Лабораторная работа №3

По дисциплине «Автоматизация научных исследований»

Тема: Генерация диаграмм UML с помощью ИИ-моделей

1. Выбранная предметная область

Объектом исследования является **Система автоматизации заказа еды в ресторане**. Проект направлен на оптимизацию процесса взаимодействия клиента с заведением путём внедрения цифрового меню и системы самостоятельного оформления заказов. Это позволяет минимизировать ошибки персонала (официантов), ускорить передачу заказа на кухню и повысить прозрачность статуса готовности для клиента.

2. Действующие лица (Актеры)

На основе анализа пользовательских историй были выделены три ключевые роли:

1. **Посетитель:** Конечный пользователь, который взаимодействует с цифровым меню, формирует корзину, оформляет заказ и отслеживает его статус.
2. **Сотрудник кухни (Повар):** Взаимодействует с системой для получения информации о новых заказах и изменения их статуса (в работе / готов).
3. **Администратор меню:** Сотрудник, ответственный за актуализацию перечня блюд, цен и доступности позиций в системе.

3. Функциональные требования

Основные функции системы, подлежащие моделированию:

- Просмотр цифрового меню по категориям.
- Управление корзиной (добавление, изменение количества).
- Оформление заказа с выбором способа получения (в зале / навынос).
- Мониторинг статуса заказа в реальном времени.
- Управление очередью заказов на стороне кухни.
- Модерация контента (блюд) администратором.

4. Часть 1а. Диаграмма вариантов использования (Use Case Diagram)

Описание диаграммы

Диаграмма отображает границы системы и ключевые взаимодействия актеров с функциональными блоками.

- Связь **<<include>>** использована для случая «Оформить заказ», так как проверка состава корзины является обязательной частью процесса.

- Связь **<<extend>>** применена для выбора способа получения заказа (опциональное уточнение при оформлении).

Сгенерированный код для PlantUML

Фрагмент кода

```
@startuml
skinparam actorStyle awesome
left to right direction

actor "Посетитель" as Visitor
actor "Сотрудник кухни" as Kitchen
actor "Администратор меню" as Admin

rectangle "Система заказа еды" {
  usecase "Просмотр меню" as UC1
  usecase "Просмотр карточки блюда" as UC2
  usecase "Управление корзиной" as UC3
  usecase "Оформить заказ" as UC4
  usecase "Выбор способа получения" as UC5
  usecase "Отслеживание статуса" as UC6
  usecase "Обновление статуса заказа" as UC7
  usecase "Управление блюдами" as UC8
  usecase "Управление доступностью" as UC9

  Visitor -- UC1
  Visitor -- UC3
  Visitor -- UC4
  Visitor -- UC6

  UC1 <.. UC2 : <<extend>>
  UC4 ..> UC3 : <<include>>
  UC4 <.. UC5 : <<extend>>

  Kitchen -- UC7

  Admin -- UC8
  Admin -- UC9
}

@enduml
```

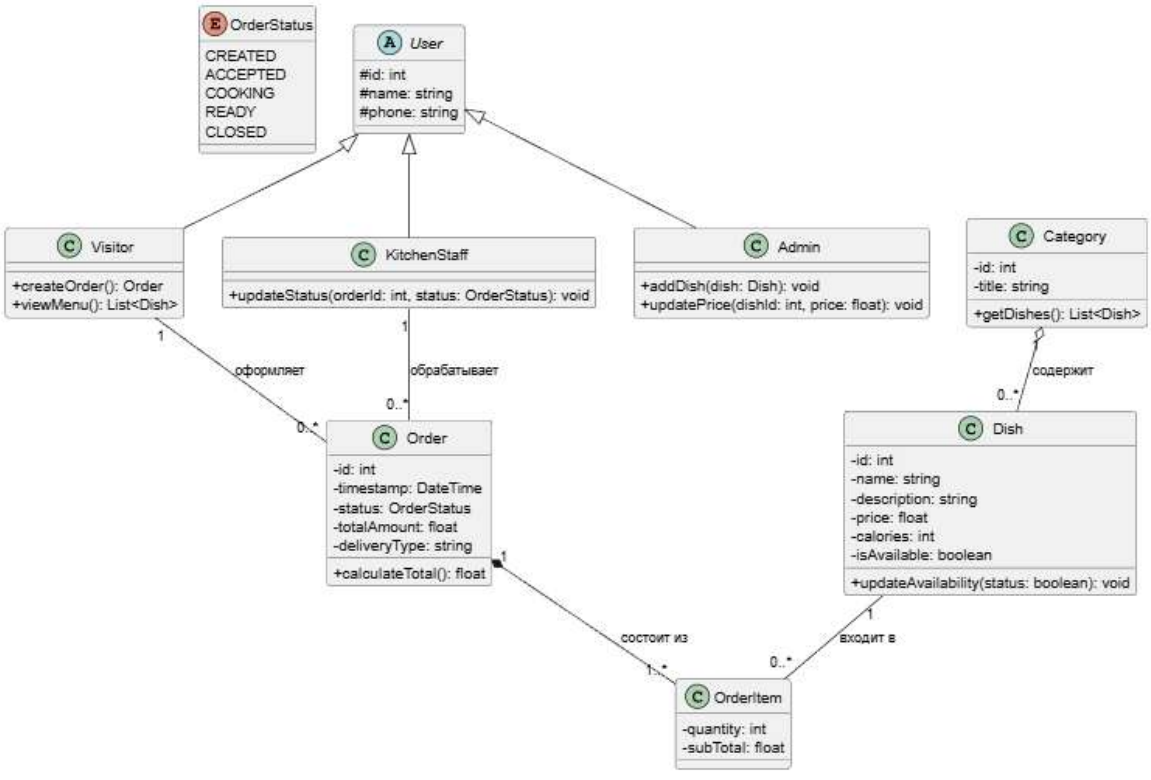
5. Часть 1б. Диаграмма классов (Class Diagram)

Сгенерированный код для PlantUML

Фрагмент кода

```
@startuml
skinparam classAttributeIconSize 0

enum OrderStatus {
  CREATED
  ACCEPTED
  COOKING
  READY
  CLOSED
}

abstract class User {
  # id: int
  # name: string
  # phone: string
}

class Visitor extends User {
  + createOrder(): Order
  + viewMenu(): List<Dish>
}

class KitchenStaff extends User {
  + updateStatus(orderId: int, status: OrderStatus): void
}

class Admin extends User {
  + addDish(dish: Dish): void
  + updatePrice(dishId: int, price: float): void
}

class Category {
```

```
  - id: int
  - title: string
  + getDishes(): List<Dish>
}

class Dish {
  - id: int
  - name: string
  - description: string
  - price: float
  - calories: int
  - isAvailable: boolean
  + updateAvailability(status: boolean): void
}

class Order {
  - id: int
  - timestamp: DateTime
  - status: OrderStatus
  - totalAmount: float
  - deliveryType: string
  + calculateTotal(): float
}

class OrderItem {
  - quantity: int
  - subTotal: float
}

' Relationships
Category "1" o-- "0..*" Dish : содержит
Order "1" *-- "1..*" OrderItem : состоит из
Dish "1" -- "0..*" OrderItem : входит в
Visitor "1" -- "0..*" Order : оформляет
KitchenStaff "1" -- "0..*" Order : обрабатывает

@enduml
```



6. Анализ результатов проектирования

В ходе выполнения лабораторной работы была проведена автоматизация этапа проектирования информационной системы с использованием модели искусственного интеллекта.

Основные выводы по результатам проектирования:

1. **Соответствие требованиям:** Сгенерированные диаграммы полностью покрывают сценарии, описанные в документах «Видение проекта» и «Бизнес-процессы», которые прикреплялись к промпту.
2. **Эффективность ИИ:** Использование нейросети для генерации UML-кода позволило сократить время на техническое рисование диаграмм. Модель корректно интерпретировала текстовые описания и перевела их в строгие нотации классов и связей.
3. **Логическая целостность:** Архитектура классов предусматривает масштабируемость (например, легкое добавление новых ролей пользователей или типов оплаты). Использование стандартов PlantUML обеспечило чистоту и читаемость схем.
4. **Исследовательский аспект:** В рамках ЛР было протестировано, насколько точно ИИ может соблюдать специфические требования UML (например, различия между композицией и агрегацией). Результат показал высокую точность в определении связей как композиции, что критически важно для целостности данных БД.