**МГТУ им. Н.Э.Баумана**

**Кафедра ИУ7**

**«Проектирование программного обеспечения»**

**Сезон 2025**

**Лабораторная работа № 2**

**Проектирование**

Подготовить в репозитории ветку *lab2* и создать merge request в ветку *master*.

Все пункты продолжать выполнять в Readme.md.

1. Описание типа приложения и выбранного технологического стека (Console App/Desktop/Mobile/Web MPA/ Web SPA/Cross-platform и т.д.);

2. Верхнеуровневое разбиение на компоненты (в следующих лабах сможете уточнить): на базовом уровне выделяем компонент доступа к данным, компонент бизнес-логики и компонент реализации UI. Отобразить диаграммой компонентов. Подумать про *принцип инверсии зависимостей*.

3. UML диаграммы классов для двух отдельных компонентов - компонента доступа к данным и компонента с бизнес-логикой ( не забыть «модельные» классов сущностей: сущности базы данных, сущности системы и транспортные сущности. Если уровень транспортных сущностей пока сложно спроектировать, можно скопировать системные. Уточните их на следующих этапах)

**Примечание 1.** Для реализации компонента доступа к данным рекомендуется рассмотреть использование паттерна Repository и его многочисленных аналогов.

**Примечание 2.** Для реализации компонента UI и его связи с остальным приложением рекомендуется рассмотреть использование MV\*-подобного паттерна (MVC, MTV, MVP, MVVM, MVPVM, MVI и др.).

**Примечание 3:** Если вы считаете, что бизнес логику логичнее разделить на несколько компонентов - разделяйте, но укажите причины этого решения.

**Требования к технологическому стеку.**

**Рекомендуемые ЯП для использования в ЛР**

**Требование:** язык программирования должен быть полноценным ООП-языком со статической типизацией и поддержкой интерфейсов (полностью абстрактных классов).

1. Рекомендуемые языки:

*C#, Java, Kotlin, Scala, C++, TypeScript, Delphi*

2. Допустимые не полноценно ООП-языки (с учетом согласования с семинаристом способов реализации слоев луковичной чистой архитектуры, интерфейсов, инверсии зависимостей, компонентов и т.д.)

*Go, Rust*

3. Условно разрешенные языки (с учетом согласования с семинаристом)

*Python*

При этом, требуется неукоснительное следование рекомендациям:

[Методичка "Рекомендации по применению ЯП Python в курсе ППО" Жаров С.В.](https://docs.google.com/document/d/1UHMdQHijQnUXg2e7bYSmtLlhFW6WAUnHsARnynSRCUU/edit#)

4. Запрещенные языки

*Ruby, PHP, Perl, JavaScript, 1С, Lua*

**Примечание:**

Если планируется компонент UI на базе SPA-подхода, то вариант для этого компонента строго один - TypeScript, и компонентный фреймворк (Angular, React, Vue). При этом остальная часть ПO может быть выполнена на другом языке, естественно.