Курсовой проект

по разработке информационной системы для сотрудников фитнес-клуба

ФМФИТ, КИ, МОКС, III курс

Воронич Мария

Цель

Разработка информационной системы для предметной области фитнес-клуб:

- организация учета сотрудников, трудовых договоров, оформленных при найме сотрудников;
- предоставление необходимой информации сотрудникам;
- оформления абонементов;
- создание расписаний спортивных занятий;
- контроль за тренерами, их деятельностью;
- модифицикация данных о фитнес-клубе.

Задачи для достижения цели

- Выполнить анализ предметной области;
- Определить требования к ИС;
- Разработать архитектуру системы;
- Разработать систему классов, реализующие доступ к данным фитнес-клуба;
- Реализовать элементы интерфейса, дающие возможность удобного просмотра, редактирования и удаления данных.
- Реализовать клиентское приложение, которое предоставит возможность просмотра и редактирования данных фитнес-клуба.
- Обеспечить разграничение доступа со стороны различных категорий пользователей.

Задачи пользователей

1) Тренер фитнес-клуба

- ❑Просмотр персональной информации клиентов, имеющихся услуг, расписаний, типов абонементов
- ❑Регистрация нового клиента
- ❑Оформление абонементов
- ❑Создание нового расписания занятий для клиентов
- ❑Продление срока действия абонементов
- ❑Удаление расписания

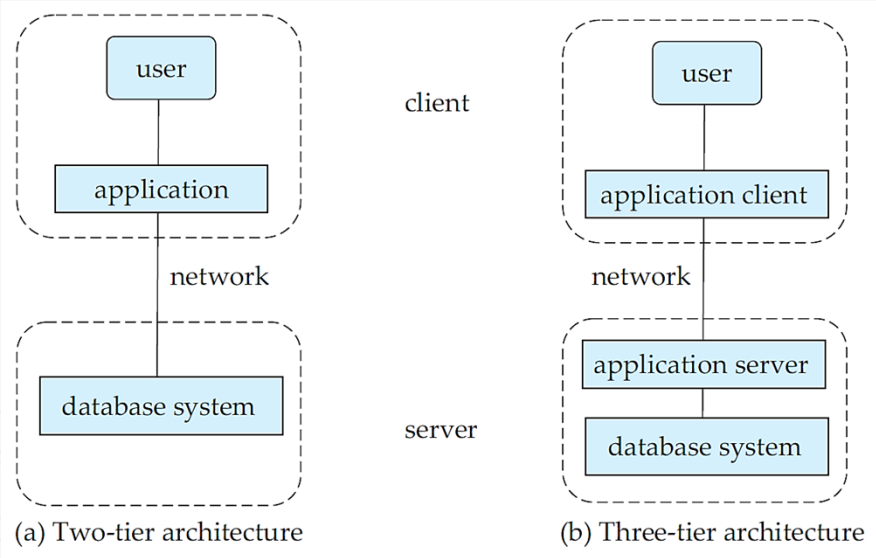
2) Администратор фитнес-клуба

- ❑Оформление нового сотрудника на работу
- ❑Увольнение сотрудника
- ❑Добавление новой услуги
- ❑Просмотр персональной информации сотрудников

Проектирование: Двухуровневая архитектура

- **Двухуровневая архитектура (клиент-сервер)**
- **«Толстый» клиент – отображение данных, содержит бизнес-логику, обработка информации напрямую из БД**

- Трехуровневая архитектура
- «Тонкий» клиент – отображение данных, обращение к серверу приложений, а не БД
- Бизнес-логику содержит сервер приложений.



(a) Two-tier architecture (b) Three-tier architecture

Паттерн: MVP

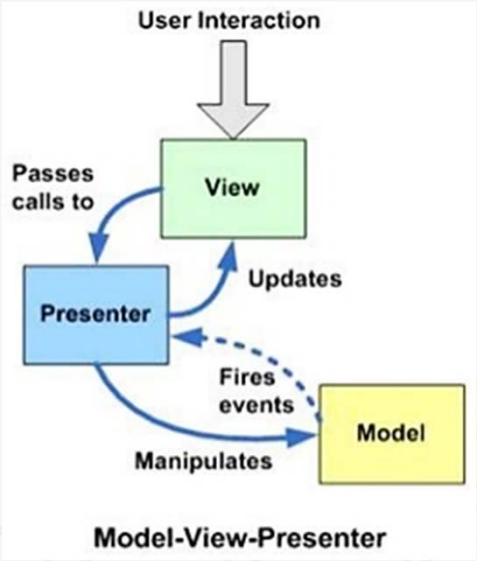
MVP

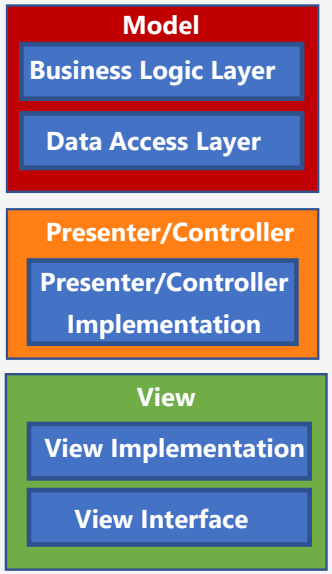
- Desktop-приложения

- Model — хранит в себе всю бизнес-логику, при необходимости получает данные из хранилища.

- View — реализует отображение данных (из Model), обращается к Presenter за обновлениями.

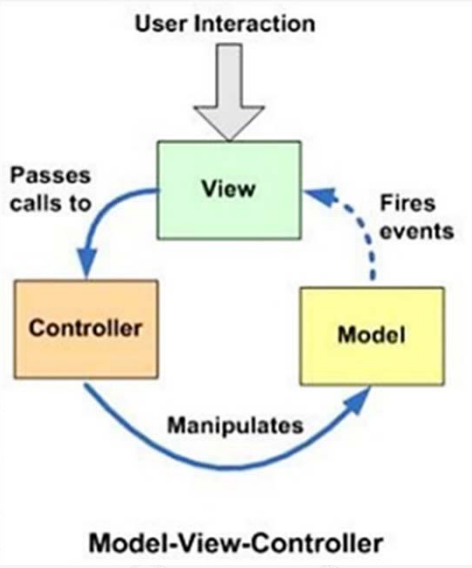
- Presenter — реализует взаимодействие между Model и View.





MVC

- Web-приложения
- Model - предоставляет данные и реагирует на команды контроллера, изменяя своё состояние.
- View - отвечает за отображение данных модели (напр. HTML), реагируя на изменения модели.
- Controller - интерпретирует действия пользователя, оповещая Model о необходимости изменений.



ПО

- СУБД – Postgres, разработка в PgAdmin
- Клиентское приложение – C#, библиотеки: Npgsql, Windows Forms
- Среда программирования – Microsoft Visual Studio Community 2019

ER-диаграмма

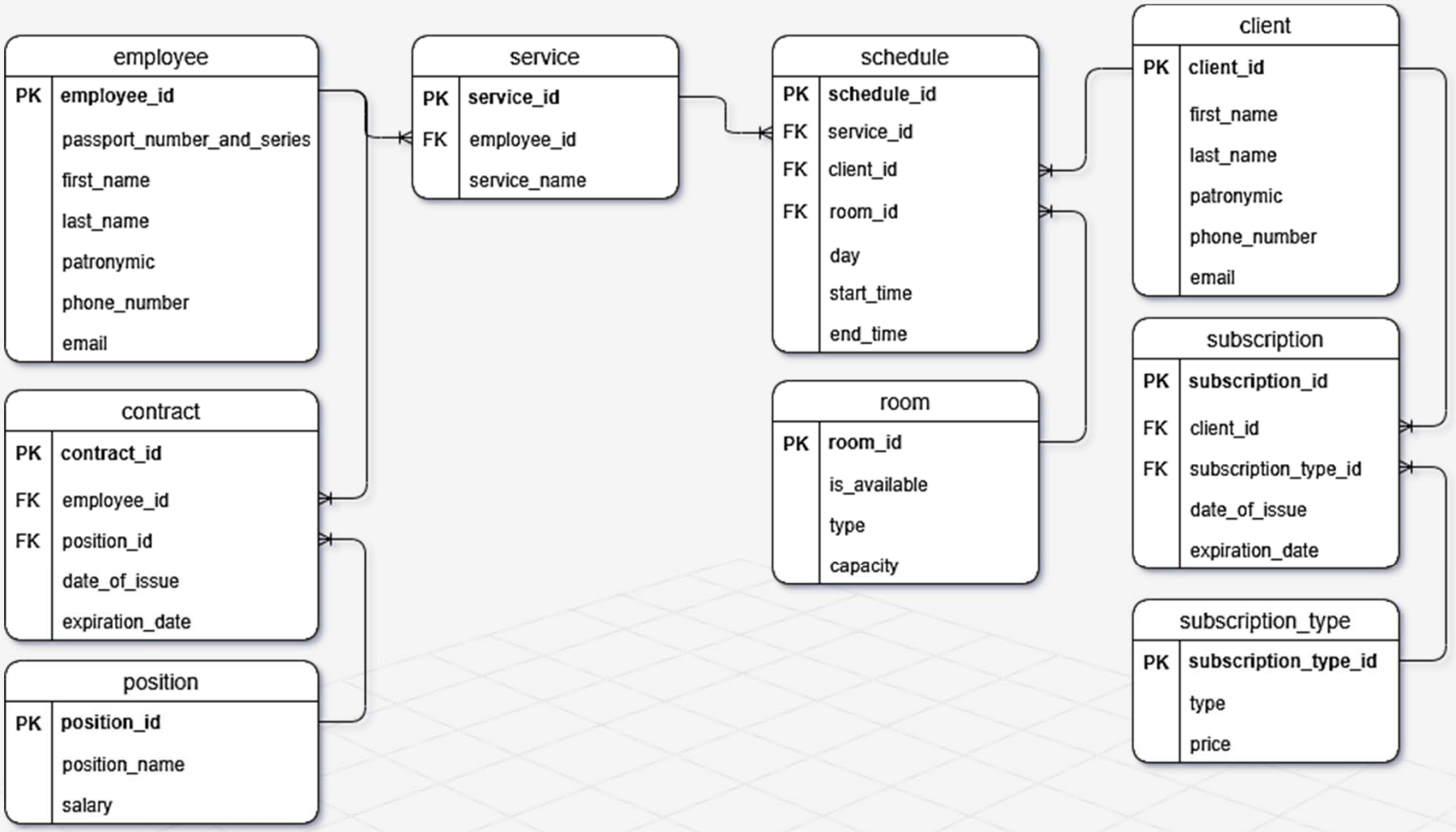


Диаграмма классов