Вариант 28

Учащийся группы 581061 Шейна Александр

Постановка задачи:

Вы работаете в парикмахерской.

Ваша парикмахерская стрижет клиентов в соответствии с их пожеланиями и некоторым каталогом различных видов стрижки. Так, для каждой стрижки определены название, принадлежность полу (мужская, женская), стоимость работы. Для наведения порядка Вы, по мере возможности, составляете базу данных клиентов, запоминая их анкетные данные (фамилия, имя, отчество). Начиная с 5-ой стрижки, клиент переходит в категорию постоянных и получает скидку в 3% при каждой последующей стрижке. После того, как закончена очередная работа, в кассе фиксируются стрижка, клиент и дата производства работ.

Таблицы

Стрижки (Код стрижки, Название, Пол, Стоимость).

Клиенты (Код клиента, Фамилия, Имя, Отчество, Пол, Признак постоянного клиента).

Работа (Код работы, Код стрижки, Код клиента, Дата).

**1 Диаграмма вариантов использования**

Понятие варианта использования, или прецедента (Use Case) является основным элементом разработки и планирования проекта. Вариант использования представляет собой последовательность действий (транзакций), выполняемых системой в ответ на событие, инициируемое некоторым внешним объектом (действующим лицом). Прецедент описывает типичное взаимодействие между пользователем и системой.

Действующее лицо или актёр (Actor) – это роль, которую пользователь играет по отношению к системе. Действующим лицом может выступать не только пользователь, но также другая система, взаимодействующая с данной, и время. Время становится действующим лицом, если от него зависит запуск какого-либо события в системе.



Рисунок 1 – диаграмма вариантов использования

Действующими лицами в данной системе являются «Клиент», «работник» и «системаОплаты». Проектируемая система «Парикмахерская» должна выполнять следующие действия: просмотреть каталог, выбрать стрижки, предоставить информацию, оплатить, сформировать каталог, рассчитать скидку, оказать услугу, принять оплату. На диаграмме показано взаимодействие между вариантами использования и действующими лицами, отражены требования к системе с точки зрения пользователя. Можно сказать что варианты использования – это функции выполняемые системой, а актеры – заинтересованные лица, инициирующие варианты использования, либо получающие от него информацию.

**2 Диаграмма последовательности**

На диаграмме последовательностей основное внимание уделяется временной упорядоченности сообщений. На ней изображаются объекты, которые непосредственно участвуют во взаимодействии, и сообщения, которыми эти объекты обмениваются.



Рисунок 2 – Диаграмма последовательностей

На данной диаграмме изображена последовательность действий актеров данной предметной области. На первом шаге пришедший клиент, регистрируется работником, после чего как отображено на шаге 2 клиент выбирает тип стрижки, работник принимает заказ (шаг 3). По окончании оказании услуг (шаг 4), клиент оплачивает услуги как видно из шага 5.

**3 Диаграмма коопераций**

Диаграмма коопераций, в отличие от диаграммы последовательности, акцентирует внимание на организации объектов, принимающих участие во взаимодействии. Она представляется в виде графа, вершины которого – объекты, участвующие во взаимодействии, а дуги – связи, соединяющие объекты. Связи дополняются сообщениями, которые объекты принимают и посылают. Таким образом, диаграмма коопераций даёт визуальное представление о потоке управления в контексте структурной организации кооперирующихся объектов.

Рисунок 3 – диаграмма коопераций

**4 Диаграмма классов**

Диаграмма классов определяет структуру проекта, типы классов системы, а также различного рода связи, существующие между ними.

На диаграмме UML класс отображается в виде прямоугольника. Имя класса – текстовая строка. Имя абстрактного класса выделяется курсивом.

Атрибут – это именованное свойство класса, включающее описание множества значений, которые могут принимать экземпляры этого свойства. Он представляет собой свойство моделируемой сущности, общее для всех объектов данного класса.

Операции реализуют связанное с классом поведение, иными словами, абстракцию того, что позволено делать с объектом.

Рисунок 4 – Диаграмма классов

На данной диаграмме можно увидеть классы с их полями и методами, относятся к постановке задачи, такие как: «Клиент», «работник», «Стрижка», «системаОплаты».

**5 Диаграмма состояний**

Диаграммы состояний (State Machine Diagrams) отражают все возможные состояния, в которых может находиться конкретный объект, а также процесс смены состояний объекта в результате некоторых событий.



Рисунок 5 – Диаграмма состояний

**6 Диаграмма компонентов**

Диаграмма компонентов описывает особенности физического представления системы. Она позволяет определить архитектуру разрабатываемой системы, установив зависимости между программными компонентами, в роли которых могут выступать исходный и исполняемый код. Основными графическими элементами диаграммы компонентов являются компоненты, интерфейсы и зависимости.



Рисунок 6 – Диаграмма компонентов

На данной диаграмме отображены компоненты программного средства, такие как файлы классов программы и файл СУБД MS SQL Server 2008.

**7 Диаграмма развертывания**

Диаграммы развертывания предназначены для представления общей конфигурации и топологии распределенной программной системы. Диаграмма развертывания служит средством визуализации элементов и компонентов программы, существующих лишь на этапе ее исполнения (runtime). При этом представляются только компоненты-экземпляры программы, являющиеся исполняемыми файлами или динамическими библиотеками, а компоненты, которые не используются на этапе исполнения, например, тексты программ, на диаграмме развертывания не показываются.



Рисунок 7 – Диаграмма развертывания