**Упражнение по софтуерно инженерство: Проектиране и разработка на софтуерно приложение**

**Задача 1: Анализ на изискванията**

1. **Контекст**: Трябва да разработите софтуерно приложение за управление на библиотека. Приложението трябва да позволява на потребителите да добавят нови книги, да търсят по автор, заглавие или жанр, както и да проследяват състоянието на наличните книги (налични, заети, върнати).
2. **Изисквания**:
   * Приложението трябва да има възможност за регистрация на потребители.
   * Потребителите трябва да могат да търсят книги по различни критерии (автор, жанр, заглавие).
   * Системата трябва да поддържа различни категории книги и да ги показва на потребителя.
   * Всеки потребител трябва да може да вижда текущия статус на книгите (наличност и дата на заемане).
3. **Задача**:
   * Опишете какви изисквания ще поставите към системата.
   * Разработете кратко описание на функционалността на софтуера.
   * Напишете списък с функционални и нефункционални изисквания.

**Задача 2: Проектиране на система (Use Case диаграма)**

1. **Задача**:
   * Създайте **Use Case диаграма**, която описва основните функционалности на вашето приложение. Тя трябва да включва следните актьори и функции:
     + Актьор: Потребител
       - Функции: Търсене на книги, Записване на нова книга, Проверка на състояние на книга.
     + Актьор: Администратор
       - Функции: Добавяне/редактиране на книги, Управление на потребители.
   * Опишете ролите на актьорите и какви операции могат да извършват.

**Задача 3: Проектиране на база данни (ER диаграма)**

1. **Задача**:
   * Проектирайте основна **ER (Entity-Relationship) диаграма** за вашето приложение. Включете следните основни таблици:
     + **Книги**: съдържа информация за заглавие, автор, жанр, състояние (наличност, заемане).
     + **Потребители**: съдържа информация за име, email, потребителска роля (потребител, администратор).
     + **Заемания**: връзка между потребители и книги, която съдържа дата на заемане и дата на връщане.
   * Изберете подходящи връзки между таблиците и опишете тяхната роля в системата.

**Задача 4: Разработка на алгоритъм**

1. **Задача**:
   * Напишете алгоритъм за обработка на **заявка за заемане на книга** от потребител. Алгоритъмът трябва да проверява дали книгата е налична и ако е, да я маркира като заета и да добави запис в таблицата "Заемания".

Примерен алгоритъм на псевдокод:

plaintext

Copy

Функция заемане\_книга(потребител, книга):

ако книга.наличност == "налична":

книга.наличност = "заета"

добави\_в\_таблица\_заемания(потребител, книга)

върни "Заемането е успешно"

иначе:

върни "Книгата не е налична"

1. **Допълнителна задача**:
   * Напишете алгоритъм за търсене на книга по автор или заглавие.

**Задача 5: Тестване и отстраняване на грешки**

1. **Задача**:
   * Подгответе тестови случаи за:
     + Търсене на книга по автор.
     + Добавяне на нова книга.
     + Заемане на книга.
     + Показване на състоянието на книгата.
   * За всеки тестов случай, дефинирайте входни и очаквани изходни стойности.

**Решение и обяснение**

След като завършите тези задачи, ще трябва да анализирате и обясните как проектът се вписва в рамките на основните принципи на софтуерното инженерство, като:

* Управление на изискванията
* Проектиране на архитектура
* Алгоритми и структури от данни
* Тестване на софтуера

Това упражнение покрива основни аспекти на софтуерното инженерство, включително анализ на изискванията, проектиране, база данни, алгоритми, и тестове.

**Use Case диаграма за система за управление на библиотека**

**Актьори:**

1. **Потребител**: Лице, което използва библиотеката за заемане на книги.
2. **Администратор**: Лице, което управлява библиотеката, добавя нови книги и потребители.

**Основни Use Case-и:**

1. **Потребител**:
   * **Търсене на книги**: Потребителят може да търси книги по автор, заглавие или жанр.
   * **Заемане на книга**: Потребителят може да заеме книга, ако тя е налична.
   * **Преглеждане на наличности**: Потребителят може да види дали книгата, която иска да заеме, е налична или заета.
2. **Администратор**:
   * **Добавяне на книга**: Администраторът може да добави нова книга в библиотеката.
   * **Редактиране на книга**: Администраторът може да редактира информацията за книга (например, коригиране на заглавие, автор и т.н.).
   * **Управление на потребители**: Администраторът може да добавя нови потребители или да премахва съществуващи потребители от системата.
   * **Преглеждане на заемания**: Администраторът може да преглежда списъка със заети книги и да проследява кой ги е заел.

**Use Case диаграма:**

+------------------+ +-----------------+

| Потребител | | Администратор |

+------------------+ +-----------------+

| |

+------------------+ +-----------------------+

| Търсене на книги | | Добавяне на книга |

+------------------+ +-----------------------+

| |

+---------------------+ +----------------------+

| Заемане на книга | | Редактиране на книга |

+---------------------+ +----------------------+

| |

+-----------------------+ +------------------------+

| Преглеждане на наличности| | Управление на потребители|

+-----------------------+ +------------------------+

|

+--------------------------+

| Преглеждане на заемания |

+--------------------------+

**Обяснение на Use Case диаграмата:**

1. **Потребител**:
   * Потребителят може да извършва следните дейности:
     + **Търсене на книги**: Потребителят има възможност да търси книги по различни критерии като заглавие, автор и жанр.
     + **Заемане на книга**: Ако книгата е налична, потребителят може да я заеме.
     + **Преглеждане на наличности**: Потребителят може да проверява дали желаната книга е налична или е заета.
2. **Администратор**:
   * Администраторът има повече права и може да извършва следните дейности:
     + **Добавяне на книга**: Администраторът добавя нови книги в системата, като попълва необходимите полета (напр. заглавие, автор, жанр).
     + **Редактиране на книга**: Администраторът може да актуализира информацията за съществуваща книга, например ако информацията за автора или състоянието на книгата е променена.
     + **Управление на потребители**: Администраторът може да добавя нови потребители или да премахва съществуващи.
     + **Преглеждане на заемания**: Администраторът има достъп до информация за всички заети книги и може да вижда кой потребител е заел коя книга.

**Как се използва Use Case диаграмата:**

* **Use Case диаграмата** помага да се визуализират основните функции на системата и взаимоотношенията между различни потребители (актьори) и функциите, които могат да изпълняват.
* Тази диаграма е полезна в началния етап на проектирането на софтуерни системи, защото дава яснота за необходимите функционалности и помага при анализа на изискванията.

Това е основен пример за Use Case диаграма, който може да бъде разширен с допълнителни функции и детайли в зависимост от изискванията на конкретния проект.

Ето пример за **ER диаграма** (Entity-Relationship диаграма) на базата данни за система за управление на библиотека, базирана на предходната задача.

**Проектиране на база данни (ER диаграма)**

**Съществени таблици и тяхната структура:**

1. **Книги**:
   * **book\_id** (PK): Уникален идентификатор на книгата.
   * **title**: Заглавие на книгата.
   * **author**: Автор на книгата.
   * **genre**: Жанр на книгата.
   * **availability**: Състояние на книгата (например "налична", "заета", "върната").
2. **Потребители**:
   * **user\_id** (PK): Уникален идентификатор на потребителя.
   * **name**: Име на потребителя.
   * **email**: Email адрес на потребителя.
   * **role**: Роля на потребителя (например "потребител", "администратор").
3. **Заемания**:
   * **loan\_id** (PK): Уникален идентификатор на заема.
   * **book\_id** (FK): Идентификатор на книгата, която е заемана.
   * **user\_id** (FK): Идентификатор на потребителя, който е заел книгата.
   * **loan\_date**: Дата на заемане на книгата.
   * **return\_date**: Дата на връщане на книгата.

**ER диаграма:**

+------------------+ +------------------+ +------------------+

| Книги | | Потребители | | Заемания |

+------------------+ +------------------+ +------------------+

| book\_id (PK) |<--------| user\_id (PK) | | loan\_id (PK) |

| title | | name | | book\_id (FK) |

| author | | email | | user\_id (FK) |

| genre | | role | | loan\_date |

| availability | +------------------+ | return\_date |

+------------------+ +------------------+

| |

+-------------------------------------------------------+

(FK - foreign key)

**Обяснение на ER диаграмата:**

1. **Книги (Books)**:
   * Таблицата "Книги" съдържа информация за всяка книга в библиотеката.
   * **book\_id** е уникалният идентификатор на всяка книга (първичен ключ).
   * **title**, **author**, **genre** и **availability** съдържат информация за заглавие, автор, жанр и състояние на наличността на книгата.
2. **Потребители (Users)**:
   * Таблицата "Потребители" съдържа информация за всички потребители на библиотеката.
   * **user\_id** е уникалният идентификатор на потребителя (първичен ключ).
   * **name**, **email** и **role** съдържат информация за името на потребителя, неговия email адрес и роля (например "потребител" или "администратор").
3. **Заемания (Loans)**:
   * Таблицата "Заемания" съхранява информация за заемането на книги от потребителите.
   * **loan\_id** е уникален идентификатор на заема (първичен ключ).
   * **book\_id** и **user\_id** са външни ключове, които се свързват с таблиците "Книги" и "Потребители", за да се отбележи кой потребител е заел коя книга.
   * **loan\_date** и **return\_date** записват съответно датата на заемане и датата на връщане на книгата.

**Връзки между таблиците:**

1. **Между таблицата "Книги" и таблицата "Заемания"**:
   * В таблицата "Заемания" полето **book\_id** е външен ключ, който се свързва с уникалния идентификатор на книгата от таблицата "Книги".
   * Това означава, че всяко заемане е свързано с конкретна книга.
2. **Между таблицата "Потребители" и таблицата "Заемания"**:
   * В таблицата "Заемания" полето **user\_id** е външен ключ, който се свързва с уникалния идентификатор на потребителя от таблицата "Потребители".
   * Това означава, че всяко заемане е извършено от конкретен потребител.

**Как се използва ER диаграмата:**

* ER диаграмата ви дава ясна представа за основните обекти и връзки в системата.
* Тя е полезна за проектиране на структурата на базата данни и за планиране на SQL заявки.
* Също така е основа за създаване на релационни таблици и управление на данни в базата данни.

Тази диаграма е основна и може да бъде разширена, ако се добавят нови функции или изисквания към системата (например добавяне на таблица за "Резервации" или "Категории книги").