ПРАВИТЕЛЬСТВО РОССИЙСКОЙ ФЕДЕРАЦИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ
«ВЫСШАЯ ШКОЛА ЭКОНОМИКИ»

Факультет компьютерных наук
Образовательная программа «Программная инженерия»

Домашняя работа №3 по дисциплине “Проектирование архитектуры программных систем”

Проектирование
Архитектурный стиль

Выполнили студенты группы БПИ223
Абдуллаев А.Ш.
Жалилов А.
Курманова А.

Москва 2025

Содержание

Название

Архитектурный стиль для приложения SmartTable.

Дата создания

23 марта 2025 г.

Контекст решения

Проект SmartTable представляет собой систему для автоматизации обслуживания клиентов в ресторанах и кафе. Она включает в себя:

- Telegram MiniApp — интерфейс для клиентов
- Административную панель — для ресторанного персонала
- Общую серверную часть, обеспечивающую бизнес-логику, работу с данными, авторизацию, заказы, меню, сессии и синхронизацию

Анализ требований выявил ключевые архитектурно-значимые атрибуты качества, влияющие на организацию системы:

- Согласованность
- Производительность
- Надежность
- Отказоустойчивость
- Масштабируемость
- Удобство использования

Исходя из этих факторов и учитывая ограничения по ресурсу и этапу, необходимо выбрать архитектурный стиль, обеспечивающий баланс между целостностью, простотой реализации и технической реалистичностью.

Решение

Для реализации системы SmartTable выбран монолитный архитектурный стиль с модульной логической организацией.

Обоснование выбора

- Согласованность: обеспечивается за счет общей базы данных и единого приложения, в рамках которого легче реализовать транзакционную целостность.
- Производительность: единый процесс и память позволяют избежать сетевых задержек между компонентами, характерных для распределённых архитектур.
- Надежность: логика системы централизована, и отказ одного из компонентов легче отследить и восстановить.
- Отказоустойчивость: резервное копирование и failover-инфраструктура возможны даже в рамках монолита.
- Масштабируемость: ограниченно реализуется за счет вертикального масштабирования и кэширования, чего достаточно для MVP.
- Удобство разработки: команда из 3 человек быстрее и проще реализует систему, учитывая, что монолит легче тестировать и запускать.

Последствия

Положительные:

- Простота разработки, тестирования и деплоя
- Лёгкость трассировки ошибок и мониторинга
- Быстрая реализация MVP и прототипов
- Согласованность данных благодаря общей транзакционной модели

Отрицательные:

- Ограниченная масштабируемость
- Рост сложности при добавлении новых функций
- Сложность переработки архитектуры в будущем

Риски

- Единая точка отказа: сбой может повлиять на всю систему
- Трудности масштабирования отдельных компонентов
- Повышенная связанность: изменения в одном модуле могут затронуть другие
- Потенциальное усложнение миграции к микросервисам в будущем (для уменьшения риска была выбрана модульная логическая организация)

Альтернативные варианты

1. Микросервисная архитектура

   Плюсы:

   - Масштабируемость каждого компонента отдельно
   - Высокая отказоустойчивость
   - Независимость технологий

   Минусы:

   - Повышенная сложность инфраструктуры
   - Требуется зрелая DevOps-практика
   - Высокие накладные расходы на коммуникацию между сервисами

2. Слоистая архитектура (Layered Architecture)

   Плюсы:

   - Четкое разделение ответственности (UI / бизнес-логика / доступ к данным)
   - Упрощает тестирование отдельных слоев

   Минусы:

   - Все слои находятся в рамках одного процесса — нет изоляции между подсистемами
   - Потенциальное дублирование логики между слоями
   - Сложно расширять по горизонтали

3. Событийно-ориентированная архитектура

   Плюсы:

   - Высокая гибкость и слабая связанность между компонентами

- Хорошо подходит для масштабирования и асинхронной обработки
- Упрощает отслеживание бизнес-событий и аудит

Минусы:

- Повышенная сложность отладки и тестирования
- Требует четкой проработки схем событий и механизма доставки
- Возможны трудности с обеспечением согласованности данных

Оценка результативности

Атрибут качества	Метрика	Целевое значение
Согласованность	Время обновления статуса заказа у всех участников	≤ 200 мс
	Частота расхождений между клиентом и админкой	≤ 0.1%
Производительность	Время отклика API	≤ 500 мс
	Время загрузки меню	≤ 1 сек
	Время оформления заказа	≤ 300 мс
Надежность	Процент потерянных заказов	0%
	Частота дублирования заказов	≤ 0.01%
	Успешные транзакции	≥ 99.99%
Отказоустойчивость	Время восстановления после сбоя	≤ 1 мин
	Uptime системы	≥ 99.9%
Удобство использования	Среднее время оформления заказа	≤ 30 сек
	Средняя оценка UX по опросу	≥ 4.5 / 5
Масштабируемость	Кол-во пользователей без деградации	≥ 1000

Результативность будет проверяться по итогам нагрузочного тестирования, UX-оценок и логов ошибок.