Чек-лист для проверки ДЗ

(заполняется спикером урока)

Пример заполнения:

Что нужно знать для проверки ДЗ
1) SPRING - основы работы со spring
2) Open API - основы работы со swagger
3) Базовые знания Runnable
4) Базовые знания Fork/Join или Executor Framework
5) Знания REST api

Чек-лист:

1. Общие требования:

1.1. Форк репозитория: Убедитесь, что студент сделал форк репозитория с кодом лекции и работал в своём форке.

1.2. Ссылка на репозиторий: Убедитесь, что студент отправил ссылку на свой репозиторий для проверки.

1.3. Документация: В репозитории должна присутствовать документация (например, README.md), описывающая цель проекта, инструкцию по запуску и описание многопоточной реализации.

2. Выбор темы:

2.1. Выбор темы: Проверить, что студент выбрал одну из предложенных тем (или предложил свою тему и согласовал её с ментором).

2.2. Описания темы: Убедитесь, что в документации указана выбранная тема и каким образом в проекте используется многопоточность.

3. Реализация многопоточности:

3.1. Executor Framework или Fork/Join: Проект должен использовать Executor Framework или Fork/Join для реализации многопоточности.

3.2. Эндпойнт: В проекте должен быть как минимум один REST эндпойнт, который демонстрирует работу многопоточной системы.

3.3. Код многопоточности: Проверить наличие правильной реализации многопоточных задач, включая создание и управление потоками, задания для потоков и их исполнение.
3.4. Асинхронная обработка: Убедитесь, что задачи выполняются асинхронно и не блокируют основной поток выполнения.

4. Темы для приложений (проверить реализацию одной из предложенных тем):

4.1. Task Manager: Проверить, что задачи обрабатываются в фоновом режиме, планировщик задач для уведомлений и выполнения повторяющихся задач работает корректно.

4.2. Chat Application: Убедиться, что приложение обрабатывает сообщения от нескольких пользователей одновременно и отправляет уведомления асинхронно.

4.3. File Uploader: Проверить, что крупные файлы загружаются асинхронно и происходит параллельная проверка антивирусами.

4.4. News Aggregator: Убедиться, что происходит параллельное извлечение новостей с различных API, обработка новостей и обновление базы данных.

4.5. Weather Monitoring: Проверить асинхронное обновление данных о погоде и отправку уведомлений пользователям при изменении погодных условий.

4.6. E-commerce Backend: Убедиться, что заказы обрабатываются параллельно, подтверждения по электронной почте отправляются асинхронно, а запасы на складе обновляются.

4.7. Social Media Feed: Проверить параллельную обработку и отображение постов, асинхронную публикацию и лайки постов.

4.8. Fitness Tracker: Убедиться, что данные с фитнес-устройств обрабатываются параллельно, а уведомления о достижении целей отправляются асинхронно.

4.9. Online Quiz System: Проверить асинхронную обработку и оценку тестов, параллельное сохранение результатов.

4.10. Event Management System: Убедиться, что регистрация участников обрабатывается параллельно, рассылка приглашений и уведомлений происходит асинхронно.

4.11. Своя тема согласованная с ментором: Убедиться что проект соотвествует заявленной теме.

5. Технические детали:

5.1. Качество кода: Проверить, чтобы код был чистым, хорошо структурированным и читабельным.

5.2. Использование зависимостей: Убедиться, что проект правильно настроен с использованием Maven/Gradle и все зависимости подключены корректно.

5.4. Логирование: Убедиться, что в проекте используется инструмент журналирования (например, Logback или Log4j) для отслеживания выполнения многопоточных задач.

6. Дополнительные проверки:

6.1. Запуск проекта: Убедитесь, что проект успешно компилируется и запускается.

6.2. Работа эндпойнтов: Проверить работу REST эндпойнтов через Postman или любой другой инструмент для тестирования API.

6.3. Отсутствие deadlock'ов: Проверить проект на наличие возможных дедлоков (deadlock) в многопоточном коде.