**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ**

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

**«Национальный исследовательский ядерный университет «МИФИ»**

**Обнинский институт атомной энергетики –**

**филиал федерального государственного автономного образовательного учреждения высшего образования «Национальный исследовательский ядерный университет «МИФИ»**

**(ИАТЭ НИЯУ МИФИ)**

Отделение интеллектуальных кибернетических систем

**Отчёт по лабораторной работе №1**

по дисциплине:

«Технологии программирования»

На тему: «Web-форум»

Выполнили студенты: Старков А.

Группа: ИВТ-Б19

Дата сдачи на проверку:

Проверил: доцент Национального Исследовательского Ядерного Университета «МИФИ» (НИЯУ МИФИ), преподаватель ОИКС В.П. Тельнов

Дата защиты работы: \_\_\_. \_\_\_. 2022 г.

Оценка: «\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_»

Обнинск – 2022

# Постановка задачи

Разработать в *CASE*-средстве *Visual Paradigm* спецификацию требований и технический проект программного обеспечения для предметной области «Web-форум».

«Видеопрокат» ‒ система управления информацией об обсуждениях на форуме.

Поддерживаемые данные:

1. Пользователь

* Login/пароль
* Дата регистрации
* Права – пользователь или модератор

1. Тема

* Раздел форума
* Сообщения по теме

Пользователь, приславший сообщение

Заголовок сообщения

Дата и время поступления

Прикрепленные файлы

Поддерживаемые операции:

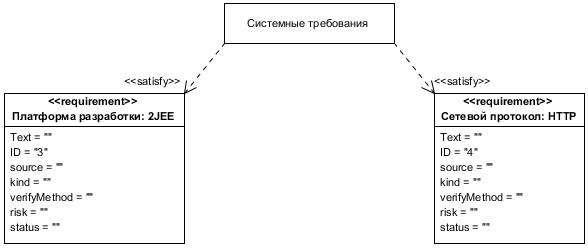
1. Получение списка пользователей, в т.ч. по участию в различных разделах и по активности (количеству сообщений в заданном интервале времени)
2. Получение списка разделов, тем в разделе, сообщений в теме
3. Для модераторов: создание/удаление раздела, удаление тем, сообщений, создание и блокирование пользователей
4. Для обычных пользователей: создание тем, создание сообщений в теме

# Выполнение работы

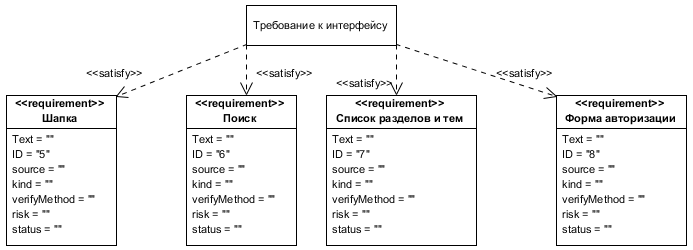
## Функциональные требования к системе



## Системные требования



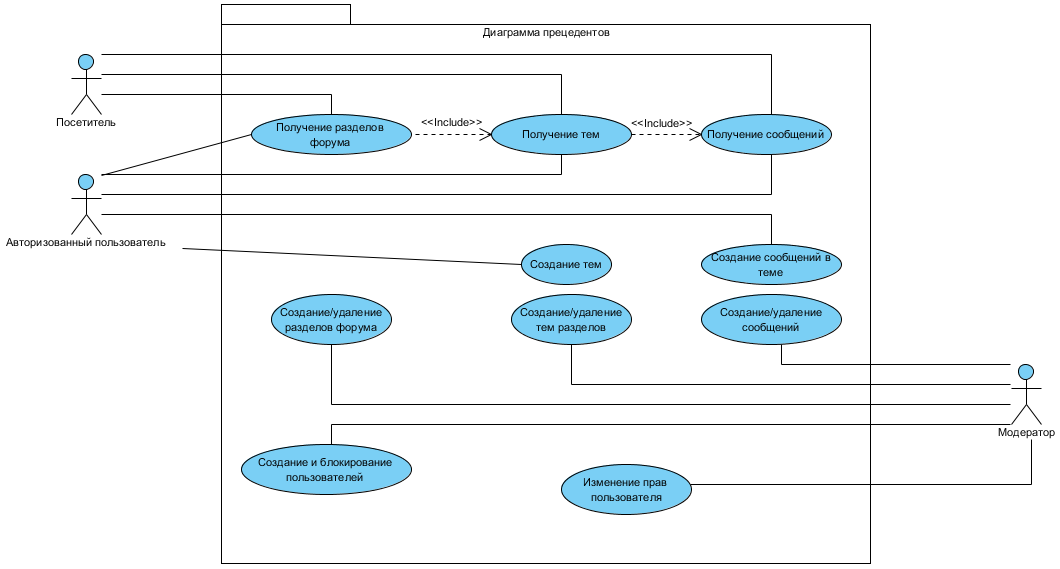
## Требования к интерфейсу



## Требования к информационной безопасности

****

## Диаграмма прецедентов



## Описание сценариев

Получение разделов форума

Раздел «Общее описание сценария»

|  |  |
| --- | --- |
| Прецедент | Получение разделов форума |
| Акторы | Посетитель, авторизованный пользователь |
| Цель | Получить список разделов форума |
| Краткое описание | Посетитель или авторизованный пользователь, используя поисковую строку, получает интересующий его список разделов, представленных на форуме |
| Тип | Базовый прецедент |
| Ссылки на другие прецеденты | – |

Раздел «Типичный ход событий»

|  |  |
| --- | --- |
| Действия акторов | Отклик системы |
| 1. Посетитель или авторизованный пользователь вводит часть строки в соответствующее поле для поиска.   Исключение №1: Отсутствуют разделы, удовлетворяющие условию поиска. | 2. Выполняется поиск данных.  3. Данные отправляются клиенту. |

Раздел «Исключения»

|  |  |
| --- | --- |
| Действия акторов | Отклик системы |
| Исключение №1: Отсутствуют разделы, удовлетворяющие условию поиска. | Сайт отображает сообщение о том, что не найдено разделов, удовлетворяющих условию поиска. |

Получение тем форума

Раздел «Общее описание сценария»

|  |  |
| --- | --- |
| Прецедент | Получение тем форума |
| Акторы | Посетитель, авторизованный пользователь |
| Цель | Получить список тем форума |
| Краткое описание | Посетитель или авторизованный пользователь, используя поисковую строку, получает интересующий его список тем, представленных на форуме |
| Тип | Базовый прецедент |
| Ссылки на другие прецеденты | – |

Раздел «Типичный ход событий»

|  |  |
| --- | --- |
| Действия акторов | Отклик системы |
| 1. Посетитель или авторизованный пользователь вводит часть строки в соответствующее поле для поиска.   Исключение №1: Отсутствуют темы, удовлетворяющие условию поиска. | 2. Выполняется поиск данных.  3. Данные отправляются клиенту. |

Раздел «Исключения»

|  |  |
| --- | --- |
| Действия акторов | Отклик системы |
| Исключение №1: Отсутствуют темы, удовлетворяющие условию поиска. | Сайт отображает сообщение о том, что не найдено тем, удовлетворяющих условию поиска. |

Получение сообщений форума

Раздел «Общее описание сценария»

|  |  |
| --- | --- |
| Прецедент | Получение сообщений форума |
| Акторы | Посетитель, авторизованный пользователь |
| Цель | Получить список сообщений форума |
| Краткое описание | Посетитель или авторизованный пользователь, используя поисковую строку, получает интересующий его список сообщений, представленных на форуме |
| Тип | Базовый прецедент |
| Ссылки на другие прецеденты | – |

Раздел «Типичный ход событий»

|  |  |
| --- | --- |
| Действия акторов | Отклик системы |
| 1. Посетитель или авторизованный пользователь вводит часть строки в соответствующее поле для поиска.   Исключение №1: Отсутствуют сообщения, удовлетворяющие условию поиска. | 2. Выполняется поиск данных.  3. Данные отправляются клиенту. |

Раздел «Исключения»

|  |  |
| --- | --- |
| Действия акторов | Отклик системы |
| Исключение №1: Отсутствуют сообщения, удовлетворяющие условию поиска. | Сайт отображает сообщение о том, что не найдено сообщений, удовлетворяющих условию поиска. |

Создание тем

Раздел «Общее описание сценария»

|  |  |
| --- | --- |
| Прецедент | Создание тем |
| Акторы | Авторизованный пользователь |
| Цель | Добавить новую тему |
| Краткое описание | Авторизованный пользователь, используя соответствующую форму, добавляет новую тему |
| Тип | Базовый прецедент. |
| Ссылки на другие прецеденты | – |

Раздел «Типичный ход событий»

|  |  |
| --- | --- |
| Действия акторов | Отклик системы |
| 1. Авторизованный пользователь заполняет соответствующую форму.   Исключение №1: Такая тема уже существует. | 1. Данные отправляются на сервер. 2. Сайт отображает сообщение о добавлении новой темы. |

Раздел «Исключения»

|  |  |
| --- | --- |
| Действия акторов | Отклик системы |
| Исключение №1: Такая тема уже существует. | Сайт отображает сообщение о том, что тема с таким названием уже существует. |

Создание сообщений в теме

Раздел «Общее описание сценария»

|  |  |
| --- | --- |
| Прецедент | Создание сообщений в теме |
| Акторы | Авторизованный пользователь |
| Цель | Добавить новую сообщение в существующую тему |
| Краткое описание | Авторизованный пользователь, используя соответствующую форму, добавляет новое сообщение в существующей теме |
| Тип | Базовый прецедент. |
| Ссылки на другие прецеденты | – |

Раздел «Типичный ход событий»

|  |  |
| --- | --- |
| Действия акторов | Отклик системы |
| 1. Авторизованный пользователь заполняет соответствующую форму.   Исключение №1: Пользователь не авторизован. | 1. Данные отправляются на сервер. 2. Сайт отображает сообщение о добавлении нового сообщения. |

Раздел «Исключения»

|  |  |
| --- | --- |
| Действия акторов | Отклик системы |
| Исключение №1: Пользователь не авторизован. | Сайт отображает сообщение о том, что пользователь, добавляющий сообщение не авторизован. |

Создание разделов

Раздел «Общее описание сценария»

|  |  |
| --- | --- |
| Прецедент | Создание новых разделов |
| Акторы | Модератор |
| Цель | Добавить новый раздел |
| Краткое описание | Модератор, используя соответствующую форму, добавляет новое сообщение в существующей теме |
| Тип | Базовый прецедент. |
| Ссылки на другие прецеденты | – |

Раздел «Типичный ход событий»

|  |  |
| --- | --- |
| Действия акторов | Отклик системы |
| 1. Модератор заполняет соответствующую форму.   Исключение №1: Такой раздел существует. | 1. Данные отправляются на сервер. 2. Сайт отображает сообщение о добавлении нового раздела. |

Раздел «Исключения»

|  |  |
| --- | --- |
| Действия акторов | Отклик системы |
| Исключение №1: Такой раздел существует. | Сайт отображает сообщение о том, что раздел с таким названием уже существует. |

Удаление сообщения в теме

Раздел «Общее описание сценария»

|  |  |
| --- | --- |
| Прецедент | Удаление сообщения в теме |
| Акторы | Модератор |
| Цель | Удалить сообщение |
| Краткое описание | Модератор, используя соответствующую кнопку, удаляет существующее сообщение |
| Тип | Базовый прецедент. |
| Ссылки на другие прецеденты | – |

Раздел «Типичный ход событий»

|  |  |
| --- | --- |
| Действия акторов | Отклик системы |
| 1. Модератор нажатием кнопки удаляет сообщение.   Исключение №1: Пользователь не является модератором. | 1. Данное сообщение удаляется. 2. Сайт отображает сообщение об удалении сообщения. |

Раздел «Исключения»

|  |  |
| --- | --- |
| Действия акторов | Отклик системы |
| Исключение №1: Пользователь не является модератором. | Сайт отображает сообщение о том, что пользователь, удаляющий сообщение не является модератором. |

Удаление темы

Раздел «Общее описание сценария»

|  |  |
| --- | --- |
| Прецедент | Удаление темы |
| Акторы | Модератор |
| Цель | Удалить тему |
| Краткое описание | Модератор, используя соответствующую кнопку, удаляет существующую тему |
| Тип | Базовый прецедент. |
| Ссылки на другие прецеденты | – |

Раздел «Типичный ход событий»

|  |  |
| --- | --- |
| Действия акторов | Отклик системы |
| 1. Модератор нажатием кнопки удаляет тему.   Исключение №1: Пользователь не является модератором. | 1. Данная тема удаляется. 2. Сайт отображает сообщение об удалении темы. |

Раздел «Исключения»

|  |  |
| --- | --- |
| Действия акторов | Отклик системы |
| Исключение №1: Пользователь не является модератором. | Сайт отображает сообщение о том, что пользователь, удаляющий сообщение не является модератором. |

Удаление раздела

Раздел «Общее описание сценария»

|  |  |
| --- | --- |
| Прецедент | Удаление раздела |
| Акторы | Модератор |
| Цель | Удалить раздел |
| Краткое описание | Модератор, используя соответствующую кнопку, удаляет существующий раздел |
| Тип | Базовый прецедент. |
| Ссылки на другие прецеденты | – |

Раздел «Типичный ход событий»

|  |  |
| --- | --- |
| Действия акторов | Отклик системы |
| 1. Модератор нажатием кнопки удаляет раздел.   Исключение №1: Пользователь не является модератором. | 1. Данный раздел удаляется. 2. Сайт отображает сообщение об удалении раздела. |

Раздел «Исключения»

|  |  |
| --- | --- |
| Действия акторов | Отклик системы |
| Исключение №1: Пользователь не является модератором. | Сайт отображает сообщение о том, что пользователь, удаляющий сообщение не является модератором. |

Добавление нового пользователя

Раздел «Общее описание сценария»

|  |  |
| --- | --- |
| Прецедент | Добавление нового пользователя |
| Акторы | Модератор |
| Цель | Добавить нового пользователя |
| Краткое описание | Модератор, используя соответствующую таблицу с заявками на авторизацию, добавляет нового пользователя |
| Тип | Базовый прецедент. |
| Ссылки на другие прецеденты | – |

Раздел «Типичный ход событий»

|  |  |
| --- | --- |
| Действия акторов | Отклик системы |
| 1. Модератор проверяет правильность заполнение заявки и добавляет нового пользователя.   Исключение №1: Не все поля заполнены | 1. Новый пользователь добавляется. 2. Сайт отображает сообщение о добавлении нового пользователя. |

Раздел «Исключения»

|  |  |
| --- | --- |
| Действия акторов | Отклик системы |
| Исключение №1: Не все поля заполнены. | Сайт отображает сообщение о том, что не все обязательные поля заполнены. |

Блокирование пользователя

Раздел «Общее описание сценария»

|  |  |
| --- | --- |
| Прецедент | Блокирование пользователя |
| Акторы | Модератор |
| Цель | Заблокировать существующего пользователя |
| Краткое описание | Модератор, используя специальную форму, блокирует пользователя |
| Тип | Базовый прецедент. |
| Ссылки на другие прецеденты | – |

Раздел «Типичный ход событий»

|  |  |
| --- | --- |
| Действия акторов | Отклик системы |
| 1. Модератор, нажатием на соответствующую кнопку, блокирует пользователя.   Исключение №1: Пользователь уже заблокирован. | 1. Пользователь блокируется. 2. Сайт отображает сообщение о блокировке пользователя. |

Раздел «Исключения»

|  |  |
| --- | --- |
| Действия акторов | Отклик системы |
| Исключение №1: Пользователь уже заблокирован. | Сайт отображает сообщение о том, что данный пользователь уже заблокирован. |

Изменение прав пользователя

Раздел «Общее описание сценария»

|  |  |
| --- | --- |
| Прецедент | Изменение прав пользователя |
| Акторы | Модератор |
| Цель | Изменить права авторизованного пользователя |
| Краткое описание | Модератор, используя специальную форму меняет права авторизованного пользователя. |
| Тип | Базовый прецедент. |
| Ссылки на другие прецеденты | – |

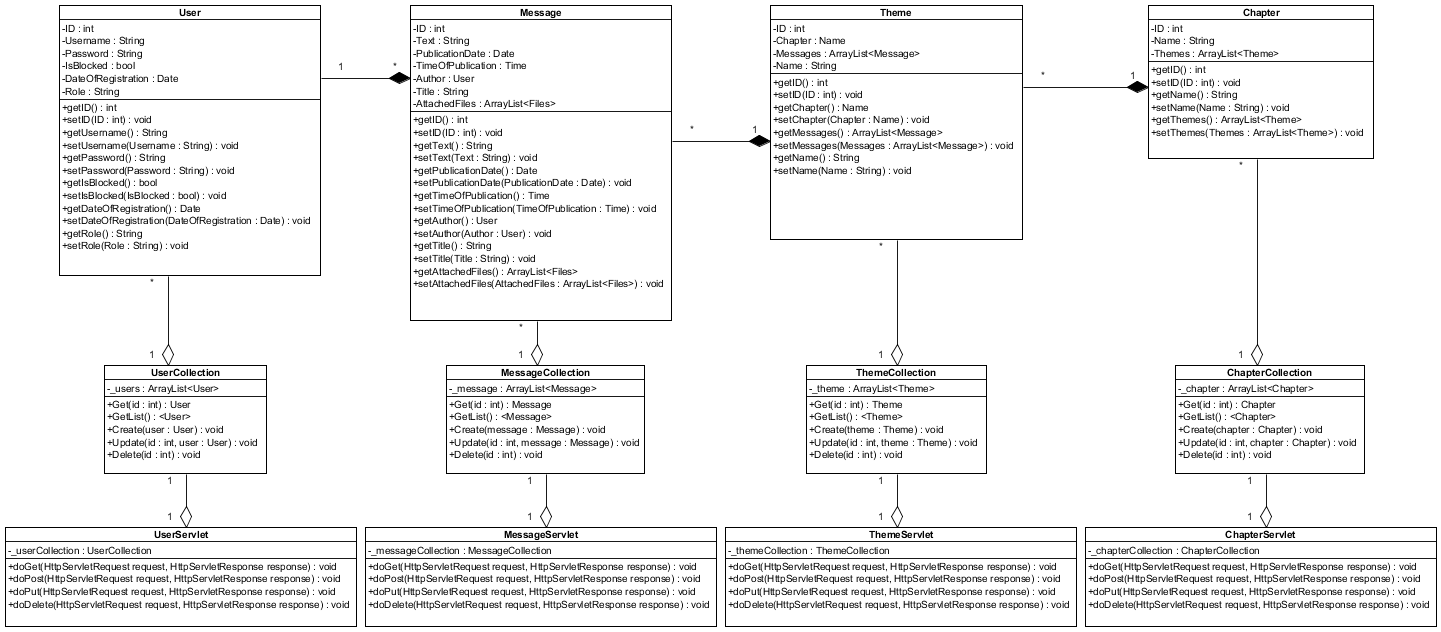
Раздел «Типичный ход событий»

|  |  |
| --- | --- |
| Действия акторов | Отклик системы |
| 1. Модератор меняет значение параметра «Права пользователя».   Исключение №1: Пользователь заблокирован. | 1. Права пользователя меняются. 2. Сайт отображает сообщение об изменении прав пользователя. |

Раздел «Исключения»

|  |  |
| --- | --- |
| Действия акторов | Отклик системы |
| Исключение №1: Пользователь заблокирован. | 1. Сайт отображает сообщение о том, что данный пользователь заблокирован. |

## Диаграмма классов



## Описание классов

|  |  |  |
| --- | --- | --- |
| **User** | **Пользователь** | |
| **Атрибут** | **Тип** | **Описание** |
| Id | int | Идентификатор |
| Username | String | Имя пользователя |
| Password | String | Пароль пользователя |
| IsBlocked | bool | Наличие блокировки |
| DateOfRegistretion | Date | Дата регистрации |
| Role | String | Роль на сайте |
| **Метод** | **Возвращаемое значение** | **Описание** |
| get\_Id () | Guid | Получить идентификатор |
| set\_Id () | void | Установить идентификатор |
| get\_Username () | String | Получить имя пользователя |
| set\_Username () | void | Установить имя пользователя |
| get\_Password () | String | Получить пароль пользователя |
| set\_Password () | void | Установить пароль пользователя |
| get\_IsBlocked () | bool | Получить наличие блокировки пользователя |
| set\_IsBlocked () | void | Установить блокировку пользователя |
| get\_DateOfRegistretion () | Date | Получить дату регистрации пользователя |
| set\_DateOfRegistretion () | void | Установить дату регистрации пользователя |
| get\_Role () | String | Получить роль пользователя |
| set\_Role () | void | Установить роль пользователя |

|  |  |  |
| --- | --- | --- |
| **Message** | **Сообщение** | |
| **Атрибут** | **Тип** | **Описание** |
| Id | int | Идентификатор |
| Text | String | Текст сообщения |
| PublicationDate | Date | Дата публикации |
| TimeOfPublication | Time | Время публикации |
| Author | User | Автор сообщения |
| Title | String | Заголовок сообщения |
| AttachedFiles | ArrayList<Files> | Прикрепленные файлы |
| **Метод** | **Возвращаемое значение** | **Описание** |
| get\_Id () | Guid | Получить идентификатор |
| set\_Id () | void | Установить идентификатор |
| get\_Text () | String | Получить текст сообщения |
| set\_Text () | void | Установить текст сообщения |
| get\_PublicationDate () | Date | Получить дату публикации |
| set\_PublicationDate () | void | Установить дату публикации |
| get\_TimeOfPublication () | Time | Получить время публикации |
| set\_TimeOfPublication () | void | Установить время публикации |
| get\_Author () | String | Получить автора сообщения |
| set\_Author () | void | Установить автора сообщения |
| get\_Title () | String | Получить заголовок сообщения |
| set\_Title () | void | Установить заголовок сообщения |
| get\_AttachedFiles () | ArrayList<Files> | Получить прикрепленные файлы |
| set\_AttachedFiles () | void | Установить прикрепленные файлы |

|  |  |  |
| --- | --- | --- |
| **Theme** | **Тема** | |
| **Атрибут** | **Тип** | **Описание** |
| Id | Int | Идентификатор |
| Chapter | Chapter | Раздел форума |
| Messages | ArrayList<Message> | Список сообщений в теме |
| Name | String | Название темы |
| **Метод** | **Возвращаемое значение** | **Описание** |
| get\_Id() | Guid | Получить идентификатор |
| set\_Id() | void | Установить идентификатор |
| get\_Chapter() | String | Получить раздел форума |
| set\_Chapter() | void | Установить раздел форума |
| get\_Message() | ArrayList<Message> | Получить сообщения в теме |
| set\_Message() | void | Установить сообщения в теме |
| get\_Name() | String | Получить название темы |
| set\_Name() | void | Установить название темы |

|  |  |  |
| --- | --- | --- |
| **Chapter** | **Раздел** | |
| **Атрибут** | **Тип** | **Описание** |
| Id | Int | Идентификатор |
| Name | String | Название темы |
| Themes | ArrayList<Theme> | Список тем в разделе |
| **Метод** | **Возвращаемое значение** | **Описание** |
| get\_Id() | Guid | Получить идентификатор |
| set\_Id() | void | Установить идентификатор |
| get\_Name() | String | Получить название темы |
| set\_Name() | void | Установить название темы |
| get\_Theme() | ArrayList<Theme> | Получить темы в разделе |
| set\_Theme() | void | Установить темы в разделе |

|  |  |  |
| --- | --- | --- |
| **UserCollection** | **Коллекция пользователей** | |
| **Атрибут** | **Тип** | **Описание** |
| \_users | ArrayList<User> | Список пользователей |
| **Метод** | **Возвращаемое значение** | **Описание** |
| Get(id : int) | User | Получить пользователя |
| GetList(id : int) | ArrayList<User> | Получить список пользователей |
| Create(user : User) | void | Создать пользователя |
| Update(id : int, user : User) | void | Редактировать пользователя |
| Delete(id : int) | void | Удалить пользователя |
| UserCollection (user : ArrayList<User>) |  | Конструктор |

|  |  |  |
| --- | --- | --- |
| **MessageCollection** | **Коллекция сообщений** | |
| **Атрибут** | **Тип** | **Описание** |
| \_message | ArrayList<Message> | Список сообщений |
| **Метод** | **Возвращаемое значение** | **Описание** |
| Get(id : int) | Message | Получить сообщение |
| GetList(id : int) | ArrayList<Message> | Получить список сообщений |
| Create(message : Message) | void | Создать сообщение |
| Update(id : int, message : Message) | void | Редактировать сообщение |
| Delete(id : int) | void | Удалить сообщение |
| MessageCollection (message : ArrayList<Message>) |  | Конструктор |

|  |  |  |
| --- | --- | --- |
| **ThemeCollection** | **Коллекция тем** | |
| **Атрибут** | **Тип** | **Описание** |
| \_theme | ArrayList<Theme> | Список тем |
| **Метод** | **Возвращаемое значение** | **Описание** |
| Get(id : int) | Theme | Получить тему |
| GetList(id : int) | ArrayList<Theme> | Получить список тем |
| Theme | void | Создать тему |
| Update(id : int, theme : Theme) | void | Редактировать тему |
| Delete(id : int) | void | Удалить тему |
| ThemeCollection (theme : ArrayList<Theme>) |  | Конструктор |

|  |  |  |
| --- | --- | --- |
| **ChapterCollection** | **Коллекция разделов** | |
| **Атрибут** | **Тип** | **Описание** |
| \_chapter | ArrayList<Chapter> | Список разделов |
| **Метод** | **Возвращаемое значение** | **Описание** |
| Get(id : int) | Chapter | Получить раздел |
| GetList(id : int) | ArrayList<Chapter> | Получить список разделов |
| Create(chapter : Chapter) | void | Создать раздел |
| Update(id : int, chapter : Chapter) | void | Редактировать раздел |
| Delete(id : int) | void | Удалить раздел |
| ChapterCollection (chapter : ArrayList<Chapter>) |  | Конструктор |

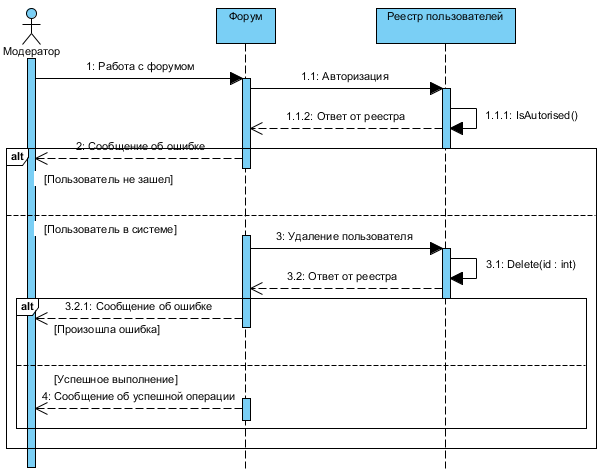
|  |  |  |
| --- | --- | --- |
| **UserServlet** | **Servlet пользователей** | |
| **Атрибут** | **Тип** | **Описание** |
| \_userCollection | UserCollection | Список пользователей |
| **Метод** | **Возвращаемое значение** | **Описание** |
| doGet(HttpServletRequest request, HttpServletResponse response) | void | Get запрос |
| doPost(HttpServletRequest request, HttpServletResponse response) | void | Post запрос |
| doPut(HttpServletRequest request, HttpServletResponse response) | void | Put запрос |
| doDelete(HttpServletRequest request, HttpServletResponse response) | void | Delete запрос |
| UserServlet(userCollection : UserCollection) |  | Конструктор |

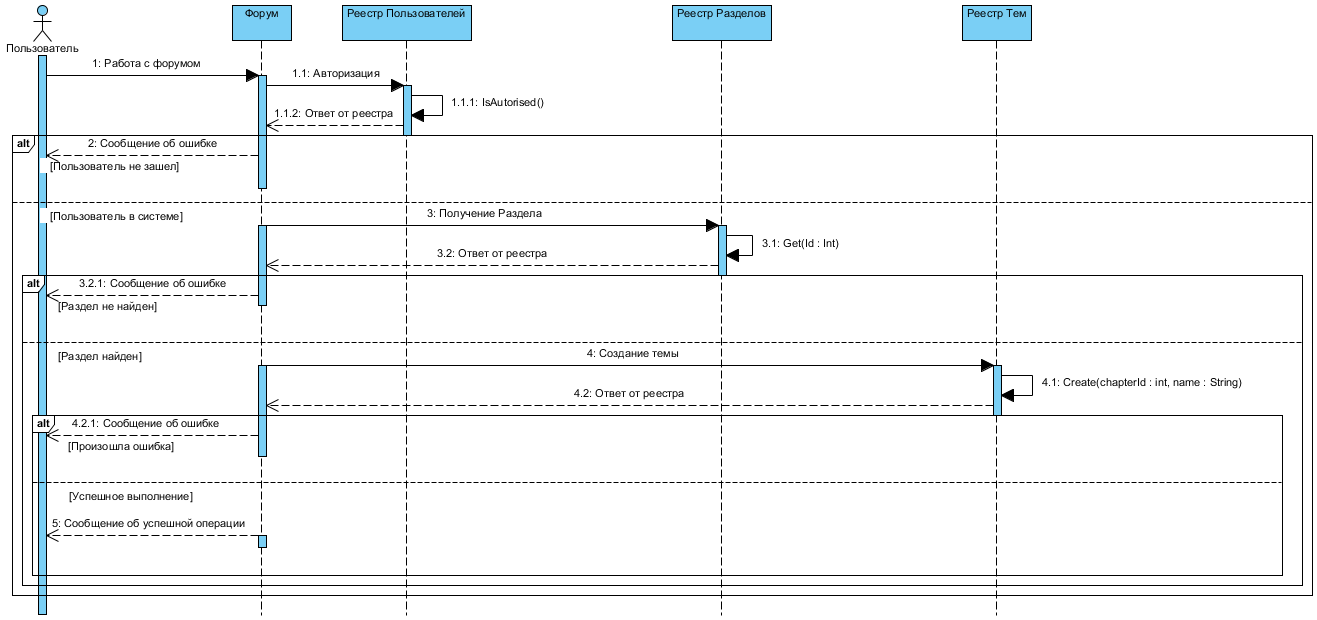
|  |  |  |
| --- | --- | --- |
| **MessageServlet** | **Servlet сообщений** | |
| **Атрибут** | **Тип** | **Описание** |
| \_messageCollection | MessageCollection | Список сообщений |
| **Метод** | **Возвращаемое значение** | **Описание** |
| doGet(HttpServletRequest request, HttpServletResponse response) | void | Get запрос |
| doPost(HttpServletRequest request, HttpServletResponse response) | void | Post запрос |
| doPut(HttpServletRequest request, HttpServletResponse response) | void | Put запрос |
| doDelete(HttpServletRequest request, HttpServletResponse response) | void | Delete запрос |
| MessageServlet(messageCollection : MessageCollection) |  | Конструктор |

|  |  |  |
| --- | --- | --- |
| **ThemeServlet** | **Servlet тем** | |
| **Атрибут** | **Тип** | **Описание** |
| \_themeCollection | ThemeCollection | Список тем |
| **Метод** | **Возвращаемое значение** | **Описание** |
| doGet(HttpServletRequest request, HttpServletResponse response) | void | Get запрос |
| doPost(HttpServletRequest request, HttpServletResponse response) | void | Post запрос |
| doPut(HttpServletRequest request, HttpServletResponse response) | void | Put запрос |
| doDelete(HttpServletRequest request, HttpServletResponse response) | void | Delete запрос |
| ThemeServlet(themeCollection : ThemeCollection) |  | Конструктор |

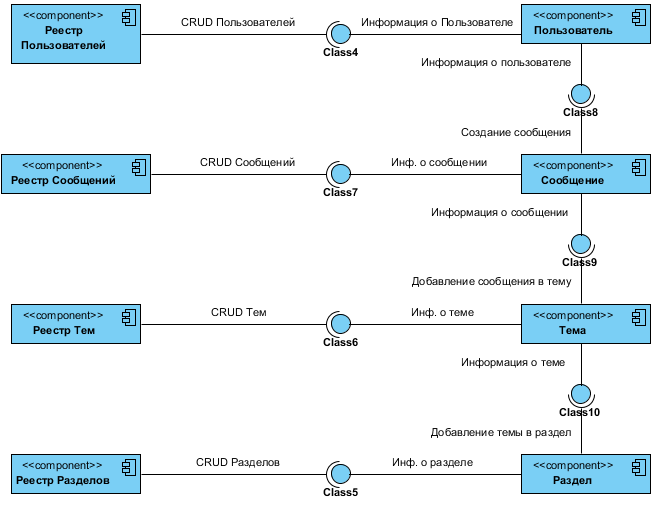
|  |  |  |
| --- | --- | --- |
| **ChapterServlet** | **Servlet разделов** | |
| **Атрибут** | **Тип** | **Описание** |
| \_chapterCollection | ChapterCollection | Список разделов |
| **Метод** | **Возвращаемое значение** | **Описание** |
| doGet(HttpServletRequest request, HttpServletResponse response) | void | Get запрос |
| doPost(HttpServletRequest request, HttpServletResponse response) | void | Post запрос |
| doPut(HttpServletRequest request, HttpServletResponse response) | void | Put запрос |
| doDelete(HttpServletRequest request, HttpServletResponse response) | void | Delete запрос |
| ChapterServlet(chapterCollection : ChapterCollection) |  | Конструктор |

## Диаграммы последовательности

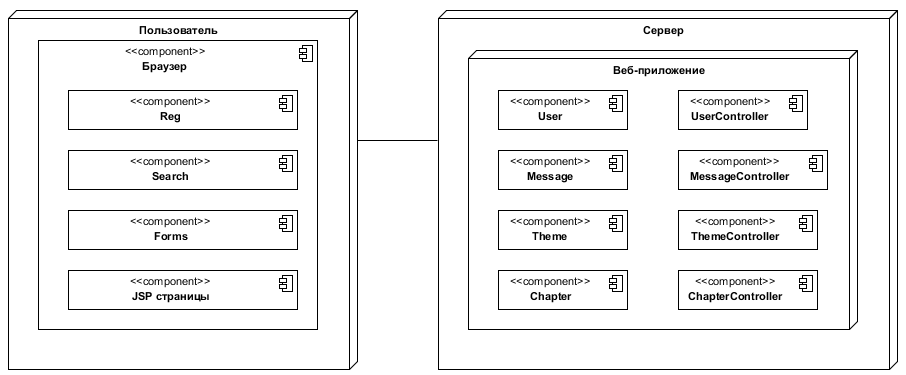




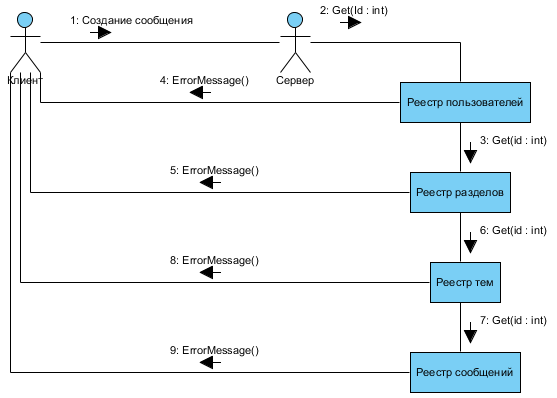
## Диаграмма компонентов



## Диаграмма развертывания



## Диаграмма коммуникаций



## Листинг

**Java:**

public class User {

private int \_iD;

private String \_username;

private String \_password;

private boolean \_isBlocked;

private Date \_dateOfRegistration;

private String \_role;

private UserCollection \_unnamed\_UserCollection\_;

public int getID() {

throw new UnsupportedOperationException();

}

public void setID(int aID) {

throw new UnsupportedOperationException();

}

public String getUsername() {

throw new UnsupportedOperationException();

}

public void setUsername(String aUsername) {

throw new UnsupportedOperationException();

}

public String getPassword() {

throw new UnsupportedOperationException();

}

public void setPassword(String aPassword) {

throw new UnsupportedOperationException();

}

public boolean getIsBlocked() {

throw new UnsupportedOperationException();

}

public void setIsBlocked(boolean aIsBlocked) {

throw new UnsupportedOperationException();

}

public Date getDateOfRegistration() {

throw new UnsupportedOperationException();

}

public void setDateOfRegistration(Date aDateOfRegistration) {

throw new UnsupportedOperationException();

}

public String getRole() {

throw new UnsupportedOperationException();

}

public void setRole(String aRole) {

throw new UnsupportedOperationException();

}

}

public class UserCollection {

private ArrayList<User> \_users;

private UserServlet \_unnamed\_UserServlet\_;

public User Get(int aId) {

throw new UnsupportedOperationException();

}

public \_User> GetList() {

throw new UnsupportedOperationException();

}

public void Create(User aUser) {

throw new UnsupportedOperationException();

}

public void Update(int aId, User aUser) {

throw new UnsupportedOperationException();

}

public void Delete(int aId) {

throw new UnsupportedOperationException();

}

public UserCollection(ArrayList<User> aUser) {

throw new UnsupportedOperationException();

}

}

public class UserServlet {

private UserCollection \_userCollection;

public UserCollection \_unnamed\_UserCollection\_;

public void doGet(Object aHttpServletRequest\_request, Object aHttpServletResponse\_response) {

throw new UnsupportedOperationException();

}

public void doPost(Object aHttpServletRequest\_request, Object aHttpServletResponse\_response) {

throw new UnsupportedOperationException();

}

public void doPut(Object aHttpServletRequest\_request, Object aHttpServletResponse\_response) {

throw new UnsupportedOperationException();

}

public void doDelete(Object aHttpServletRequest\_request, Object aHttpServletResponse\_response) {

throw new UnsupportedOperationException();

}

public ChapterServlet(ChapterCollection aChapterCollection) {

throw new UnsupportedOperationException();

}

}

public class Chapter {

private int \_iD;

private String \_name;

private ArrayList<Theme> \_themes;

private ChapterCollection \_unnamed\_ChapterCollection\_;

public int getID() {

throw new UnsupportedOperationException();

}

public void setID(int aID) {

throw new UnsupportedOperationException();

}

public String getName() {

throw new UnsupportedOperationException();

}

public void setName(String aName) {

throw new UnsupportedOperationException();

}

public ArrayList<Theme> getThemes() {

throw new UnsupportedOperationException();

}

public void setThemes(ArrayList<Theme> aThemes) {

throw new UnsupportedOperationException();

}

}

public class ChapterCollection {

private ArrayList<Chapter> \_chapter;

private ChapterServlet \_unnamed\_ChapterServlet\_;

public Chapter Get(int aId) {

throw new UnsupportedOperationException();

}

public \_Chapter> GetList() {

throw new UnsupportedOperationException();

}

public void Create(Chapter aChapter) {

throw new UnsupportedOperationException();

}

public void Update(int aId, Chapter aChapter) {

throw new UnsupportedOperationException();

}

public void Delete(int aId) {

throw new UnsupportedOperationException();

}

public ChapterCollection(ArrayList<Chapter> aUser) {

throw new UnsupportedOperationException();

}}

public class ChapterServlet {

private ChapterCollection \_chapterCollection;

public ChapterCollection \_unnamed\_ChapterCollection\_;

public void doGet(Object aHttpServletRequest\_request, Object aHttpServletResponse\_response) {

throw new UnsupportedOperationException();

}

public void doPost(Object aHttpServletRequest\_request, Object aHttpServletResponse\_response) {

throw new UnsupportedOperationException();

}

public void doPut(Object aHttpServletRequest\_request, Object aHttpServletResponse\_response) {

throw new UnsupportedOperationException();

}

public void doDelete(Object aHttpServletRequest\_request, Object aHttpServletResponse\_response) {

throw new UnsupportedOperationException();

}

public ChapterServlet(ChapterCollection aChapterCollection) {

throw new UnsupportedOperationException();

}

}

public class Theme {

private int \_iD;

private Name \_chapter;

private ArrayList<Message> \_messages;

private String \_name;

public Chapter \_unnamed\_Chapter\_;

private ThemeCollection \_unnamed\_ThemeCollection\_;

public int getID() {

throw new UnsupportedOperationException();

}

public void setID(int aID) {

throw new UnsupportedOperationException();

}

public Name getChapter() {

throw new UnsupportedOperationException();

}

public void setChapter(Name aChapter) {

throw new UnsupportedOperationException();

}

public ArrayList<Message> getMessages() {

throw new UnsupportedOperationException();

}

public void setMessages(ArrayList<Message> aMessages) {

throw new UnsupportedOperationException();

}

public String getName() {

throw new UnsupportedOperationException();

}

public void setName(String aName) {

throw new UnsupportedOperationException();

}

}

public class ThemeCollection {

private ArrayList<Theme> \_theme;

private ThemeServlet \_unnamed\_ThemeServlet\_;

public Theme Get(int aId) {

throw new UnsupportedOperationException();

}

public \_Theme> GetList() {

throw new UnsupportedOperationException();

}

public void Create(Theme aTheme) {

throw new UnsupportedOperationException();

}

public void Update(int aId, Theme aTheme) {

throw new UnsupportedOperationException();

}

public void Delete(int aId) {

throw new UnsupportedOperationException();

}

public ThemeCollection(ArrayList<Theme> aUser) {

throw new UnsupportedOperationException();

}

}

public class ThemeServlet {

private ThemeCollection \_themeCollection;

public ThemeCollection \_unnamed\_ThemeCollection\_;

public void doGet(Object aHttpServletRequest\_request, Object aHttpServletResponse\_response) {

throw new UnsupportedOperationException();

}

public void doPost(Object aHttpServletRequest\_request, Object aHttpServletResponse\_response) {

throw new UnsupportedOperationException();

}

public void doPut(Object aHttpServletRequest\_request, Object aHttpServletResponse\_response) {

throw new UnsupportedOperationException();

}

public void doDelete(Object aHttpServletRequest\_request, Object aHttpServletResponse\_response) {

throw new UnsupportedOperationException();

}

public ThemeServlet(ThemeCollection aThemeCollection) {

throw new UnsupportedOperationException();

}

}

public class Message {

private int \_iD;

private String \_text;

private Date \_publicationDate;

private Time \_timeOfPublication;

private User \_author;

private String \_title;

private ArrayList<Files> \_attachedFiles;

public Theme \_unnamed\_Theme\_;

private MessageCollection \_unnamed\_MessageCollection\_;

public User \_unnamed\_User\_;

public int getID() {

throw new UnsupportedOperationException();

}

public void setID(int aID) {

throw new UnsupportedOperationException();

}

public String getText() {

throw new UnsupportedOperationException();

}

public void setText(String aText) {

throw new UnsupportedOperationException();

}

public Date getPublicationDate() {

throw new UnsupportedOperationException();

}

public void setPublicationDate(Date aPublicationDate) {

throw new UnsupportedOperationException();

}

public Time getTimeOfPublication() {

throw new UnsupportedOperationException();

}

public void setTimeOfPublication(Time aTimeOfPublication) {

throw new UnsupportedOperationException();

}

public User getAuthor() {

throw new UnsupportedOperationException();

}

public void setAuthor(User aAuthor) {

throw new UnsupportedOperationException();

}

public String getTitle() {

throw new UnsupportedOperationException();

}

public void setTitle(String aTitle) {

throw new UnsupportedOperationException();

}

public ArrayList<Files> getAttachedFiles() {

throw new UnsupportedOperationException();

}

public void setAttachedFiles(ArrayList<Files> aAttachedFiles) {

throw new UnsupportedOperationException();

}

}

public class MessageCollection {

private ArrayList<Message> \_message;

private MessageServlet \_unnamed\_MessageServlet\_;

public Message Get(int aId) {

throw new UnsupportedOperationException();

}

public \_Message> GetList() {

throw new UnsupportedOperationException();

}

public void Create(Message aMessage) {

throw new UnsupportedOperationException();

}

public void Update(int aId, Message aMessage) {

throw new UnsupportedOperationException();

}

public void Delete(int aId) {

throw new UnsupportedOperationException();

}

public MessageCollection(ArrayList<Message> aUser) {

throw new UnsupportedOperationException();

}

}

public class MessageServlet {

private MessageCollection \_messageCollection;

public MessageCollection \_unnamed\_MessageCollection\_;

public void doGet(Object aHttpServletRequest\_request, Object aHttpServletResponse\_response) {

throw new UnsupportedOperationException();

}

public void doPost(Object aHttpServletRequest\_request, Object aHttpServletResponse\_response) {

throw new UnsupportedOperationException();

}

public void doPut(Object aHttpServletRequest\_request, Object aHttpServletResponse\_response) {

throw new UnsupportedOperationException();

}

public void doDelete(Object aHttpServletRequest\_request, Object aHttpServletResponse\_response) {

throw new UnsupportedOperationException();

}

public MessageServlet(MessageCollection aMessageCollection) {

throw new UnsupportedOperationException();

}

}

**C#**

public class User {

private int iD;

private String username;

private String password;

private bool isBlocked;

private Date dateOfRegistration;

private String role;

public int GetID() {

throw new System.Exception("Not implemented");

}

public void SetID(int iD) {

throw new System.Exception("Not implemented");

}

public String GetUsername() {

throw new System.Exception("Not implemented");

}

public void SetUsername(String username) {

throw new System.Exception("Not implemented");

}

public String GetPassword() {

throw new System.Exception("Not implemented");

}

public void SetPassword(String password) {

throw new System.Exception("Not implemented");

}

public bool GetIsBlocked() {

throw new System.Exception("Not implemented");

}

public void SetIsBlocked(bool isBlocked) {

throw new System.Exception("Not implemented");

}

public Date GetDateOfRegistration() {

throw new System.Exception("Not implemented");

}

public void SetDateOfRegistration(Date dateOfRegistration) {

throw new System.Exception("Not implemented");

}

public String GetRole() {

throw new System.Exception("Not implemented");

}

public void SetRole(String role) {

throw new System.Exception("Not implemented");

}

}

public class UserCollection {

private ArrayList<User> \_users;

public User Get(int id) {

throw new System.Exception("Not implemented");

}

public <User> GetList() {

throw new System.Exception("Not implemented");

}

public void Create(User user) {

throw new System.Exception("Not implemented");

}

public void Update(int id, User user) {

throw new System.Exception("Not implemented");

}

public void Delete(int id) {

throw new System.Exception("Not implemented");

}

public UserCollection(ArrayList<User> user) {

throw new System.Exception("Not implemented");

}

}

public class UserServlet {

private UserCollection \_userCollection;

public void DoGet(object httpServletRequest\_request, object httpServletResponse\_response) {

throw new System.Exception("Not implemented");

}

public void DoPost(object httpServletRequest\_request, object httpServletResponse\_response) {

throw new System.Exception("Not implemented");

}

public void DoPut(object httpServletRequest\_request, object httpServletResponse\_response) {

throw new System.Exception("Not implemented");

}

public void DoDelete(object httpServletRequest\_request, object httpServletResponse\_response){

throw new System.Exception("Not implemented");

}

public UserServlet(UserCollection userCollection) {

throw new System.Exception("Not implemented");

}

}

public class Message {

private int iD;

private String text;

private Date publicationDate;

private Time timeOfPublication;

private User author;

private String title;

private ArrayList<Files> attachedFiles;

public int GetID() {

throw new System.Exception("Not implemented");

}

public void SetID(int iD) {

throw new System.Exception("Not implemented");

}

public String GetText() {

throw new System.Exception("Not implemented");

}

public void SetText(String text) {

throw new System.Exception("Not implemented");

}

public Date GetPublicationDate() {

throw new System.Exception("Not implemented");

}

public void SetPublicationDate(Date publicationDate) {

throw new System.Exception("Not implemented");

}

public Time GetTimeOfPublication() {

throw new System.Exception("Not implemented");

}

public void SetTimeOfPublication(Time timeOfPublication) {

throw new System.Exception("Not implemented");

}

public User GetAuthor() {

throw new System.Exception("Not implemented");

}

public void SetAuthor(User author) {

throw new System.Exception("Not implemented");

}

public String GetTitle() {

throw new System.Exception("Not implemented");

}

public void SetTitle(String title) {

throw new System.Exception("Not implemented");

}

public void<Files> GetAttachedFiles() {

throw new System.Exception("Not implemented");

}

public void SetAttachedFiles(object<Files> attachedFiles) {

throw new System.Exception("Not implemented");

}

}

public class MessageCollection {

private ArrayList<Message> \_message;

public Message Get(int id) {

throw new System.Exception("Not implemented");

}

public <Message> GetList() {

throw new System.Exception("Not implemented");

}

public void Create(Message message) {

throw new System.Exception("Not implemented");

}

public void Update(int id, Message message) {

throw new System.Exception("Not implemented");

}

public void Delete(int id) {

throw new System.Exception("Not implemented");

}

public MessageCollection(ArrayList<Message> user) {

throw new System.Exception("Not implemented");

}

}

public class MessageServlet {

private MessageCollection \_messageCollection;

public void DoGet(object httpServletRequest\_request, object httpServletResponse\_response) {

throw new System.Exception("Not implemented");

}

public void DoPost(object httpServletRequest\_request, object httpServletResponse\_response) {

throw new System.Exception("Not implemented");

}

public void DoPut(object httpServletRequest\_request, object httpServletResponse\_response) {

throw new System.Exception("Not implemented");

}

public void DoDelete(object httpServletRequest\_request, object httpServletResponse\_response){

throw new System.Exception("Not implemented");

}

public MessageServlet(MessageCollection messageCollection) {

throw new System.Exception("Not implemented");

}

}

public class Theme {

private int iD;

private Chapter.Name chapter;

private ArrayList<Message> messages;

private String name;

public int GetID() {

throw new System.Exception("Not implemented");

}

public void SetID(int iD) {

throw new System.Exception("Not implemented");

}

public Chapter.Name GetChapter() {

throw new System.Exception("Not implemented");

}

public void SetChapter(Chapter.Name chapter) {

throw new System.Exception("Not implemented");

}

public void<Message> GetMessages() {

throw new System.Exception("Not implemented");

}

public void SetMessages(object<Message> messages) {

throw new System.Exception("Not implemented");

}

public String GetName() {

throw new System.Exception("Not implemented");

}

public void SetName(String name) {

throw new System.Exception("Not implemented");

}

}

public class ThemeCollection {

private ArrayList<Theme> \_theme;

public Theme Get(int id) {

throw new System.Exception("Not implemented");

}

public <Theme> GetList() {

throw new System.Exception("Not implemented");

}

public void Create(Theme theme) {

throw new System.Exception("Not implemented");

}

public void Update(int id, Theme theme) {

throw new System.Exception("Not implemented");

}

public void Delete(int id) {

throw new System.Exception("Not implemented");

}

public ThemeCollection(ArrayList<Theme> user) {

throw new System.Exception("Not implemented");

}

}

public class ThemeServlet {

private ThemeCollection \_themeCollection;

public void DoGet(object httpServletRequest\_request, object httpServletResponse\_response) {

throw new System.Exception("Not implemented");

}

public void DoPost(object httpServletRequest\_request, object httpServletResponse\_response) {

throw new System.Exception("Not implemented");

}

public void DoPut(object httpServletRequest\_request, object httpServletResponse\_response) {

throw new System.Exception("Not implemented");

}

public void DoDelete(object httpServletRequest\_request, object httpServletResponse\_response){

throw new System.Exception("Not implemented");

}

public ThemeServlet(ThemeCollection themeCollection) {

throw new System.Exception("Not implemented");

}

}

public class Chapter {

private int iD;

private String name;

private ArrayList<Theme> themes;

public int GetID() {

throw new System.Exception("Not implemented");

}

public void SetID(int iD) {

throw new System.Exception("Not implemented");

}

public String GetName() {

throw new System.Exception("Not implemented");

}

public void SetName(String name) {

throw new System.Exception("Not implemented");

}

public void<Theme> GetThemes() {

throw new System.Exception("Not implemented");

}

public void SetThemes(object<Theme> themes) {

throw new System.Exception("Not implemented");

}

}

public class ChapterCollection {

private ArrayList<Chapter> \_chapter;

public Chapter Get(int id) {

throw new System.Exception("Not implemented");

}

public <Chapter> GetList() {

throw new System.Exception("Not implemented");

}

public void Create(Chapter chapter) {

throw new System.Exception("Not implemented");

}

public void Update(int id, Chapter chapter) {

throw new System.Exception("Not implemented");

}

public void Delete(int id) {

throw new System.Exception("Not implemented");

}

public ChapterCollection(ArrayList<Chapter> user) {

throw new System.Exception("Not implemented");

}

}

public class ChapterServlet {

private ChapterCollection \_chapterCollection;

public void DoGet(object httpServletRequest\_request, object httpServletResponse\_response) {

throw new System.Exception("Not implemented");

}

public void DoPost(object httpServletRequest\_request, object httpServletResponse\_response) {

throw new System.Exception("Not implemented");

}

public void DoPut(object httpServletRequest\_request, object httpServletResponse\_response) {

throw new System.Exception("Not implemented");

}

public void DoDelete(object httpServletRequest\_request, object httpServletResponse\_response){

throw new System.Exception("Not implemented");

}

public ChapterServlet(ChapterCollection chapterCollection) {

throw new System.Exception("Not implemented");

}

}