README.md -o README.docx Можно использовать следующий Makefile 1 FILES = \$(patsubst %.md, %.docx, \$(wildcard \*.md)) 2 FILES += \$(patsubst %.md,

*%pdf, \$(wildcard \*.md))*





3. Начинаем заполнять основную часть работы: прописываем шаги выполнения лабораторной работы, добавляем иллюстрации и подписи к ним. (рис. fig. 4.3).

```

# Выполнение лабораторной работы

## Установка программного обеспечения

1. Установка git, скачивая на свой компьютер пакетов git. (см. @fig:001).

!$[Классика git]$(image/1.png)@fig:001 width=700)

2. Установка пакета sh (см. @fig:002).

!$[Классика sh]$(image/2.png)@fig:002 width=700)

## Базовая настройка git.

3. Заполнить имя и email базовых репозитория. Настройками user-а и имени репозитория.

$[Классика git]$(image/3.png)@fig:003 width=700)

```

Рис. 4.3: Заполнение части “Выполнение лабораторной работы”

4. Прописываем вывод, отвечаем на контрольные вопросы к лабораторной работе. (рис. fig. 4.4).

**Вводные**

1. **Цель задачи:** автоматизировать процесс внедрения и сопровождения клиентских решений. Необходимо обеспечить интеграцию с существующими системами, разработать удобный интерфейс и обеспечить безопасность данных.

**Составляющие задачи:**

- 1.1. **Исследовать существующие решения (CRM, ERP) и определить требования к новому решению/модулю.**
  - 1.1.1. Анализ текущих процессов и выявление痛点 (pain points).
  - 1.1.2. Определение функциональных требований (функционал, интеграции, отчеты).
  - 1.1.3. Определение нефункциональных требований (производительность, безопасность, масштабируемость).
  - 1.1.4. Выбор технологий и инструментов разработки.
- 1.2. **Разработать архитектуру и дизайн системы.**
  - 1.2.1. Выбор архитектуры (микросервисы, монолит).
  - 1.2.2. Проектирование баз данных (нормализация, оптимизация).
  - 1.2.3. Разработка API (REST, GraphQL).
  - 1.2.4. Проектирование пользовательского интерфейса (UI/UX).
- 1.3. **Разработать и реализовать функционал.**
  - 1.3.1. Разработка backend-логики (сервисы, контроллеры).
  - 1.3.2. Разработка frontend-логики (клиентская часть).
  - 1.3.3. Реализация интеграций с существующими системами.
  - 1.3.4. Настройка безопасности (аутентификация, авторизация, шифрование).
- 1.4. **Провести тестирование и внедрение.**
  - 1.4.1. Разработка тестов (юнит-тесты, интеграционные тесты).
  - 1.4.2. Проведение тестирования (функциональное, нагрузочное, безопасность).
  - 1.4.3. Подготовка документации (техническая, пользовательская).
  - 1.4.4. Внедрение системы и обучение пользователей.

2. **Оценить результаты работы и определить дальнейшие шаги.**

Рис. 4.4: Заполнение части “Выводы” и “Контрольные вопросы”

5. С помощью команды `make` создаем отчет в форматах `.docx` и `.pdf`. (рис. fig. 4.5).

```
tvbondar@vbox:~/work/study/2024-2025/Операционные системы/os-intro/study_2024-2025_os-intro/labs/lab02/report$ make
pandoc "report.md" --filter pandoc-crossref --number-sections --citeproc -o "report.docx"
pandoc "report.md" --filter pandoc-crossref --pdf-engine=lualatex --pdf-engine-opt=--shell-escape --citeproc --number-sections -o "report.pdf"
```

Рис. 4.5: Команда make

## 5 Выводы

Мы научились оформлять отчёты с помощью легковесного языка разметки Markdown.

# Список литературы

Руководство по оформлению Markdown файлов. [Электронный ресурс]. GitHub  
Gist URL: <https://gist.github.com/Jekins/2bf2d0638163f1294637>