МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

«КУБАНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

(ФГБОУ ВО «КубГУ»)

**Факультет компьютерных технологий и прикладной математики**

**Кафедра информационных технологий**

**ОТЧЕТ**

**О выполнении лабораторной работы №4**

**по дисциплине**

**«Технологии проектирования программного обеспечения»**

Работу выполнил студент группы 4ММ\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Н.П. Демьяненко

Работу принял Доцент кафедры И.Т.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ А.Н. Полетайкин

Краснодар

2023

Тема: Проектирование функциональной структуры программного продукта: объектно-ориентированный подход.

Цель: изучение методики объектно-ориентированного подхода программной инженерии для разработки и описания функциональности разрабатываемого программного обеспечения.

Индивидуальная тема: ИС приема и выполнения заказов фотоцентра.

Задание

1. Проанализировать описание функционирования программной системы, разработанного при выполнении лабораторной работы №4, на предмет выявления набора абстракций предметной области проектируемой ПС. В качестве предварительных кандидатов в абстракции принять подлежащие, выделенные из текста анализируемого потока событий.

2. Разделить выделенные абстракции на три типа: абстракции сущности, абстракции поведения, абстракции интерфейсы. Результат представить в виде таблицы. Для каждой абстракции указать ее класс согласно следующей классификации: − Люди − Места − Предметы − Инструменты − Организации − Концепции − События − Показатели

3. Проанализировать поведение выделенных абстракций. Выделить возможное поведение каждой абстракции в пределах функциональности проектируемой ИС, представленной моделью требований UML (рис. 7). Заполнить таблицу 5. Таблица 5. Абстракции подсистемы и их поведение № Абстракция Требование согласно модели UML Описание поведения

4. Построить диаграмму классов UML (class diagram), указывая при этом лишь имена классов без указания свойств класса.

5. На основе анализа описания предметной области, разработанного при выполнении лабораторной работы №1, выявить атрибуты и операции классов. Заполнить секции атрибутов и операций классов.

6. Выбрать в модели классов такой класс, которых характеризуется наиболее частой сменой состояний, и построить для него диаграмму состояния (statechart diagram).

7. На основе анализа функциональных моделей, разработанных при выполнении лабораторной работы №4, для каждого из базовых вариантов использования построить диаграмму деятельности (activity diagram). Для вариантов использования, с которыми связаны несколько действующих лиц, диаграмму деятельности построить в виде дорожек с привязкой к исполнителям конкретных операций алгоритма

8. Для каждого варианта использования выделить список объектов участвующих во взаимодействии в этом прецеденте, заполнить таблицу 6. Таблица 6. Список объектов для каждого потока событий № п.п. Прецедент Объект Описание объекта

9. Создать диаграммы последовательности (sequence diagram) для перечисленных прецедентов (одну диаграмму для всех объектов из табл. 6).

10. Для наиболее сложных диаграмм последовательности создать кооперативные диаграммы (collaboration diagram) и доработать их, если это необходимо.

**Задание 1**

Абстракции подсистемы

|  |  |  |  |
| --- | --- | --- | --- |
| № | Абстракция | Тип | Описание |
| 1 | Кассир | сущность | Пользователь системы, взаимодействующий с клиентом и системой |
| 2 | Фотолаборант | сущность | Пользователь системы, выполняющий заявки кассира |
| 3 | Товаровед | сущность | Пользователь системы, ответственный за выдачу материалов для работы фотолаборанта |
| 4 | Вход | поведение | Проведение аутентификации пользователя |
| 5 | Отправка сообщения | поведение | Формирование на основе полученной информации сообщения. |
| 6 | Получение информации о собеседниках | поведение | Получение основной информации о собеседниках, а именно: имя, фамилия, номер телефона |
| 7 | Отображение сообщений | поведение | Получение информации о сообщении: текст сообщения, время отправки, отправитель |
| 8 | Просмотр профиля | поведение | Получение информации о пользователе: имя, фамилия, пол, номер телефона, возраст, город проживания |

**Задание 2**

Классификация абстракции

|  |  |  |
| --- | --- | --- |
| № | Класс | Список абстракций |
| 1 | Люди | Кассир, Фотолаборант , Товаровед |
| 2 | Предметы |  |
| 3 | События |  |

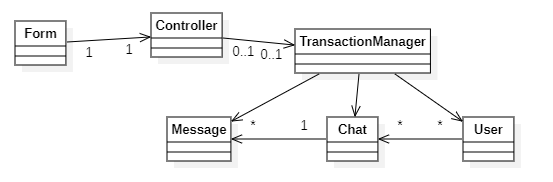
**Задание 3**

Абстракции подсистемы и их поведение

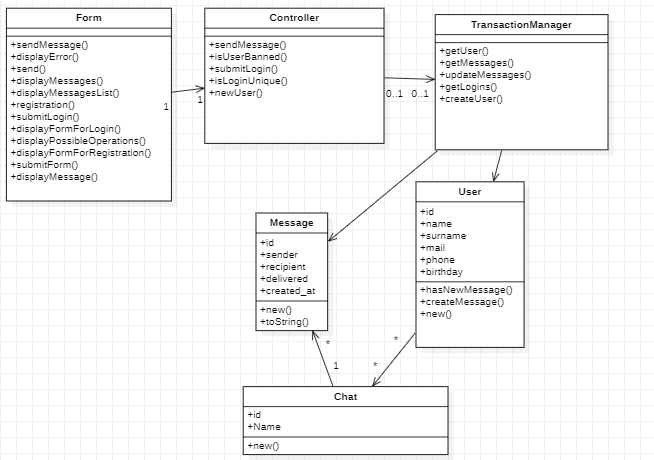
|  |  |  |  |
| --- | --- | --- | --- |
| № | Абстракция | Поведение | Описание поведения |
| 1 | Пользователь | Предоставить,  Получить,  отправить | Предоставление пользователем личной информации, получить сообщение, отправить сообщение |
| 2 | Сообщение | Ввести | Ввод сообщения пользователем |
| 3 | Отправка сообщения | Сформировать | Формирование пакета с сообщением на основе предоставленной информации |
| 4 | Информация о сообщении | Получить | Получение пользователем информации о сообщении |
| 5 | Информация о собеседниках | Получить | Получение пользователем информации о собеседниках |
| 6 | Просмотр профиля | Предоставить | Пользователю предоставляется информация о другом пользователе |
| 7 | Регистрация | Получить | Получение данных пользователя и занесение их в БД |
|  | Вход | Выполняет | Пользователь производит вход в систему |

**Задание 4**

Диаграмма классов



**Задание 5**



При разработке информационной системы использовался

архитектурный приём MVC – «модель-представление-контроллер» (Model,

View, Controller). Это способ организации кода, который предполагает

выделение блоков, отвечающих за решение разных задач. Один блок

отвечает за данные приложения, другой отвечает за внешний вид, а третий

контролирует работу приложения.

Компоненты MVC:

Модель — этот компонент отвечает за данные, а также

определяет структуру приложения.

Представление — этот компонент отвечает за взаимодействие с

пользователем. То есть код компонента view определяет внешний вид

приложения и способы его использования.

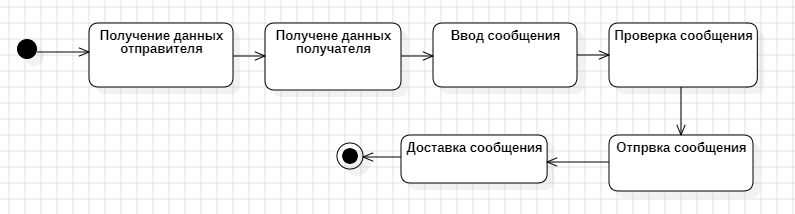
Контроллер — этот компонент отвечает за связь

Между model и view. Код компонента controller определяет, как сайт

реагирует на действия пользователя.

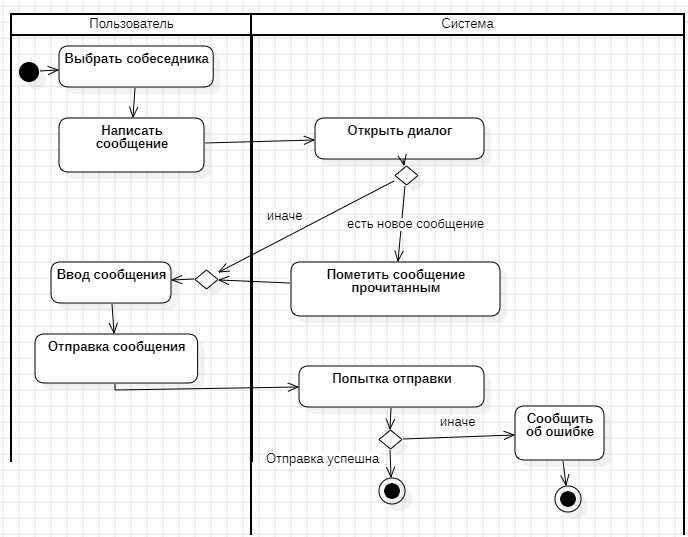
**Задание 6**

Диаграмма состояний класса «сообщения»

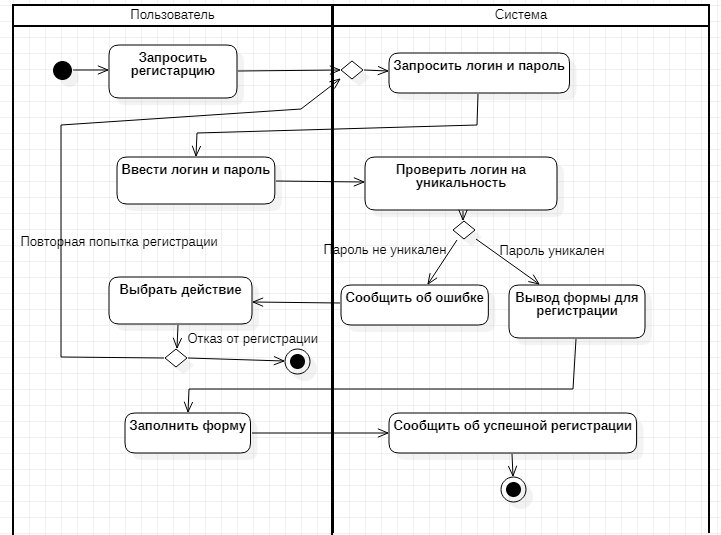


**Задание 7**

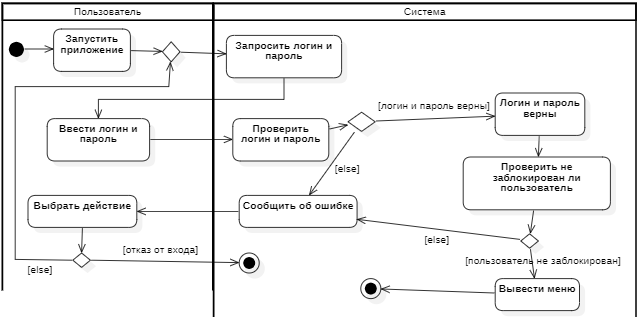
Диаграмма деятельности «Отправка сообщения»



Регистрация



Вход



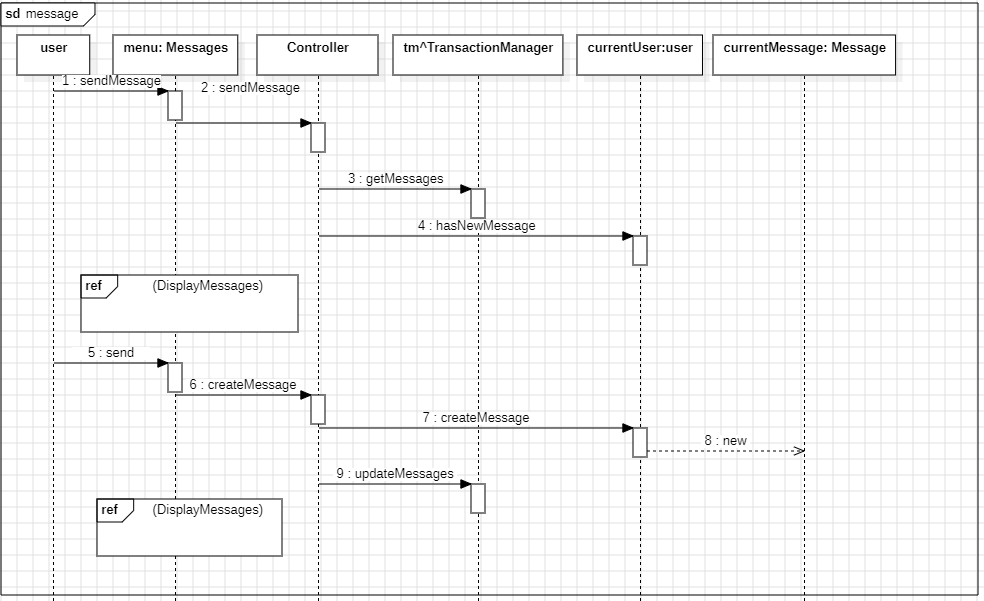
**Задание 8**

Список объектов для каждого потока событий

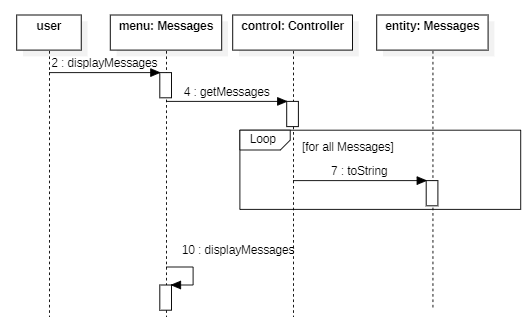
|  |  |  |  |
| --- | --- | --- | --- |
| № п.п | Прецедент | Объект | Описание объекта |
| 1 | Отправить сообщение | Модуль отправки сообщения | Часть системы, ответственная за отправку сообщения |
| 2 | Зарегистрировать пользователя | Модуль регистрации пользователя | Часть системы, ответственная за регистрацию пользователей |

**Задание 9**

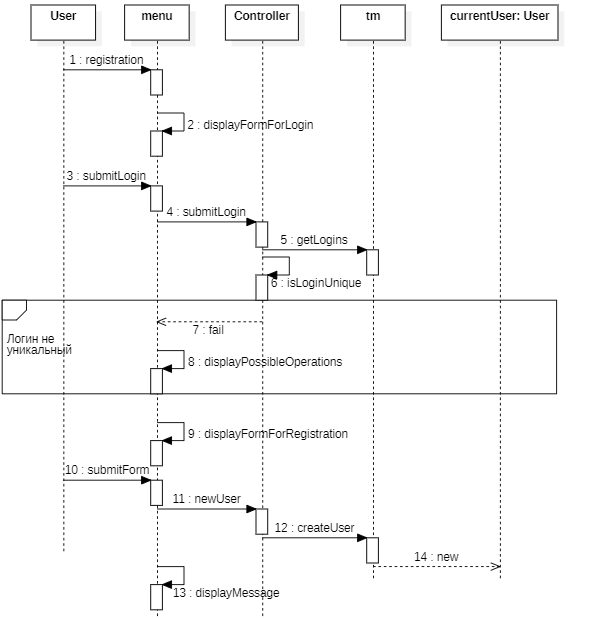
Диаграмма последовательности



Фрагмент с оператором взаимодействия ref ссылается на диаграмму последовательности Display Messages. Поскольку вывод списка сообщений возникает в нескольких подчинённых потоках, есть смысл смоделировать его однажды, и многажды ставить ссылку на взаимодействие, моделирующее его.



Регистрация



Вход 