|  |  |
| --- | --- |
| ДИСЦИПЛИНА | **Архитектура, проектирование и разработка программных средств** |
|  | (полное наименование дисциплины без сокращений) |
| ИНСТИТУТ | **информационных технологий** |
| КАФЕДРА | **инструментального и прикладного программного обеспечения** |
|  | (полное наименование кафедры) |
| ВИД УЧЕБНОГО | **Материалы для практических/семинарских занятий** |
| МАТЕРИАЛА | (в соответствии с пп.1-11) |
| ПРЕПОДАВАТЕЛЬ | **Алпатов Алексей Николаевич** |
|  | (фамилия, имя, отчество) |
| СЕМЕСТР | **3, 2023-2024** |
|  | (указать семестр обучения, учебный год) |

**Практическая работа №2**

**Атрибутный метод проектирования архитектуры**

**Цель работы**

Цель данной практической работы заключается в знакомстве с

методологией создания программных архитектур на основе методологии

атрибутного проетирования.

Методика **Attribute-Driven Design (ADD)** — это метод проектирования архитектуры программного обеспечения, в котором основное внимание уделяется атрибутам качества (quality attributes), таким как производительность, масштабируемость, безопасность и т.д. Процесс ADD строится вокруг декомпозиции архитектуры в соответствии с этими атрибутами.

**Этапы проектирования архитектуры по методике ADD**

**Шаг 1: Определение входных данных для проектирования**

На этом шаге нужно собрать все исходные данные, которые повлияют на проектирование архитектуры.

**Результаты первой практической работы:**

1. **Функциональные требования**: Модульный подход с функциями учета, инвентаризации и отчетности.
2. **Атрибуты качества**:
   * **Производительность**: Обработка большого количества запросов в реальном времени.
   * **Масштабируемость**: Возможность работы на нескольких складах.
   * **Безопасность**: Ограниченный доступ к данным и защита от утечек.
   * **Интегрируемость**: Интеграция с существующими ERP-системами.
3. **Ограничения**: Сроки 12 месяцев, бюджет 10 млн рублей, команда из 6 человек.

**Шаг 2: Выбор модулей архитектуры**

Основные модули архитектуры WMS:

* Модуль учета товаров
* Модуль инвентаризации
* Модуль управления запасами
* Модуль отчетности
* Модуль интеграции с ERP-системами

**Шаг 3: Выбор атрибутов качества для модулей**

Для каждого модуля необходимо определить критические атрибуты качества:

1. **Модуль учета товаров**:
   * **Производительность**: Высокая скорость обработки запросов на регистрацию товаров.
   * **Масштабируемость**: Поддержка большого количества пользователей.
2. **Модуль инвентаризации**:
   * **Точность**: Минимизация ошибок при учете товаров.
   * **Безопасность**: Ограничение доступа на уровне пользователя.
3. **Модуль управления запасами**:
   * **Масштабируемость**: Возможность расширения с учетом роста складских помещений.
   * **Интегрируемость**: Интеграция с ERP-системами для управления поставками.
4. **Модуль отчетности**:
   * **Производительность**: Генерация отчетов в режиме реального времени.
   * **Масштабируемость**: Возможность работать с большими объемами данных.

**Шаг 4: Выбор тактик для достижения атрибутов качества**

Каждый атрибут качества может быть достигнут с помощью определенных архитектурных решений и тактик. Выбираем тактики для каждого модуля:

1. **Производительность**:
   * Кэширование запросов на уровне базы данных.
   * Асинхронная обработка задач.
2. **Масштабируемость**:
   * Использование микросервисной архитектуры.
   * Автоматическое масштабирование серверных ресурсов.
3. **Интегрируемость**:
   * Использование стандартных API для интеграции с ERP.
   * Модульность системы для облегчения интеграции новых компонентов.
4. **Безопасность**:
   * Внедрение ролей и политик доступа.
   * Шифрование данных.

**Шаг 5: Применение тактик к каждому модулю**

1. **Модуль учета товаров**:
   * Внедрить кэширование данных для быстрого доступа к часто используемым товарам.
   * Разделить нагрузку на сервер с помощью микросервисов для отдельных задач (например, прием товаров, учет перемещений).
2. **Модуль инвентаризации**:
   * Внедрить строгие правила доступа на основе ролей.
   * Добавить механизмы кэширования для ускорения доступа к данным инвентаризации.
3. **Модуль управления запасами**:
   * Использовать API для взаимодействия с внешними системами.
   * Разделить функционал управления запасами на несколько микросервисов для повышения масштабируемости.
4. **Модуль отчетности**:
   * Применить генерацию отчетов в асинхронном режиме для повышения производительности.
   * Использовать кэш для хранения промежуточных данных отчетов.

**Шаг 6: Декомпозиция системы**

На этом этапе выполняется детализированное разбиение системы на модули и компоненты. Это включает:

1. Разбиение каждого модуля на функциональные блоки (например, прием, размещение, инвентаризация и т.д.).
2. Определение взаимодействия между компонентами через API.
3. Уточнение технологий, которые будут использоваться для реализации модулей (база данных, инструменты DevOps, средства интеграции).

**Шаг 7: Проверка соответствия атрибутам качества**

После выбора тактик необходимо проверить, соответствуют ли они заявленным атрибутам качества. Это можно сделать с помощью анализа или проведения тестов на прототипах:

1. **Производительность**: Тестирование системы с имитацией большого количества пользователей.
2. **Масштабируемость**: Проверка на возможность добавления новых складов и пользователей.
3. **Безопасность**: Проведение аудита безопасности на всех этапах разработки.

**Шаг 8: Повторение цикла**

После завершения первого итерационного цикла необходимо провести ревизию результатов и, при необходимости, вернуться на предыдущие шаги для улучшения архитектурных решений. Например:

* Если система не справляется с нагрузками, нужно пересмотреть решения по производительности.
* Если интеграция с ERP-системой занимает больше времени, возможно, потребуется доработать API или изменить логику интеграции.

**Вывод**

Итерационный процесс проектирования архитектуры с использованием ADD позволяет гибко адаптировать систему под требования к атрибутам качества. Каждый цикл дает возможность совершенствовать архитектурные решения и детализировать структуру системы с учетом бизнес-требований и технологических ограничений.