ПРАКТИЧЕСКОЕ ЗАНЯТИЕ № 9
РАЗРАБОТКА ПЕРВОЙ РАБОТАЮЩЕЙ ВЕРСИИ

Дисциплина: CSE842 «Capstone Project». **Образовательная программа:** 6B06102 «Computer Science», 4 курс, 8 семестр. **Связь с лекцией:** лекция 9 «Реализация функционального прототипа». **Результаты обучения:** РО5. **Объём:** 3 часа практики, 7 часов самостоятельной работы, включая СРСП. **Максимальный балл:** 4.

Цель занятия: реализовать основные функции прототипа в виде «ходячего скелета», настроить качество кода и подготовить инструкцию запуска.

Задачи:

1. Реализовать вертикальный срез сквозного сценария через все слои.
2. Реализовать требования Must ядра и слой хранения по модели данных.
3. Настроить линтер, форматтер, модульные тесты и рецензирование через pull request.
4. Подготовить README с инструкцией запуска и проверить её «тестом новичка».
5. Отметить версию тегом v0.9.

Программное обеспечение и инструменты: Язык и фреймворк по выбору проекта, Git, GitHub Actions/GitLab CI, ruff/flake8, black, unittest/pytest.

1. Краткие теоретические сведения

«Ходячий скелет» — тонкий вертикальный срез через интерфейс, API, ядро и БД, который собирается и тестируется автоматически. Код организуется по архитектуре: domain (правила), services (сценарии), adapters (БД, внешние сервисы, HTTP). SQL-запросы используют параметры; конфигурация и секреты — в переменных окружения; версии зависимостей зафиксированы.

Пример ядра со слоем хранения:

```
def book(repo, eq, usr, s, e):
    if repo.overlaps(eq, s, e):
        return {"status": 409,
                "error": "SLOT_TAKEN"}
    return {"status": 201,
            "id": repo.add(eq, usr, s, e)}
```

Таблица 1 — Определение готовности задачи

№	Условие	Проверка
1	код соответствует критериям приёмки	ссылка на FR в PR
2	тесты написаны и проходят	зелёный CI
3	линтер без замечаний	шаг lint
4	рецензия другим участником	одобрение PR
5	документация обновлена	изменения docs/

2. Порядок выполнения работы

1. Создайте структуру кода по архитектуре; настройте окружение и файл зависимостей.
2. Реализуйте вертикальный срез сценария варианта (таблица 2) с минимальным интерфейсом.
3. Реализуйте инвариант варианта в ядре и в ограничениях БД; напишите модульные тесты.
4. Настройте линтер, форматтер и CI; работайте через ветки и pull request с рецензией.
5. Подготовьте скрипт демо-данных и README с инструкцией запуска.
6. Проведите «тест новичка» с участником другой команды; исправьте найденные проблемы.
7. Отметьте версию тегом v0.9 и подготовьте короткое видео сценария (1–2 мин).

3. Индивидуальные задания

Номер варианта совпадает с номером темы проекта, выбранной на практическом занятии № 1, и сохраняется до конца семестра. Команда (до 4 человек) выполняет задание своего варианта; распределение работы между участниками указывается в отчёте.

Таблица 2 — Индивидуальные задания: вертикальный срез

№	Сквозной сценарий (минимальный срез)	Инвариант, реализуемый в ядре и БД
1	выбрать врача → выбрать слот → подтвердить запись → SMS пациенту → отметка о приёме	один слот приёма занимает не более одного пациента
2	преподаватель открывает занятие → динамический QR → студент сканирует → отметка → отчёт эдвайзеру	одна отметка студента на занятие; QR действителен 30 с
3	датчик измеряет PM2.5 → публикация в MQTT → запись в БД → график на дашборде → оповещение при превышении	измерения не теряются при кратковременном разрыве связи (буфер на устройстве)
4	студент загружает решение → очередь → запуск тестов в песочнице → оценка → отчёт	решение исполняется изолированно, с лимитами времени и памяти, без доступа к сети
5	поиск товара → рекомендации → корзина → заказ → уведомление	товар не продаётся сверх остатка

№	Сквозной сценарий (минимальный срез)	Инвариант, реализуемый в ядре и БД
6	найти книгу → забронировать → выдача по штрихкоду → напоминание → возврат	экземпляр выдан не более чем одному читателю
7	вопрос студента → поиск ответа → ответ или передача сотруднику → оценка ответа	вопрос без ответа не теряется (эскалация сотруднику)
8	кадр камеры → статус мест → карта свободных мест → водитель выбирает место	статус места на карте не старше 30 с
9	приёмка → сканирование → размещение в ячейку → отгрузка → отчёт об остатках	остаток товара в ячейке не бывает отрицательным
10	заказ → ресторан принял → курьер назначен → трекинг на карте → доставлено	заказ назначен не более чем одному курьеру
11	журнал → парсинг → признаки окна → детектор → оповещение администратору	каждая аномалия высокой серьёзности порождает оповещение
12	фото чека → распознавание суммы → категория → операция в бюджет → отчёт за месяц	сумма операций равна балансу счёта
13	заявление → согласование методистом → подпись декана → PDF → уведомление студенту	подписанный документ не изменяется (контроль хеша)
14	выбор площадки → слот → временная блокировка → оплата → QR-пропуск → вход	слот площадки не продаётся дважды
15	датчик движения → правило → включить свет → уведомление в приложении	команда исполняется не более одного раза; ручная команда имеет приоритет
16	кадр → номер → проверка пропуска → открыть шлагбаум → запись в журнал	шлагбаум открывается только для действующего пропуска
17	счётчик → сбор → хранение → прогноз на сутки → оповещение о перерасходе	пропуски данных помечаются и не заполняются молча
18	преподаватель создаёт тест → студент проходит → автосохранение → автопроверка → результаты	ответы после окончания времени не принимаются; ответ не теряется при разрыве связи

№	Сквозной сценарий (минимальный срез)	Инвариант, реализуемый в ядре и БД
19	откуда и куда → маршрут с пересадками → время прибытия → показ на карте	маршрут не использует рейсы, ушедшие до момента запроса
20	выбрать преподавателя → слот → подтвердить → ссылка на встречу → напоминание	консультация не пересекается с занятиями преподавателя

4. Содержание отчёта

1. Ссылка на тег v0.9 и описание реализованных функций.
2. Отчёт CI (модульные тесты, линтер).
3. README с инструкцией запуска и протокол «теста новичка».
4. Видео или демонстрация упрощённого сценария.
5. Вклад участников (коммиты, задачи, рецензии).

5. Форма сдачи

Артефакты занятия размещаются в репозитории проекта (каталог docs/ или указанный в задании), в трекере задач создаются и закрываются соответствующие задачи. Преподавателю сдаётся ссылка на версию (коммит или тег) и краткий отчёт.

6. Критерии оценивания

Максимальный балл за работу — **4** (календарный план силлабуса). Баллы распределяются по трём критериям силлабуса.

Таблица 3 — Критерии оценивания текущего задания

Критерий	Доля	Что проверяется
Полнота и соответствие требованиям задания	40%	все пункты задания и индивидуального варианта выполнены
Техническая корректность	40%	решения обоснованы, расчёты и код верны, терминология точна
Проверяемость материалов	20%	артефакты в репозитории, ссылки, команды запуска, данные

Работа сдаётся через канал, указанный преподавателем; в командном проекте каждый участник указывает свой вклад. Использование генеративного ИИ раскрывается по правилам университета.

7. Контрольные вопросы

1. Что такое «ходячий скелет»?
2. Как структура кода отражает архитектуру?
3. Как реализован инвариант в вашем проекте?

4. Как хранятся конфигурация и секреты?
5. Что показал «тест новичка»?

Литература

1. Sommerville I. Software Engineering. — 10th ed. — Pearson, 2016.
2. IEEE Computer Society. SWEBOK Guide v4.0a.
3. Fowler M. Continuous Integration. — martinfowler.com.