

УДК 004

РОЛЬ ТРЕХУРОВНЕВОЙ АРХИТЕКТУРЫ В МАЛЫХ ПРОЕКТАХ

Д. И. Ильина, студент гр.ТРСм-1-24, 1 курс

Научный руководитель: Зарипова Р.С., к.т.н., доцент

Казанский государственный энергетический университет, г.Казань

Трехуровневая архитектура — это подход к проектированию приложений, который разделяет систему на три логических уровня: презентационный, бизнес-логики и доступа к данным. Эта модель улучшает поддерживаемость, масштабируемость и безопасность приложений. Используют для веб-приложений, корпоративных систем и проектов с потенциальным ростом [1].

Рассмотрим структуру трехуровневой архитектуры. Она включает в себя такие уровни, как презентационный, бизнес-логики, доступа к данным. Первый уровень отвечает за взаимодействие с пользователем (UI). В веб-приложениях это может быть фронтенд (HTML, CSS, JavaScript), в десктопных — формы и интерфейсы. Второй уровень содержит основные правила и алгоритмы работы приложения и обрабатывает данные, полученные от презентационного уровня, и передаёт их уровню данных [2]. Третий уровень обеспечивает взаимодействие с базой данных или внешними хранилищами и выполняет CRUD-операции (Create, Read, Update, Delete).

Одним из главных преимуществ использования в малых проектах является модульность и поддерживаемость. Чёткое разделение ответственности между слоями упрощает внесение изменений [3]. Например, замена базы данных не затронет бизнес-логику или интерфейс. Даже в небольших проектах возможен рост. Трехуровневая архитектура позволяет легче добавлять новые функции. Недостатки и ограничения:

- Избыточность для очень простых проектов. Если приложение состоит из нескольких форм и пары запросов к БД, трехуровневая архитектура может добавить ненужную сложность;

- Увеличение времени разработки. Требуется больше кода для разделения слоёв, что может быть излишним для MVP или прототипа;

- Дополнительные накладные расходы. Взаимодействие между уровнями может замедлять работу в сравнении с монолитным подходом.

Рассмотрим пример использования презентационного уровня. Для этого напишем код с применением HTML и JavaScript (рис. 1). Он позволит отображать список задач, создавать/редактировать их. Данный уровень предназначен для того, чтобы продемонстрировать пользователю возможности системы с помощью визуализации (рис. 2).

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <title>To-Do List</title>
5 </head>
6 <body>
7   <h1>Мои задачи</h1>
8   <input type="text" id="taskInput" placeholder="Новая задача">
9   <button onclick="addTask()">Добавить</button>
10  <ul id="taskList"></ul>
11
12  <script>
13    async function loadTasks() {
14      const response = await fetch('/api/tasks');
15      const tasks = await response.json();
16      const list = document.getElementById('taskList');
17      list.innerHTML = tasks.map(task => `
18        <li>
19          ${task.title}
20          <button onclick="deleteTask(${task.id})">Удалить</button>
21        </li>
22      `).join('');
23    }
24
25    async function addTask() {
26      const input = document.getElementById('taskInput');
27      await fetch('/api/tasks', {
28        method: 'POST',
29        headers: { 'Content-Type': 'application/json' },
30        body: JSON.stringify({ title: input.value })
31      });
32      input.value = '';
33      loadTasks();
34    }
35
36    async function deleteTask(id) {
37      await fetch(`/api/tasks/${id}`, { method: 'DELETE' });
38      loadTasks();
39    }
40
41    loadTasks();
42  </script>
43 </body>
44 </html>
```

Рисунок 1 – Код программы

Мои задачи

Новая задача

Рисунок 2 – Результат

Трехуровневая архитектура в малых проектах часто позволяет модернизировать процессы и упростить их. Например, в небольших проектах разработчики часто пишут весь код в одном файле (например, фронтенд, логика и работа с БД). Со временем это превращается в хаос. Для данного случая можно разделить на уровни и презентационный уровень будет отвечает только за отображение, бизнес-логика - изолирована в отдельном слое, а работу с данными можно вынести в DAL [4].

Малый проект может внезапно вырасти (новые фичи, клиенты, нагрузки). В таком случае можно заменить SQLite на PostgreSQL, поменяв только DAL, добавить мобильное приложение, оставив тот же бэкенд или вынести бизнес-логику в микросервис позже.

Еще один кейс: в монолитах валидация часто дублируется или пропускается. В данном случае валидацию перенести в бизнес-уровне (единая точка истины) и презентационный уровень вынести для отображения ошибки.

Трехуровневую архитектуру не смогут применить [5]:

1. Очень маленькие проекты. Код программ менее 200 строк;
2. Проекты, у которых MVP неокончательный и будет переписан;
3. Проекты, в которых применены микросервисы.

В заключении можно сделать вывод, что трехуровневая архитектура — это мощный инструмент, который можно адаптировать даже для небольших проектов. Даже в небольшом проекте трехуровневая архитектура помогает:

- Упростить поддержку (изменение БД не затрагивает фронтенд).
- Обеспечить тестируемость (можно мокать DAL при тестах API).
- Готовить приложение к масштабированию (например, переход с SQLite на PostgreSQL потребует изменений только в DAL).

Список литературы:

1. Волков В.В. Управление автономными системами: трехуровневые и двухуровневые архитектуры // Исследование различных направлений современной науки. Сборник материалов LIII-ой международной очно-заочной научно-практической конференции. В 3-х томах. Москва, 2024. С. 43-45.
2. Селяков М.А. Трехуровневая архитектура микросервиса в ASP.NET // Вестник науки. 2020. Т. 1. № 4 (25). С. 76-80.
3. Сумбулов И.С. Применение средств виртуализации и контейнеризации в разработке и эксплуатации приложения на основе трехуровневой архитектуры // Актуальные вопросы эксплуатации систем охраны и защищенных телекоммуникационных систем. Сборник материалов конференции. Воронеж, 2022. С. 59-61.
4. Захлевный Д.С., Мелешенко К.И., Карельская К.А., Хабаров А.Р. Особенности применения трехуровневой архитектуры при разработке информационной кадровой системы // Информационные ресурсы и системы в экономике, науке и образовании. Сборник статей XI Международной научно-практической конференции. Под редакцией А.П. Ремонтова. Пенза, 2021. С. 53-58.
5. Зарипова Р.С., Ильина Д.И. Разработка программного продукта для автоматизации процесса продаж оптовой продукции // Естественно-гуманитарные исследования. 2023. № 5 (49). С. 104-108.