**Отчет по практической работе №**3

**по дисциплине МДК 01.02 “Инструментальные средства разработки программного обеспечения”.**

Выполнил: студент

группы 319

Муртазин Руслан Шамилевич

Дата 22.10.2024

**Цель работы:** Изучение и применение системы контроля версий на практике, включая настройку типов импортируемых файлов, путей, фильтров и других параметров для эффективного управления репозиториями.

**Основная структура задания**

Проект:мобильное приложение для заказа еды на дом.

Cписок типов файлов, которые следует включить в репозиторий системы контроля версий:

**Клиентская часть (Android & iOS):**

* **Код:**
  + **Android:**
    - .java, .kt: Файлы с кодом на Java или Kotlin.
    - .xml: Файлы с разметкой XML для UI (Activity, Layout и др.).
    - .gradle: Файлы с конфигурацией для Gradle build системы.
  + **iOS:**
    - .swift, .m: Файлы с кодом на Swift или Objective-C.
    - .storyboard: Файлы с разметкой Storyboard для UI.
    - .xib: Файлы с разметкой XIB для UI.
    - .plist: Файлы с конфигурационными данными.
* **Изображения:**
  + .png, .jpg, .jpeg, .svg: Файлы с изображениями для UI и иконками.
* **Ресурсы:**
  + .drawable, .mipmap: Папки с ресурсами для Android (изображения, иконки).
  + .assets: Папка с ресурсами для Android и iOS.
* **Тесты:**
  + .test.java, .test.kt: Файлы с тестами для Android (JUnit).
  + .test.swift, .test.m: Файлы с тестами для iOS (XCTest).

**Серверная часть:**

* **Код:**
  + .py, .js, .java, .php: Файлы с кодом на Python, JavaScript, Java или PHP.
  + .sql: Файлы с SQL запросами.
* **Конфигурационные файлы:**
  + .json, .yaml: Файлы с конфигурационными данными для сервера и API.
* **Документация:**
  + .md: Файлы с документацией в формате Markdown.
  + .pdf: Файлы с документацией в формате PDF.
* **Тесты:**
  + .test.py, .test.js, .test.java, .test.php: Файлы с тестами для серверного кода.

В Git нет прямого способа создать “белый список” файлов, которые должны отслеживаться. Поэтому разработчику придется использовать  git add для добавления конкретных файлов в область индексации.

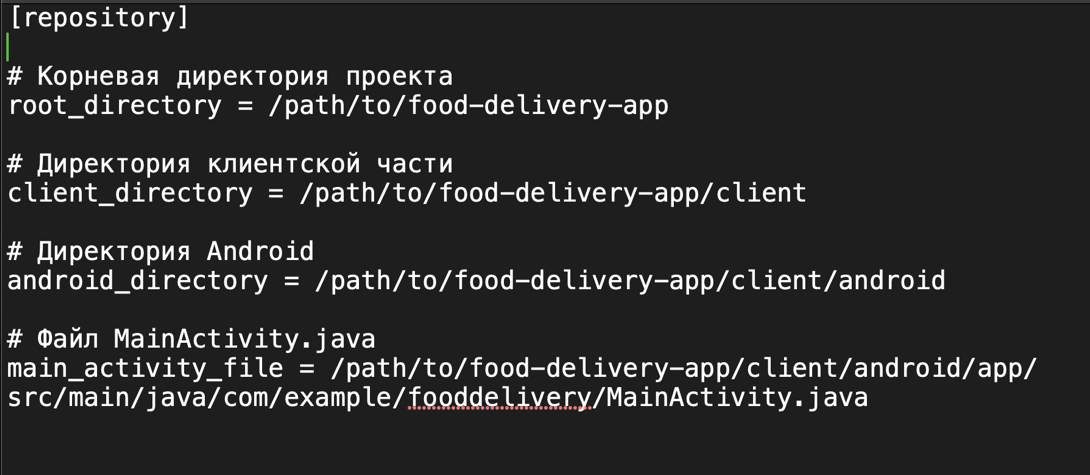
**Пример:**

Предположим, разработчик хочет отслеживать только файл src/components/RestaurantCard.jsx. Это компонент React, отображающий информацию о ресторане. В то же время он не хочет, чтобы Git отслеживал другие файлы проекта. Разработчик использует  git add для добавления файла в область индексации:

git add src/components/RestaurantCard.jsx

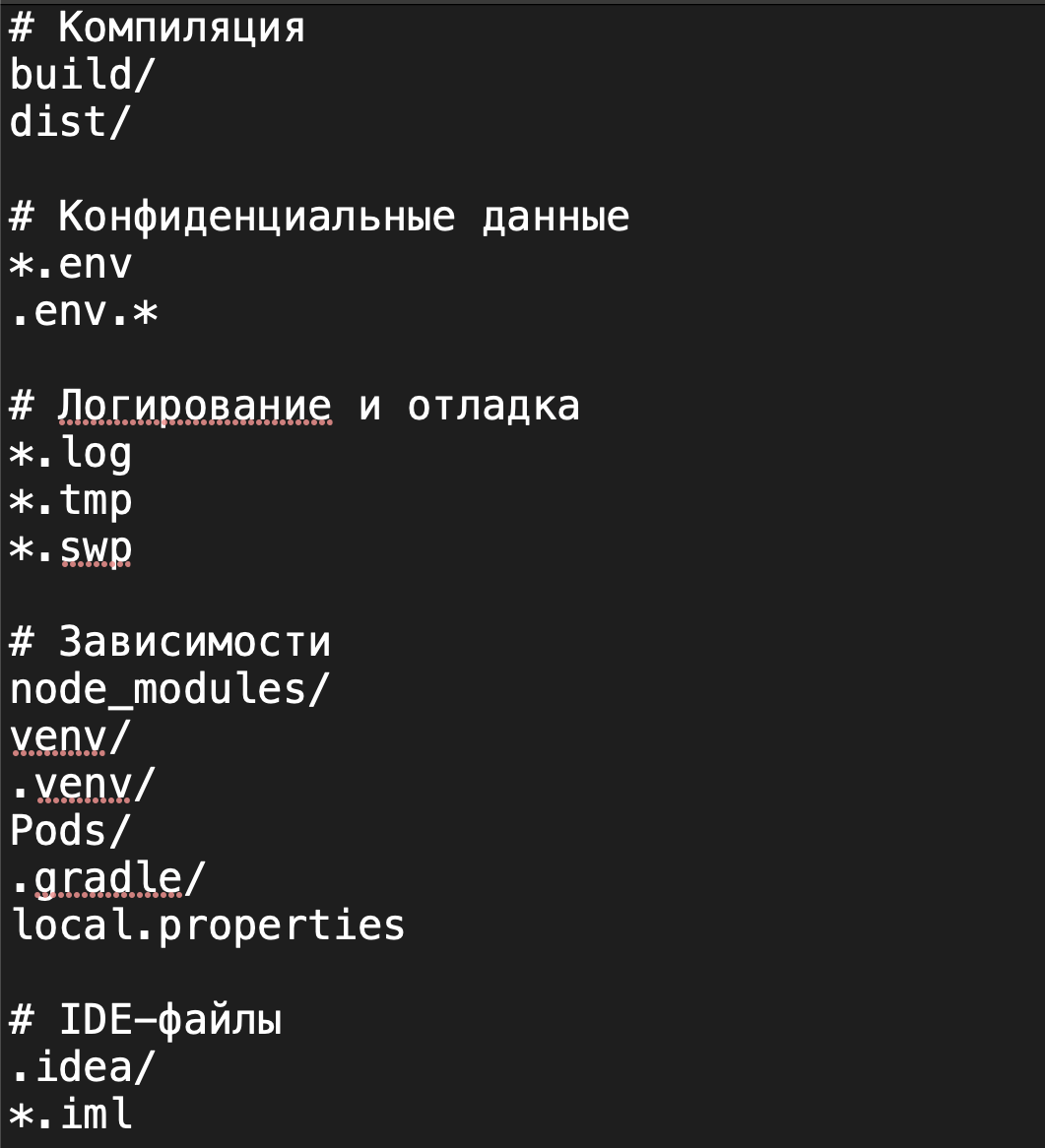
Теперь Git будет отслеживать только файл RestaurantCard.jsx, а все остальные файлы будут проигнорированы.

Абсолютные пути к основным директориям:



Но настраивать пути в конфигурации ненужно, так как в Git пути к файлам и каталогам устанавливаются неявно, с помощью команды  git add и хранятся в индексе репозитория.

Пример файла .gitignore для проекта “Заказ еды на дом”, который исключает шаблоны файлов и директории:



В Git, по умолчанию, при возникновении конфликтов используется стратегия слияния “merge”. Чтобы настроить инструмент для разрешения конфликтов, используйте команду:

git config --global merge.tool <имя\_инструмента>

Git поддерживает несколько инструментов, таких как meld, kdiff3, vimdiff.

**Выводы работы:** В ходе практической работы я углубился в использование Git, поняв ключевые моменты работы с файлами и директориями, а также настройки поведения при слиянии. Теперь я готов использовать Git более эффективно.