**Отчет по проекту "Зоопарк" на языке Golang**

**Введение**

Краткое описание проекта

Проект "Зоопарк" представляет собой программное обеспечение для управления зоопарком, которое позволяет следить за животными, сотрудниками, вольерами и посетителями. Система обеспечивает администрирование, учет кормления животных, медицинское обслуживание и организацию визитов.

**Цели проекта**

Основные цели проекта:

- Эффективное управление ресурсами зоопарка.

- Обеспечение удобного интерфейса для сотрудников.

- Автоматизация процессов, связанных с уходом за животными.

- Улучшение пользовательского опыта для посетителей.

**Анализ предметной области**

Идентификация основных сущностей

1. Животные: представляют собой основные объекты, содержащие информацию о видах, возрасте, здоровье и других характеристиках.

2. Сотрудники: включают охранников, ветеринаров и обслуживающий персонал, которые взаимодействуют с животными.

3. Вольеры: служат местом обитания животных и содержат информацию о размере, типе и состоянии.

4. Посетители: пользователи системы, которые могут покупать билеты, записываться на экскурсии и получать информацию о животных.

Описание их свойств и взаимосвязей

- Животные имеют связь с вольерами (каждое животное помещается в определенный вольер).

- Сотрудники могут взаимодействовать с животными для их ухода и лечения.

- Посетители могут взаимодействовать с сотрудниками для получения информации и услуг.

Выявление динамических аспектов системы

- Кормление животных: расписание и учет кормов.

- Лечение: записи о медицинских процедурах и состояниях здоровья.

- Посещения: управление экскурсиями и билетами.

**Паттерн "Фабрика"**

Обоснование выбора

- Проблемы, решаемые паттерном: упрощение создания объектов, таких как животные или вольеры, за счет абстракции процесса создания.

- Улучшение структуры кода: уменьшение связанности между классами и упрощение тестирования.

- Альтернативные паттерны: Паттерн "Строитель" рассматривался, но был отвергнут из-за большей сложности в реализации.

**Паттерн "Стратегия"**

Обоснование выбора

- Проблемы, решаемые паттерном: необходимость в изменении поведения объектов во время выполнения. В контексте зоопарка это может быть связано с различными способами ухода за животными или методами кормления, которые могут варьироваться в зависимости от вида животного.

- Улучшение структуры кода: повышение гибкости и повторного использования кода. Каждый способ ухода за животными может быть реализован как отдельная стратегия, что упрощает добавление новых методов.

- Альтернативные паттерны: Паттерн "Команда" был рассмотрен, но не был выбран, так как не обеспечивал необходимой гибкости в изменении поведения.

**Реализация системы**

**Основные компоненты**

1. Модель данных

- Определение структур для животных, сотрудников, вольеров и посетителей.

- Использование struct для создания сущностей с необходимыми полями и методами.

2. Слой доступа к данным

- Реализация CRUD-операций для каждой сущности.

- Использование встроенных механизмов Go для работы с базами данных (например, database/sql).

3. Логика бизнес-процессов

- Реализация алгоритмов для работы с животными, сотрудниками и посетителями.

- Включение валидации данных и бизнес-правил.

4. Интерфейс пользователя

- Создание простого командного интерфейса для взаимодействия с системой.

- Возможное использование веб-фреймворков для создания более сложного интерфейса (например, Gin или Echo).

**Тестирование**

Подходы к тестированию

- Юнит-тестирование: написание тестов для каждой функции и метода.

- Интеграционное тестирование: проверка взаимодействия между компонентами системы.

- Тестирование производительности: оценка скорости выполнения операций, особенно в контексте работы с базой данных.

**Заключение**

Итоговая оценка проекта

Проект "Зоопарк" представляет собой полноценное решение для управления зоопарком, которое эффективно использует язык Go и паттерны проектирования для создания гибкой и масштабируемой системы. Реализация проекта способствует улучшению взаимодействия между сотрудниками и животными, а также повышает уровень обслуживания посетителей.