**Техническое задание на командную разработку проекта**

**Общая цель**

Цель проекта — командная разработка полноценного веб-приложения с применением профессиональных методик проектирования и ведения разработки. Участники должны пройти все этапы создания продукта: от идеи до MVP, используя современные инструменты, фреймворки и **одну из методологий управления проектом**, выбранную заранее для каждой команды.

**Методология разработки**

Каждая команда работает по одной из следующих **гибких или каскадных методологий** управления проектами:

**1. Waterfall (Каскадная модель)**

* Этапы строго последовательно: анализ → проектирование → разработка → тестирование → релиз.
* Подходит для команд с чёткими ролями и стабильными требованиями.

**2. Agile**

* Гибкий подход с итеративной разработкой и постоянной адаптацией.
* Основное внимание на быструю поставку функционала и обратную связь от "заказчика".
* В команде — частые встречи (стендапы), ретроспективы, планирование спринтов.

**3. Kanban**

* Метод визуального управления задачами через доску (Trello, Notion, Jira и пр.).
* Акцент на непрерывный поток задач, минимум завязки на спринты.
* Задачи перемещаются по колонкам: "To Do", "In Progress", "Review", "Done".

**4. Scrum**

* Разработка по спринтам (обычно 1–2 недели).
* Роли: Scrum-мастер, Product Owner, команда разработчиков.
* Прозрачность, скорость поставки и гибкость в приоритетах.

**Каждая команда должна описать в финальной документации, как была применена выбранная методология, какие инструменты использовались (например, Trello, Jira, Notion) и как строилось взаимодействие внутри команды.**

**Выбор проекта (тематика)**

Выберите одну из следующих тематик:

1. **Новостной портал**
2. **Сайт для бумажных стаканчиков (интернет-магазин)**
3. **Сайт для интровертов (сообщество/платформа)**

**Стек технологий**

* **Backend**: Django (Python)
* **Frontend**: HTML, CSS (SCSS или TailwindCSS), JavaScript
* **База данных**: PostgreSQL / SQLite
* **Дизайн и прототипы**: Figma или аналогичный редактор

**Обязательные элементы проекта**

1. **ER-диаграмма базы данных**
   * Полностью проработанная схема со связями.
   * Оформлена в Figma или другом редакторе.
2. **UI/UX прототип**
   * Минимум 5 уникальных и интерактивных экранов.
   * Оформление в Figma, согласование дизайна внутри команды.
3. **MVP**
   * Полностью рабочее веб-приложение с минимально необходимым функционалом.
   * Пользователь может взаимодействовать с сайтом (авторизация, действия с контентом и пр.).
4. **Система контроля версий**
   * Репозиторий на GitHub/GitLab обязательно.
   * Использование веток, осмысленных коммитов и pull request'ов.
   * Ветка main — всегда актуальна и рабочая.
5. **Документация каждого этапа**
   * Хранится в docs/ или оформлена в README.
   * Включает: методологию, архитектуру, описание функциональности, запуск проекта, роли участников, анализ итогов.
6. **Разделение ролей**
   * Указать, кто выполнял какие задачи:
     + Backend-разработка
     + Frontend-верстка и логика
     + UI/UX-дизайн
     + Документация и организация процесса
     + Деплой и тестирование

**Примерная структура проекта**

/project-root

│

├── backend/ # Django-приложение

├── frontend/ # Шаблоны, статика

├── docs/ # Техническая документация

├── design/ # UI/UX-макеты (Figma, PDF)

├── requirements.txt # Зависимости Python

├── README.md # Инструкция к проекту

└── .gitignore

**✅ Итог**

* Каждый участник команды вносит вклад и несёт ответственность за свою часть проекта.
* Проект должен быть не просто сдан, а продемонстрирован как **работающее приложение**, с обоснованной структурой и оформлением.
* Итоговая презентация включает:
  + Краткий рассказ о функционале и дизайне
  + Пояснение по методологии и ролям
  + Демонстрация работающего MVP