Солдатов Артём, ИСТ-111

**Этап 3. Структура бизнес-логики и веб-интерфейса системы.**

1. **Структура распределенной системы.**

Структура распределенной программной системы «Кинотеатр» представлена на рисунке 1.



Рисунок 1. Структура РПС «Кинотеатр»

Распределенная программная система кинотеатр состоит из четырёх слоев:

1. Java Server Faces (View Layer) – фреймворк для разработки веб-приложений, представляющий набор стандартных графических компонентов для создания интерфейса. Этот слой управляет запросами пользователя, а также отвечает за корректное отображение данных на веб-страницах. В приложении этот слой будет представлен CDI-бинами, и xhtml-представлениями.
2. Enterprise Java Beans (Business Layer) – спецификация технологии написания и поддержки серверных компонентов, содержащих бизнес-логику. Этот слой является бизнес-логикой приложения. В приложении этот слой будет представлен сессионными компонентами.
3. Java Persistence API (Data Access Layer) – технология, обеспечивающая объектно-реляционное отображение простых JAVA объектов, и предоставляющая API для управления этими объектами. В приложении этот слой представлен классами-сущностями.
4. MySQL – реляционная база данных MySQL.

**2. Слой пользовательского интерфейса.**

Схема веб-сайта представлена на рисунке 2.



Рисунок 2. Схема веб-сайта.

Каждая страница веб-сайта условно состоит из четырех частей:

1) Header – верхняя часть, которая представлена панелью навигации по основным разделам веб-сайта.

2) LoginMenu – панель авторизации в системе. Состоит из двух полей: логин, пароль. Если пользователь авторизован, то эта панель меняется на приветствие, ссылку на Logout, а также ссылку на страницу информации о бонусном счёте.

3) Content – центральная часть, на которой находится содержание веб-страницы.

4) Footer – copyright на права данного веб-сайта.

Панель навигации по основным разделам веб-сайта состоит из 5 частей:

1) Расписание кинотеатра – schedule.xhtml.

2) Вакансии кинотеатра - vacations.xhtml.

3) Наши кинозалы – cinemaHalls.xhtml.

4) О кинотеатре – index.xhtml.

5) Акции и скидки – discounts.xhtml.

Переход на остальные страницы происходит из основных разделов веб-сайта.

Веб-сайт состоит из 10 страниц:

1) index.xhtml – страница, которая содержит приветствие, и ознакомительную информацию о кинотеатре, такую как адрес, контактные данные, а также информация о услугах, предоставляемых кинотеатром.

2) discounts.xhtml – страница, которая содержит информацию о скидках и акциях, действующих в кинотеатре.

3) register.xhtml – страница, которая содержит форму, на которой необходимо ввести следующую информацию: логин, пароль, повтор пароля, фамилия, имя, отчество, дата рождения. Регистрация в системе происходит по нажатию кнопку после заполнения всех полей.

4) cinemaHalls.xhtml – страница, на которой расположены данные о кинозалах, а также схемы кинозалов.

5) vacations.xhtml – страница, которая содержит информацию об открытых вакансиях кинотеатра, а также данные для связи с отделом кадров кинотеатра.

6) schedule.xhtml – страница, на которой представлена информация о фильмах (название, картинка, жанр, режиссер, страна, описание фильма, длительность фильма), идущих в кинотеатре в данный момент. Под каждым фильмом находится две ссылки: ссылка перехода на страницу комментариев об этом фильме (commentaries.xhtml), и ссылка перехода на страницу просмотра сеансов этого фильма (sessions.xhtml).

7) commentaries.xhtml – страница, на которой представлена информация о конкретном фильме (название, картинка, жанр, режиссер, страна, описание фильма, длительность фильма), а также панель комментариев. На панели комментариев показаны все комментарии, относительно данного фильма. Имеется форма для добавления комментария, а также ссылка для удаления комментария, находящаяся справа от своего комментария.

8) sessions.xhtml – страница, на которой представлена информация о сеансах конкретного фильма. На странице находится таблица сеансов (дата и время, цена билета, ссылка на переход на страницу просмотра мест на сеанс), а также информация о конкретном фильме (название, картинка, жанр, режиссер, страна, описание фильма, длительность фильма).

9) busyPlaces.xhtml – страница просмотра мест на определенный сеанс. На странице представлена следующая информация : название фильма, время начала сеанса, цена билета, номер кинозала, схема кинозала. Схема кинозала состоит из рядов и мест, место может быть закрашено одним из трёх цветов:

фиолетовый – место свободно, зеленый – место забронировано, красный – место занято. При нажатии на свободное место, произойдет переход на страницу бронирования места (reservePlacing.xhtml). При нажатии на занятое или забронированное место ничего не произойдет.

10) reservePlacing.xhtml – страница бронирования места. На странице представлена информация, такая как ряд, место, цена билета, а также форма для ввода данных, состоящая из полей: фамилия, имя, отчество. Также на этой странице находится radioButton «Оплатить часть билета» бонусными баллами, и поле для ввода количества бонусных баллов, а также итоговая цена билета с учетом бонусных баллов. Если пользователь авторизован в системе, то фамилия, имя, отчество заполняются автоматически. Бронирование билета происходит по нажатию кнопки.

11) bonusScore.xhtml – страница информации о бонусном счете. На странице показано количество баллов, которое в данный момент имеется на бонусном счете, а также история операций с баллами за последнее время.

**3. Слой бизнес-логики.**

Слой бизнес-логики содержит классы, ответственное за выполнение основных функций приложения. Эти классы будут сессионными бинами Enterprise Java Beans. Они связаны с базой данных с помощью классов сущностей JPA. В CDI-компонентах происходит вызов методов классов бизнес-логики, обработка данных, затем результаты отображаются на фэйслетах. Вызов методов классов бизнес-логики в CDI-бинах будет происходить через локальный интерфейс.

Ниже представлены основные классы, которые реализуют все основные методы бизнес-логики приложения:

1. RegAndAuthBean – Stateful-бин, отвечающий за регистрацию/авторизацию.

- logIn(String login, String password) : Boolean – авторизация пользователя в системе, с помощью логина и пароля;

- logOut() : Boolean – выход пользователя из системы;

- register(Client client, Account account) : Boolean – регистрация пользователя в системе;

- checkOnTwoMatchesOfLogin(String login, String login2) : Boolean – метод, с помощью которого происходит проверка двух логинов на совпадение;

- checkLoginAndPassword(String login, String password) : Boolean – метод, с помощью которого происходит проверка логина и пароля на соответствие;

2) FilmActionBean – бин, отвечающий за управление информацией о фильмах.

- showAllFilms() : List<Films> - показать информацию обо всех фильмах;

- showSessionsOfFilm(Film film) : List<Session> - показать все сеансы фильма.

- showPlacesOnSession(Session session) : List<Tickets> - показать все свободные/занятые/забронированные места на сеанс.

- showAllCommentariesOfFilm(Film film) : List<Commentary> - показать все комментарии к фильму.

- addCommentaryToFilm(Commentary commentary, Film film) : Boolean – добавить комментарий к фильму;

-deleteCommentary(Commentary commentary) : Boolean – удалить свой выбранный комментарий;

-findBonusScoreOfClient(Client client) : Bonusscore – найти бонусный счёт клиента;

-reservePlaceWithBonusPoints(Session session, Client client, Ticket ticket, History history, Bonusscore bonusscore, int score, Discount discount) : Boolean – забронировать место с расчетом цены билета, с помощью бонусных баллов. Этот метод содержит в ***себе распределенные транзакции***. Внутри этого метода вызываются методы createDiscountForSession, minusScoreFromBonusscore, addNewRecordToHistory, reservePlaceWithBonusPoints. После того, как выполняется последний метод, происходит commit (фиксирование) изменений, внесенных в базу данных. В противном случае, если хоть один метод не выполнился, происходит rollback(откат) изменений, внесенных в базу данных.

3) OtherActionBean – бин, отвечающий за остальные действия пользователь на сайте.

-showAllVacations() : List<Vacation> – показать все вакансии;

-showInfoAboutCinema() : String – показать информацию о кинотеатре;

-showCinemaHalls() : List<CinemaHall> - показать залы кинотеатра;

**Распределенная транзакция.**

В распределённой транзакции будет выполняться бронирование билетов с учётом бонусных баллов (метод reservePlaceWithBonusPoints). Диаграмма последовательности для распределенной транзакции представлена на рисунке 3.



Рисунок 3.Диаграмма последовательности.

1) createDiscountForSession(Discount discount) : Integer – рассчитать итоговую цену билета, создать скидку на сеанс. Метод возвращает итоговую цену билета.

2) minusScoreFromBonusscore(Bonusscore bonusscore, int count) : Boolean – вычесть бонусные баллы с бонусного счёта. При вызове этого метода указанное количество бонусных баллов снимается со счёта клиента.

3) addNewRecordToHistory(History history) : Boolean – добавить новую запись в историю;

4) reservePlace(Session session, Client client, Ticket ticket) : Boolean – забронировать место на сеанс. При вызове этого метода добавляется запись в таблицу «Ticket».

**Диаграмма деятельности бизнес-процесса «Бронирование билета с учётом бонусных баллов».**

На рисунке 4 представлена диаграмма деятельности бизнес-процесса «Бронирование билета с учетом бонусных баллов».

Описание диаграммы деятельности «Бронирование билета с учетом ББ».

1. Посмотреть список фильмов – клиент просматривает список фильмов, которые идут в кинотеатре (метод showAllFilms).
2. Выбрать конкретный фильм – клиент выбирает конкретный фильм из списка фильмов, которые идут в кинотеатре.
3. Выбрать конкретный сеанс – клиент просматривает список сеансов, которые относятся к данному фильму (метод showSessionsOfFilm), и выбирает конкретный сеанс.
4. Выбрать место в зале – Клиент просматривает места на фильм (метод showPlacesOnSession), и выбирает место, которое ему нравится.
5. Место свободно? Если статус места – «Свободно», то переходим к пункту 6, в противном случае возвращаемся в к пункту 4.
6. Оплатить часть билета бонусными баллами? Если да, переходим к пункту 7, в противном случае переходим к пункту 11.
7. Ввод количества бонусных баллов для оплаты – клиент вводит количество бонусных баллов, которые пойдут как часть стоимости билета.
8. Имеется ли столько баллов у клиента на счету? Если имеется, то переходим к пункту 9. В противном случае, возвращаемся к пункту 7.
9. Количество указанных бонусных баллов меньше 70% стоимости билета? Если да, то переходим к пункту 10, в противном случае, возвращаемся к пункту 7.
10. Расчёт стоимости билета – система рассчитывает стоимость билета, исходя из количества бонусных баллов, указанных клиентом (метод createDiscountForSession).
11. Снятие со счета бонусных баллов – система снимает количество бонусных баллов со счета клиента, рассчитанных в шаге 10 (метод minusScoreFromBonusscore).
12. Добавление записи в историю – происходит фиксирование операции о снятии бонусных баллов с бонусного счета (шаг 11) (метод addNewRecordToHistory).
13. Изменение статуса места – статус места меняется на «Забронировано» (метод reservePlace).



Рисунок 4. Диаграмма деятельности «Бронирование билета с учетом ББ»

**4. Слой доступа к данным.**

На этом слое располагаются располагаются компоненты(классы), отвечающие за доступ к данным. Эти классы представлены классами-сущностями JPA, сгенерированными по таблицам базы данных. На рисунке 5 представлена диаграмма классов слоя доступа к данным.

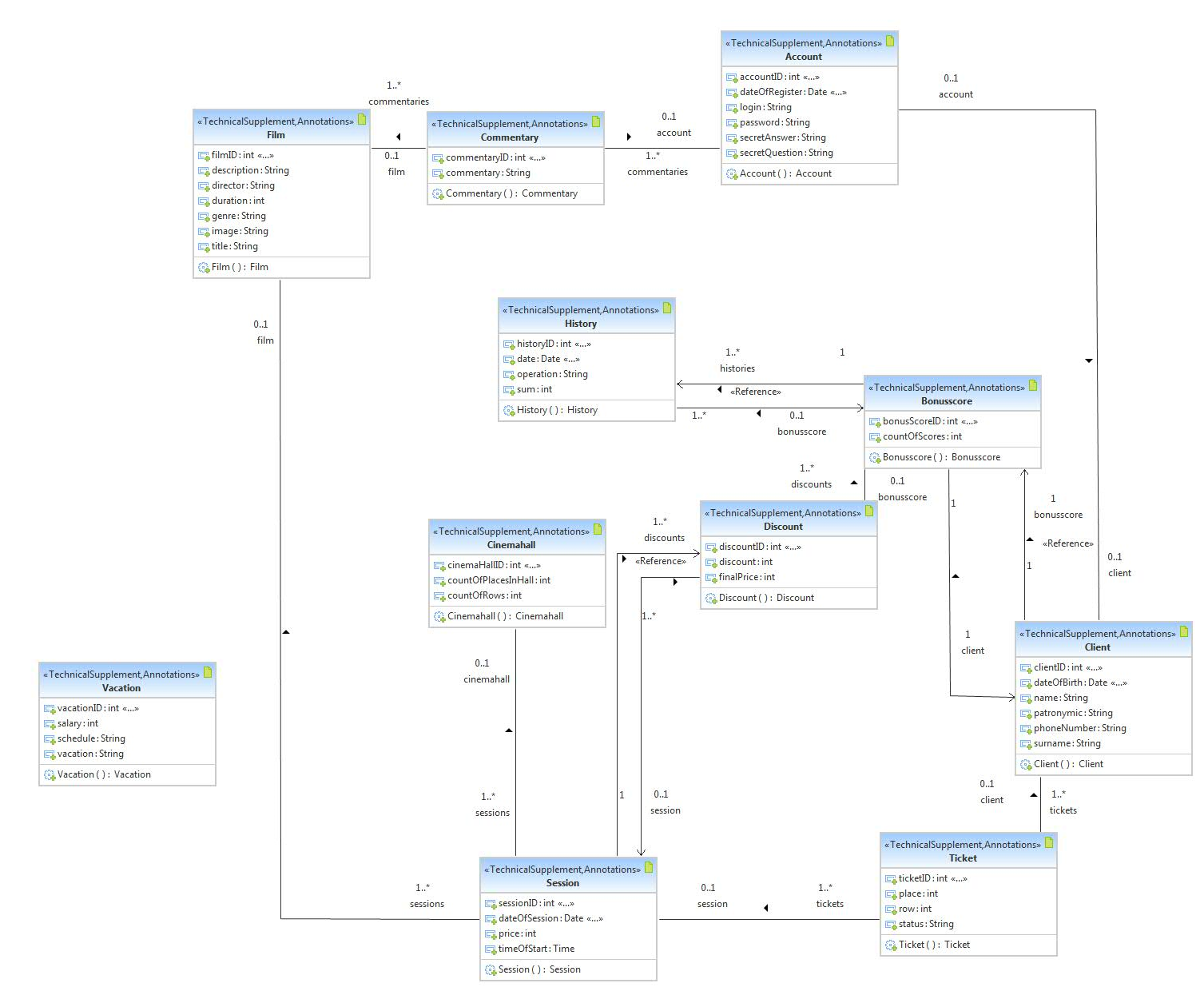


Рисунок 5. Диаграмма классов слоя доступа к данным.

**Описание классов.**

1. Film – данный класс представляет собой фильм, который проходит в кинотеатре в данный момент.

**Атрибуты:**

+ filmID : int - уникальный идентификатор фильма;

+ descrition : String - словесное описание основных событий и действий фильма;

+ director : String - фамилия и инициалы режиссёра фильма;

+ duration : int - длительность фильма в минутах;

+ genre : String - жанр фильма;

+ image : String - ссылка на картинку-скриншот фильма;

+ title : String - название фильма;

+ sessions : List<Session> - список сеансов данного фильма;

+ commentaries : List<Commentary> - список комментариев к фильму;

1. Cinemahall – данный класс представляет собой кинозал, в котором проходит фильм.

**Атрибуты:**

+ cinemaHallID : int - уникальный идентификатор кинозала;

+ countOfPlacesInHall : int - количество мест в зале;

+ countOfRows : int - количество рядов в зале;

+ sessions : List<Session> - список сеансов, которые проходят в данном кинозале;

1. Session – данный класс представляет собой сеанс показа фильма.

**Атрибуты:**

+ sessionID : int - уникальный идентификатор сеанса;

+ dateOfSession : Date - дата сеанса;

+ price : int - цена билета на сеанс;

+ timeOfStart : Time - время начала сеанса;

+ discounts : List<Discount> - список скидок на сеанс;

+ film : Film - фильм, по которому проходят сеансы;

+ cinemaHall : Cinemahall - зал, в котором проходят сеансы;

+ tickets : List<Ticket> - список билетов на сеанс фильма;

1. Ticket – данный класс представляет собой билет на сеанс фильма.

**Атрибуты:**

+ ticketID : int - уникальный идентификатор билета;

+ place : int - номер места в зале;

+ row : int - номер ряда в зале;

+ status : String - статус места в зале (занято/забронировано/свободно);

+ client : Client - клиент, приобрёвший билет на сеанс;

+ session : Session - сеанс, на которой был приобретен билет;

1. Client – данный класс представляет собой клиента, посетителя кинотеатра, который покупает билеты на сеансы фильмов.

**Атрибуты:**

+ clientID : int – уникальный идентификатор клиента;

+ dateOfBirth : Date – дата рождения клиента;

+ name : String – имя клиента;

+ surname : String – фамилия клиента;

+ patronymic : String – отчество клиента;

+ phoneNumber : String – номер телефона клиента;

+ bonusScore : Bonusscore – бонусный счёт клиента;

+ account : Account – учетная запись клиента;

+ tickets : List<Ticket> - список билетов, приобретенных клиентом;

1. Account – данный класс представляет собой учетную запись клиента в системе.

**Атрибуты:**

+ accountID : int – уникальный идентификатор учетной записи;

+ dateOfRegister : Date – дата регистрации учетной записи;

+ login : String – логин учетной записи;

+ password : String – пароль учетной записи;

+ secretQuestion : String – секретный вопрос на восстановление учетной записи;

+ secretAnswer : String – ответ на секретный вопрос;

+ client : Client – клиент-владелец учётной записи;

+ commentaries : List<Commentary> - список комментариев, написанных с этой учетной записи;

1. Commentary – данный класс представляет собой комментарий, сделанный к фильму.

**Атрибуты:**

+ commentaryID : int – уникальный идентификатор комментария;

+ commentary : String – текст комментария;

+ film : Film – фильм, к которому сделан комментарий;

+ account : Account – учетная запись, с которой сделан комментарий;

1. Vacation – данный класс представляет собой вакансию, активную в данный момент в кинотеатре.

**Атрибуты:**

+ vacationID : int – уникальный идентификатор вакансии;

+ salary : int – заработная плата;

+ schedule : String – график работы;

+ vacation : String – название вакансии;

1. BonusScore – данный класс представляет собой счёт клиента, на котором хранятся его бонусные баллы.

**Атрибуты:**

+ vacationID : int – уникальный идентификатор счёта клиента;

+ countOfScores : int – количество очков на счету;

+ discounts : List<Discount> - список скидок, сделанных на билет, которые задействовали этот бонусный счет;

+ histories : List<History> - список операций, проведенных с бонусными баллами этого счёта;

10) Discount – данный класс представляет собой скидку на покупку билета.

**Атрибуты:**

+ discountID : int – уникальный идентификатор скидки;

+ discount : int – скидка на билет в рублях;

+ finalPrice : int – итоговая цена билета после вычета скидки;

+ session : Session – сеанс, на который была сделана скидка;

+ bonusscore : Bonusscore – бонусный счет, с которого была сделана скидка;

11) History – данный класс представляет собой историю операций с бонусным счетом.

**Атрибуты:**

+ historyID : int – уникальный идентификатор истории;

+ date : Date – дата проведения операции;

+ operation : String – название операции (добавление/снятие баллов);

+ sum : int – сумма, на которую была проведена операция;

+ bonusScore : Bonusscore – бонусный счет, с которым была проведена операция;