# ОПИСАНИЕ ПРИМЕНЕНИЯ ПАТТЕРНОВ ПРОЕКТИРОВАНИЯ

В целях оптимизации разрабатываемого кода мною были выбраны следующие паттерны проектирования:

* Модель-Представление-Контроллер (MVC) для оптимизации структуры проекта в целом;
* Заместитель (Proxy) для улучшенного контроля доступа в БД;
* Посредник (Mediator) для связи серверной части приложения с клиентом.

Шаблон Модель-Представление-Контроллер представляет собой разделение данных приложения, пользовательского интерфейса и логики приложения между собой. Суть данного паттерна в том, чтобы каждый из компонентов программы: модель, представление и контроллер могли существовать независимо друг от друга. Более того каждый компонент может изменяться не затрагивая другие компоненты и не меняя при этом общей структуры проекта. На рисунке 7.1 представлена общая структура паттерна «Модель-Представление-Контроллер».



Рисунок 7.1 – Структура шаблона «Модель-Представление-Контроллер»

Данный паттерн очень популярен среди веб-разработчиков. Многие фреймворки, в том числе и Ruby on Rails, построены на основе данного шаблона, что позволяет веб-разработчикам строить распределенные приложения быстро и качественно.

Паттерн Заместитель, с английского Proxy, - это структурный шаблон проектирования, который необходим для контроля доступа к другому объекту. В разрабатываемом приложении данный паттерн используется для контроля доступа к объектам базы данных. Класс ApplicationRecord и его наследники Train, Way, Ticket и User представляют собой так называемые Proxy-классы и перехватывают все попытки доступа к базе. Такой подход позволяет разместить базу на удаленном сервере, а также контролировать запросы и отправлять непосредственно в базу только заведомо корректные.

Достоинства шаблона Заместитель:

* удаленный доступ;
* виртуальный заместитель;
* защищает доступ к объектам.

Недостатки шаблона Заместитель:

* увеличенное время доступа к объектам.

Поведенческий паттерн Посредник, или Mediator, необходим для связи компонентов приложения, при этом не вынуждая их явно ссылаться друг на друга. Так паттерн медиатор инкапсулирует взаимодействие объектов, выступая при этом посредником между ними.

На рисунке 7.2 представлена структура паттерна Посредник.

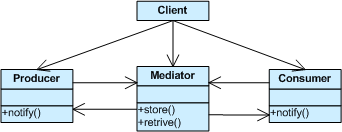


Рисунок 7.2 – Структура паттерна Посредник

На рисунке 7.3 представлена диаграмма классов системы управления билетно-кассовыми операциями на железнодорожной станции, демонстрирующая применение описанных выше паттернов проектирования.

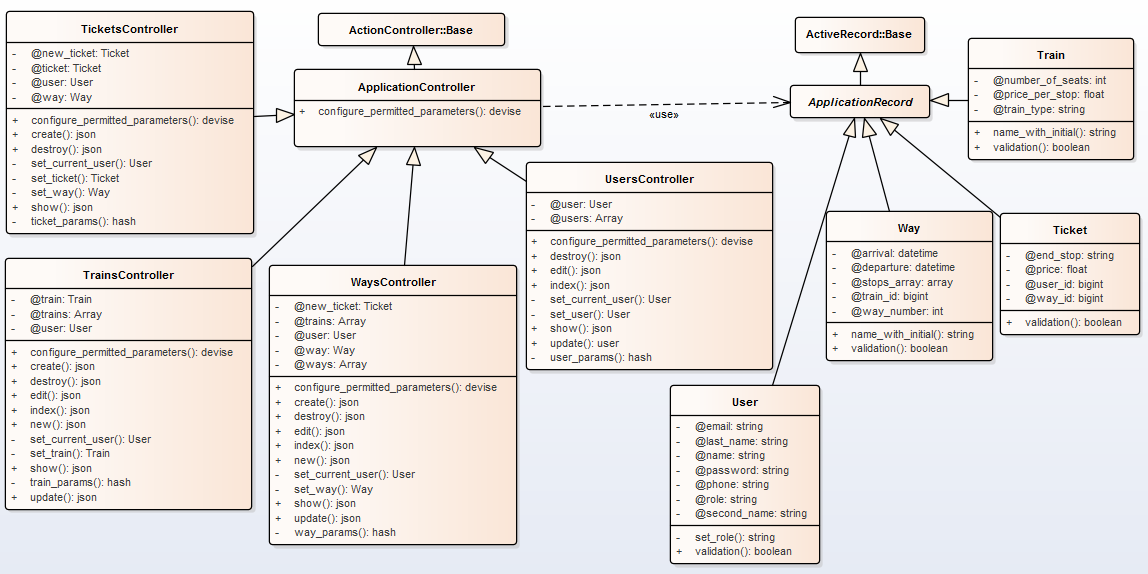


Рисунок 7.1 – Диаграмма классов системы управления билетно-кассовыми операциями на железнодорожной станции