NTT DaTa
Global IT Innovator

S. Aravindhan
184759

Mini Case Study

CD Gallery

Java Stream

NTT DATA, Inc.

TABLE OF CONTENTS

1 Specification

1.1 Objective

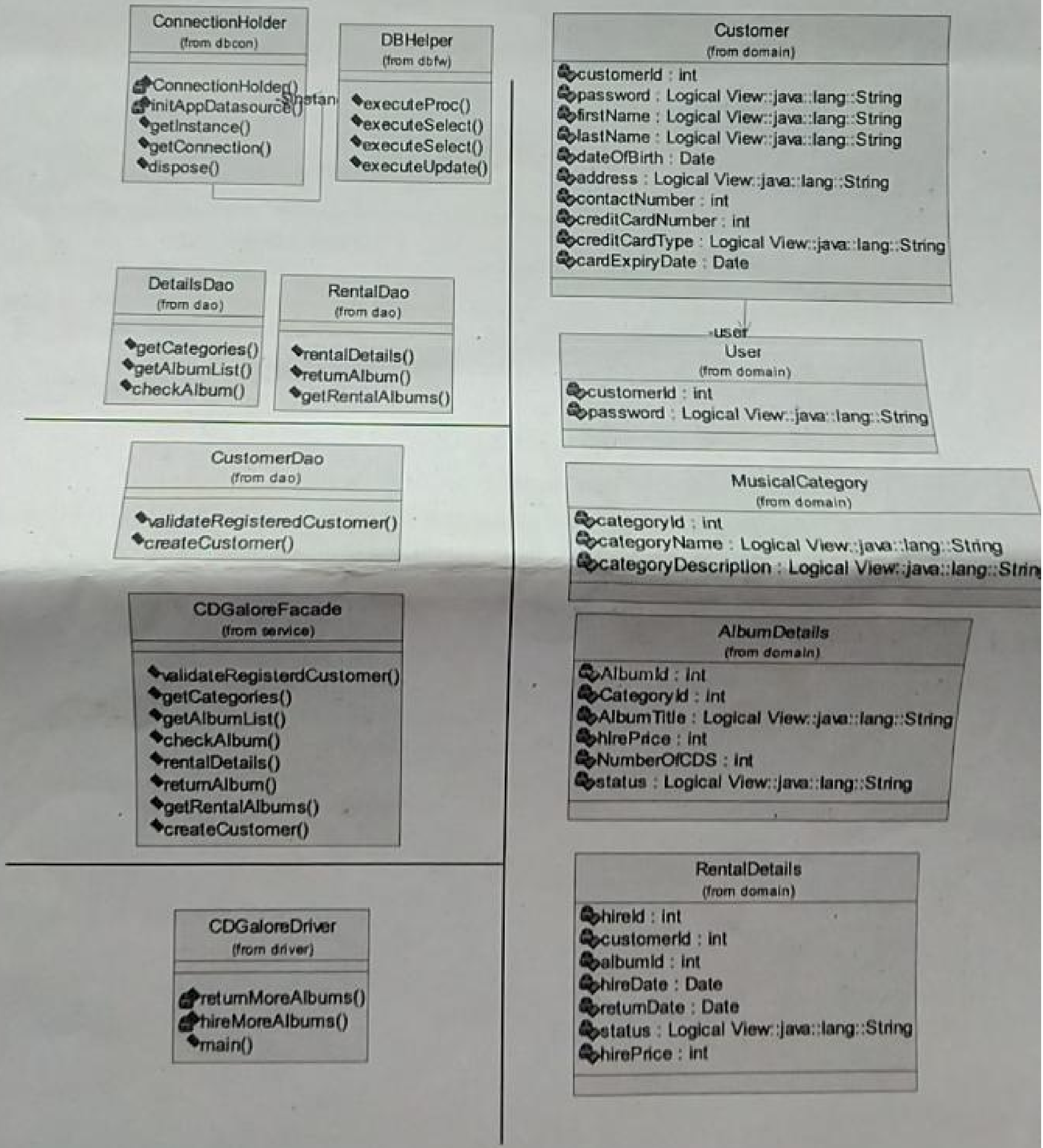
TO develop an online album application, which will enable the customer to view the list of albums, hire the albums and return the hired albums.

1.2 Scenarios

Role	Privileges
Admin	Register.
	View Albums.
	Hire Albums.
	Return Albums.

- New customer can register into the system.
- Existing customer can view the list of albums and check *their availability*.
- Customer can hire multiple albums.
- Customer can return multiple albums.

1.3 Architecture



1.4 Database Design

1.4.1 Tables

1.4.1.1 Customer

Column Name	Data Type	Constraints
CustomerId	Number(5)	Primary key
Password	Varchar2(20)	
First_Name	Varchar2(30)	
Second_Name	Varchar2(30)	
DateOfBirth	Date	
Address	Varchar2(60)	
ContactNumber	Number(9)	
CreditCardNumber	Number(16)	
CreditCardType	Varchar2(10)	
CardExpiryDate	Date	

1.4.1.2 User

Column Name	Data Type	Constraints
Userid	Number(5)	foreign key references customer(customerId)
Password	varchar2(25)	

1.4.1.3 MusicalCategory

Column Name	Data Type	Constraints
CategoryId	Number(10)	Primary key
CategoryName	Varchar2(45)	NOT NULL Unique
CategoryDescription	Varchar2(45)	

1.4.1.4 AlbumDetails

Column Name	Data Type	Constraints
AlbumId	Number(10)	Primary Key
CategoryId	Number(10)	Foreign key references musicalcategory(CategoryId)
AlbumTitle	Varchar2(30)	NOT NULL
HirePrice	Number(5,2)	
NumberOfCDs	Number(2)	NOT NULL
Status	Varchar2(4)	NOT NULL

1.4.1.5 RentalDetails

Column Name	Data Type	Constraints
HireId	Number(6)	Primary Key
CustomerId	Number(5)	Foreign key references Customer(CustomerId)
AlbumId	Number(10)	Foreign key references AlbumDetails(AlbumId)
HireDate	Date	
ReturnDate	Date	
Status	Varchar2(4)	NOT NULL
TotalHirePrice	Number(5,2)	

1.5 Java Components

1.5.1 Java Beans

com.nttdata.cdgallery.domain

1.5.1.1 User

Attribute Name	Data type
customerId	Private int
Password	Private String
Constructor public User(int cutomerId, String password)	

1.5.1.2 MusicalCategory

Attribute Name	Data type
categoryId	Private int
categoryName	Private String
categoryDescription	Private String

1.5.1.3 AlbumDetails

Attribute Name	Data type
albumId	Private int
categoryId	Private int
albumTitle	Private String
hirePrice	Private int
noOfCDs	Private int
Status	Private String
Constructor public AlbumDetails(int albumId,String albumTitle, int price,int noOfCDS, String Status)	

1.5.1.4 Customer

Attribute Name	Data type
customerId	Private int
Password	Private String
firstName	Private String
secondName	Private String
dateOfBirth	Private Date
Address	Private String
contactNo	Private int
creditCardNo	Private int
CreditCardType	Private String
CardExpiryDate	Private Date
User	Private User
Constructor	
public Customer(String password, String firstName, String lastName,Date dateOfBirth, String address, int contactNumber, int creditCardNumber, String creditCardType, Date cardExpiryDate)	

1.5.1.5 RentalDetails

Attribute Name	Data type
hireId	Private int
customerId	Private int
albumId	Private int
hireDate	Private Date
returnDate	Private Date
Status	Private String
totalPrice	Private int
rentalList	Private List
Constructor	
public RentalDetails(int hireId, int custId, int albumId, Date carryDate) Create a rentalDetails object to hire the album. public RentalDetails(int hireId,int price) Create a rentalDetails object to return the hired album.	

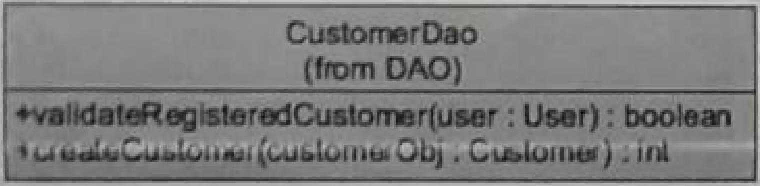
1.5.2 DAO classes

com.nttdata.cdgallery.dao

1.5.2.1 CDGalleryDAOException.java

- public CDGalleryDAOException(String message, Throwable cause)
- public CDGalleryDAOException(String message)

1.5.2.2 CustomerDAO.java


CustomerDao
(from DAO)
+validateRegisteredCustomer(user : User) : boolean
+createCustomer(customerObj : Customer) : int

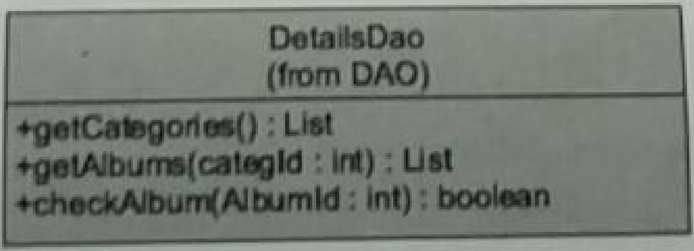
- public boolean validateRegisteredCustomer(User user) throws CDGalleryDAOException

  Create SQL query to validate the cutomerId and password

- *public int createCustomer(Customer customerObj) throws CDGalleryDAOException*

  Insert the record and fetch the last inserted ID and return.

1.5.2.3 DetailsDAO.java


DetailsDao
(from DAO)
+getCategories() : List
+getAlbums(categId : int) : List
+checkAlbum(AlbumId : int) : boolean

- public List<MusicalCategory> getCategories() throws CDGalleryDAOException

  Create an SQL statement to obtain all available categories. Return a list of Category objects.

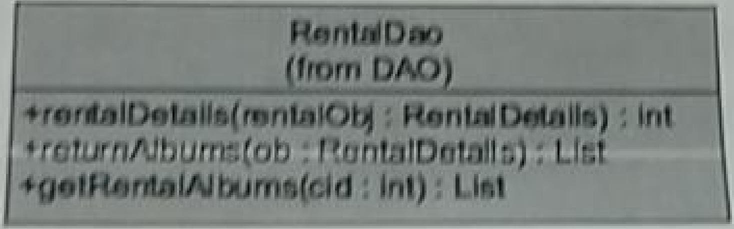
- public List<AlbumDetails> getAlbums(int categoryId) throws CDGalleryDAOException

  Create an SQL statement to obtain all the albums for the incoming categoryId. Return a list of Album Objects.

- public boolean checkAlbum(int albumId) throws CDGalleryDAOException

  Create an SQL statement to check the status of the album's availability for the incoming albumId is "A"(available) or " B"(borrowed). Return a true or false respectively.

1.5.2.4 RentalDAO.java



RentalDao
(from DAO)

+rentalDetails(rentalObj : RentalDetails) : int
+returnAlbums(ob : RentalDetails) : List
+getRentalAlbums(cid : int) : List

- *public int rentalDetails(RentalDetails rentalObj) throws CDGalleryDAOException*

  Update the status of the AlbumDetails table from "A" to "B". Insert a new record into the Rental Details table with the status "C" and return the RentalId.

- public List<AlbumDetails> getRentalAlbums(int customerId) throws CDGalleryDAOException

  Create an SQL statement to obtain all the albums for the incoming customerId with status "C" (carried). Return a list of Album Objects.

- public List<AlbumDetails> returnAlbums(RentalDetails rentalObj) throws CDGalleryDAOException

  Create and execute the below statements. Update the status of the AlbumDetails table from "B" to "A". Update the RentalDetails table change the status from "C" to "R". Update the return date with the sysdate and calculate the price

-CDGallery

thod

reId call the

er for all the

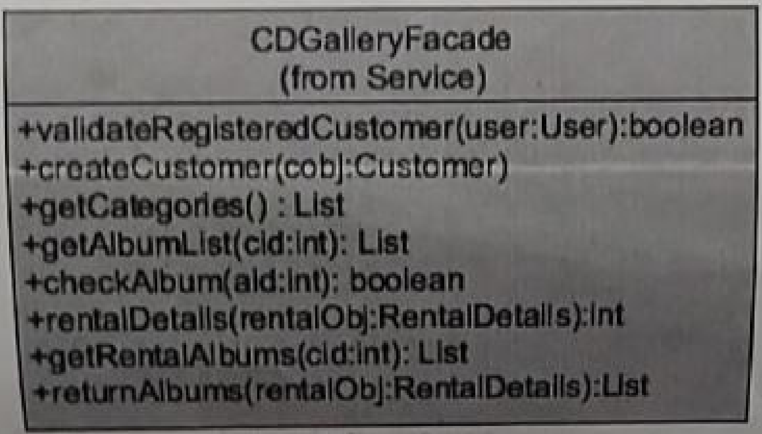
1.5.3 ServiceLayer

com.nttdata.cdgallery.service

1.5.3.1 InfraAppException.java

- public CDGalleryException(String message, Throwable cause)
- public CDGalleryException(String message)

1.5.3.2 CDGalleryFacade.java



```
CDGalleryFacade
(from Service)
+validateRegisteredCustomer(user:User):boolean
+createCustomer(cobj:Customer)
+getCategories() : List
+getAlbumList(cid:int): List
+checkAlbum(aid:int): boolean
+rentalDetails(rentalObj:RentalDetails):int
+getRentalAlbums(cid:int): List
+returnAlbums(rentalObj:RentalDetails):List
```

- public boolean validateRegisteredCustomer(User user) throw CDGalleryException

  Call validateRegisteredCustomer(user) from CustomerDAO.

- public int createCustomer(Customer customerObject)throws CDGalleryException

  Call createCustomer(customerObject)from CustomerDAO.

- public List<MusicalCategory> getCategories() throws CDGalleryException

  Call getCatagories()from DetailsDAO.

- public List<AlbumDetails> getAlbumList(int categoryId)throws CDGalleryException

  Call GetAlbumList(categoryId)from DetailsDAO.

- public boolean checkAlbum(int albumId) throws CDGalleryException

  Call checkAlbum(albumId) DetailsDAO.

- public int rentalDetails(RentalDetails rentalObj)throws CDGalleryException
  Call rentalDetails(rentalObj) from DetailsDAO.

- public List<AlbumDetails> getRentalAlbums(int cutomerId) throws CDGalleryException
  Call getRentalAlbums(customerId) from RentalDAO.

- public List<AlbumDetails> returnAlbums(RentalDetails rentalobj) throws CDGalleryException
  Call returnAlbums(rentalObj) from RentalDAO.

5.3.3 CDGalleryDriver.java

- *If the customer is an existing Customer then prompt to enter customerId anc password.*
  Call validateRegisteredCustomer(User user) from CourseRegFacade .

If the customer is a new customer then accept the password,firstname,lastName,date of birth,address,contact number,credit number, card type and card expiry date. Build a Customer object using this information and call createCustomer() by passing the Customer object.

- Display to the new registered customer his/her CustomerId.
  Either validation is successful or registration is successful then,
- Display two options
  - View Albums To Hire
  - Return The Hired Albums

View Albums to Hire

- Display all hired albums by passing the customerId to the method getRentalAlbums(customerid).
- Prompt to enter a hireId, create a rentalDetails object with hireId call the returnAlbums(rentalDetails) method.
- Display the hire price of the album returned.
- In the similar way display the total price to be paid by the customer for all the returned albums.

1.6 Implementation Specification

1.6.1 Mini Case Study General Specifications

1. Follow the naming conventions and best practices.
2. Exception handling is a must. Use custom exceptions.
3. DAO should not have any class-level member variables.
4. In DAO methods, always release the connection objects after using in the finally block.
5. DDLs for creating tables and populating data should be submitted in text format.
6. Create Domain Objects with getter and setter methods and use them to pass data between the Main class, facade and DAO class.
7. JUnit Test cases a must for each DAO.
8. Self and Peer Review for self-improvement.