ПРАКТИЧЕСКОЕ ЗАНЯТИЕ № 2
СРАВНЕНИЕ АНАЛОГОВ И ВЫБОР РЕШЕНИЯ

Дисциплина: CSE842 «Capstone Project». **Образовательная программа:** 6B06102 «Computer Science», 4 курс, 8 семестр. **Связь с лекцией:** лекция 2 «Анализ существующих решений и техническая осуществимость». **Результаты обучения:** РО1, РО3. **Объём:** 3 часа практики, 7 часов самостоятельной работы, включая СРСП. **Максимальный балл:** 2.

Цель занятия: проанализировать существующие решения, сравнить технические альтернативы по согласованным критериям и обосновать выбор с учётом осуществимости и рисков.

Задачи:

1. Найти и описать 3–5 аналогов, определить нишу проекта.
2. Для ключевой подсистемы предложить не менее трёх технических альтернатив.
3. Согласовать критерии и веса, построить матрицу альтернатив и выполнить анализ чувствительности.
4. Оценить пять видов осуществимости и составить реестр рисков.

Программное обеспечение и инструменты: Python 3 (расчёт матрицы), электронные таблицы, Markdown, Google Scholar, GitHub.

1. Краткие теоретические сведения

Матрица альтернатив вычисляет взвешенную оценку $S(a) = \Sigma\, w_i \cdot x_i(a)$, где w_i — веса критериев ($\Sigma\, w_i = 1$), $x_i(a)$ — оценки альтернативы по шкале 1–5. Веса согласуются до выставления оценок; оценки подтверждаются фактами (документация, бенчмарк, пробный запуск). Анализ чувствительности проверяет, меняется ли лидер при разумном изменении весов: если порог переключения близок к базовому весу, выбор проверяется экспериментом на неделе 6.

Осуществимость оценивается по пяти видам: техническая, экономическая, временная, операционная, юридическая. Приоритет риска — произведение вероятности (1–5) на влияние (1–5); риски с оценкой ≥ 15 требуют действий.

Шаблон расчёта:

```
crit = {"срок освоения": .30, "производительность":
.25,
        "стоимость": .20, "зрелость": .25}
alts = {"A": [5, 3, 5, 4], "B": [3, 5, 4, 5],
        "C": [4, 4, 2, 4]}
w = list(crit.values())
for a, v in alts.items():
    print(a, round(sum(x * k for x, k in zip(v, w)),
2))
```

2. Порядок выполнения работы

1. Найдите аналоги (продукты, открытые проекты, публикации) и заполните таблицу сравнения по 5–7 критериям.
2. Сформулируйте вывод о нише проекта: чем решение отличается от аналогов.
3. Для подсистемы своего варианта (таблица 2) опишите три альтернативы; добавьте четвёртую при необходимости.
4. Согласуйте с руководителем 4–6 критериев и веса; выставьте оценки с обоснованием.
5. Рассчитайте матрицу кодом, выполните анализ чувствительности по самому спорному весу, постройте график.
6. Оцените осуществимость по пяти видам и выполните proof of concept ключевой библиотеки или сервиса.
7. Составьте реестр рисков (≥ 5) с приоритетами и реакцией; сохраните docs/alternatives.md.

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: подсистема и альтернативы

№	Подсистема	Альтернативы для сравнения	Ключевой критерий
1	хранение и API	Django + PostgreSQL; Node.js + MongoDB; Firebase	целостность записи
2	клиентское приложение	Flutter; React Native; PWA	доступ к камере и офлайн
3	хранилище временных рядов	InfluxDB; TimescaleDB; SQLite	скорость агрегирования
4	песочница	Docker; firejail; Judge0	изоляция и накладные расходы
5	рекомендации	популярные товары; item-kNN; ALS	precision@5
6	полнотекстовый поиск	PostgreSQL FTS; Elasticsearch; SQLite FTS5	морфология и скорость
7	поиск ответа	TF-IDF; BM25; векторные эмбеддинги	доля верных в топ-3
8	распознавание занятости	классический CV; CNN; ультразвуковые датчики	точность и стоимость

№	Подсистема	Альтернативы для сравнения	Ключевой критерий
9	офлайн-работа	SQLite + синхронизация; PouchDB/CouchDB; Firebase offline	конфликты синхронизации
10	передача координат	WebSocket; Server-Sent Events; периодический опрос	задержка и нагрузка
11	детектор	правила; Isolation Forest; LSTM-автоэнкодер	F1 и скорость
12	распознавание	Tesseract; EasyOCR; облачный OCR	точность суммы
13	генерация PDF	WeasyPrint; LaTeX; DOCX-шаблон с конвертацией	качество и скорость
14	блокировка слота	блокировка в БД; распределённая блокировка Redis; оптимистическая версия	двойные продажи
15	брокер	Mosquitto; EMQX; HTTP-опрос	задержка и надёжность
16	распознавание	EasyOCR; детектор YOLO + OCR; OpenALPR	точность и скорость на CPU
17	прогноз	сезонная наивная модель; градиентный бустинг; LSTM	MAE
18	серверный фреймворк	Django; FastAPI; Node.js Express	производительность и срок освоения
19	алгоритм маршрутизации	Дейкстра по графу; Connection Scan Algorithm; RAPTOR	время ответа
20	синхронизация календаря	Google Calendar API; CalDAV; экспорт iCal	двусторонность

4. Содержание отчёта

1. Таблица аналогов и вывод о нише.
2. Матрица альтернатив с весами, расчётом и обоснованием оценок.
3. Анализ чувствительности (таблица и график).
4. Оценка осуществимости и результат proof of concept.
5. Реестр рисков.
6. Обоснованный выбор и план отступления.

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 — **2** (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. Как строится матрица альтернатив?
2. Зачем нужен анализ чувствительности?
3. Какие виды осуществимости вы оценили?
4. Что такое TRL?
5. Как вычисляется приоритет риска?

Литература

1. Sommerville I. Software Engineering. — 10th ed. — Pearson, 2016.
2. NASA. Systems Engineering Handbook. NASA/SP-2016-6105 Rev2.