FIT5042 Design Report (Major Tasks)

Banking Enterprise Application

Student Name: Duo Pan
Student ID: 27554074
Tutor Name: ABM Russel
Tutorial Time: Mon 18:00-20:00
Lecturer: Chris Messom

Table of Contents

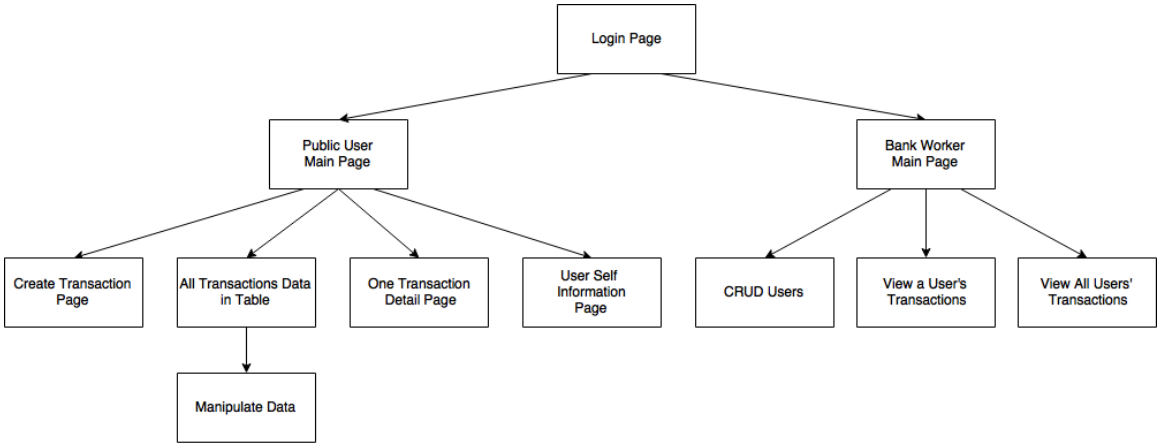
1. Overview

This assignment is to build a web-based Java prototype of a Banking Enterprise Application as well as a desktop client application version. Some Java enterprise technologies should be used in the assignment: JSF, RESTful web services, EJB, Persistence API and so on.

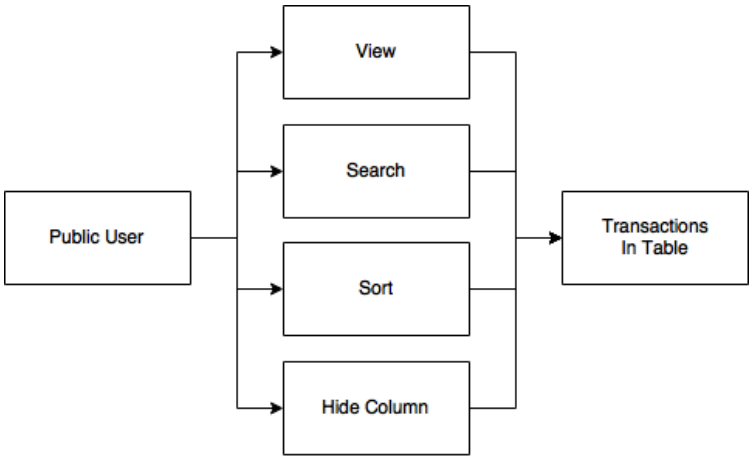
The basic function is that users can login in the system by their username and password. System will then redirect the user to the specific page. If the user is a public customer, he/she can view and edit his/her own information, and can view, create, sort, multiple search their transaction records. If it is a bank worker, he/she can view, create, update, delete and search users.

2. Functional diagram

When the application is running, it first displays login page. User can enter this system as a public user or bank worker, and he/she can be redirected to the relevant main page. Public user main page provides a table to display his/her own data, while bank worker main page display all users' data. Some buttons are set to navigate to other pages. Every other page has a specific function and a button to go back to the main page.

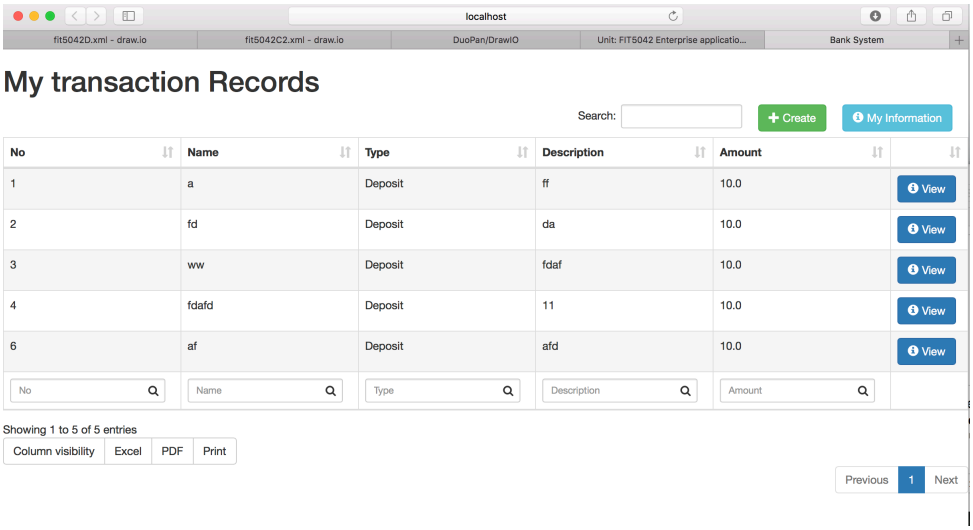


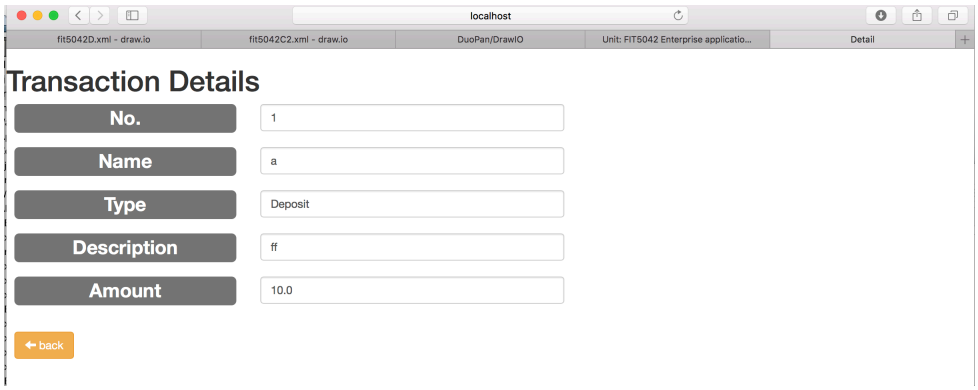
3. Core program functionality



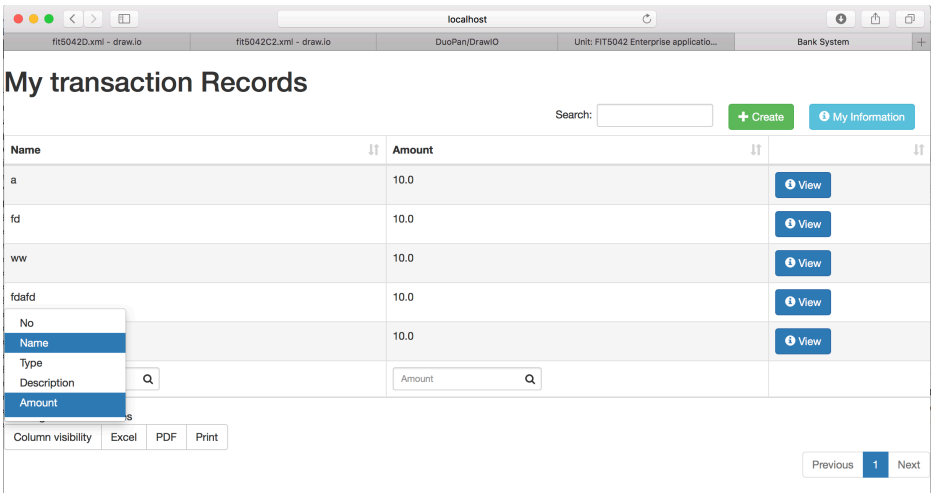
Use BootsFaces library to build the page, which is designed for JSF and based on Bootstrap. It provides a table control, and through changing its attributes, I can get functions I need.

- View all data in table, click view button, can see detail of this transaction record.

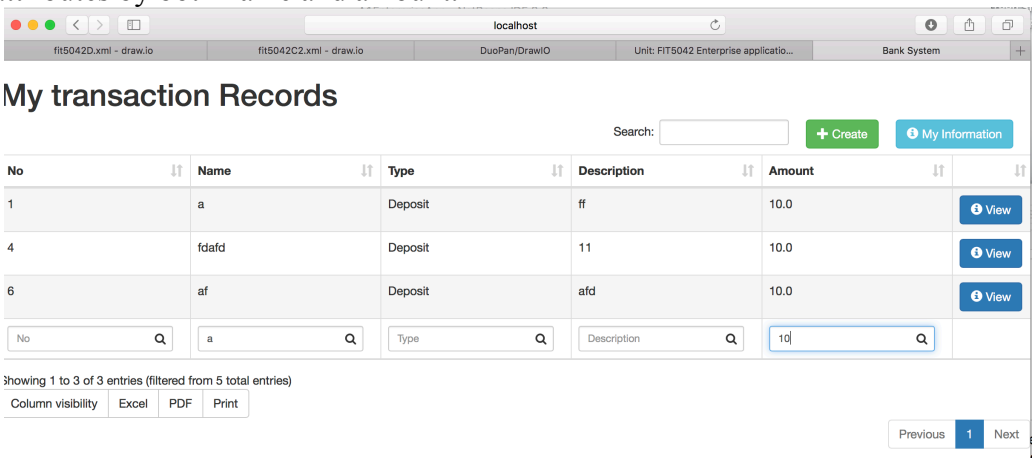




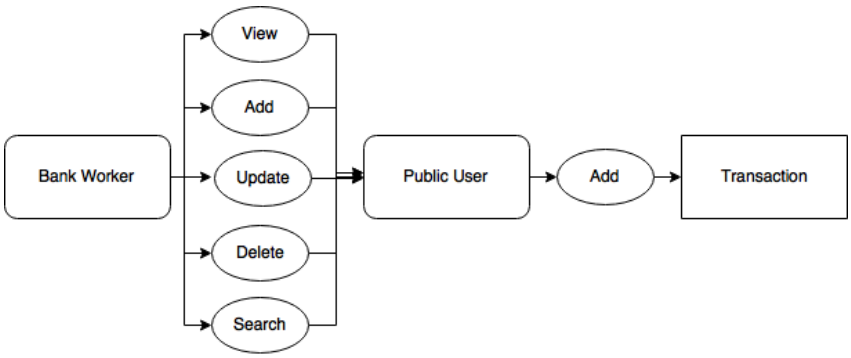
- Can choose which attributes to show on table.



- Multiple search. I put a search textbox under each column, so users can input data in the corresponding search textbox to do multiple search. The example search attributes by both name and amount.

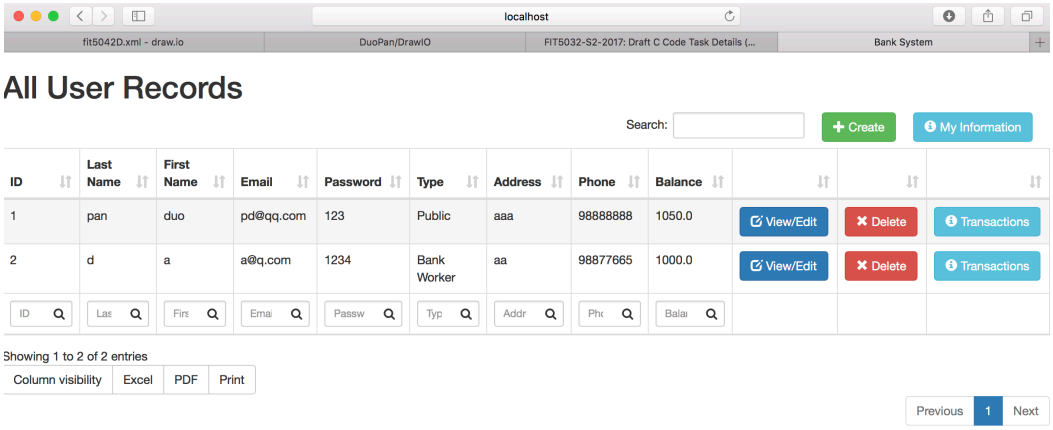


- Navigation: Provide buttons for user to view transaction details and come back to main page.

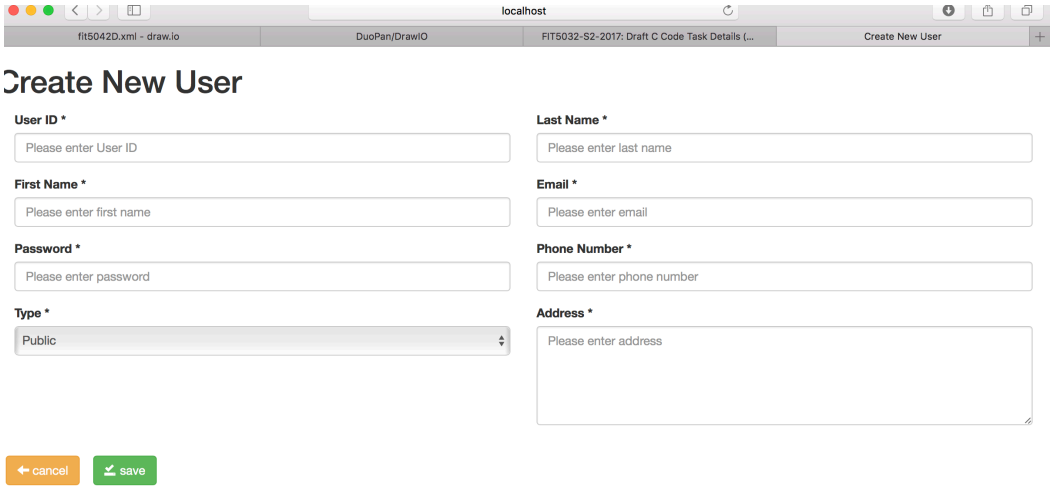


Bank worker can view/add/update/delete users, and can search by multiple attributes of a user.

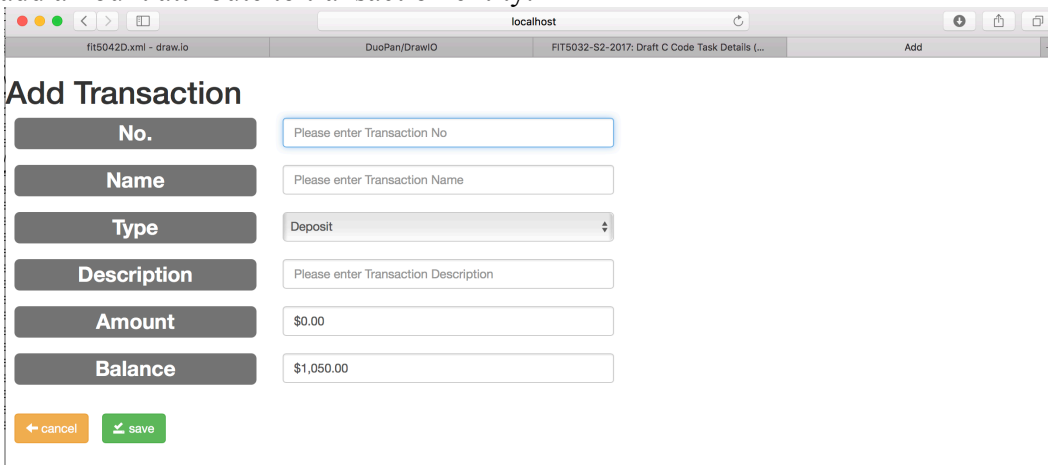
- Bank worker CRUD user functions:
  View all users / delete a user / view one user's all transactions:



Create/Edit a user:

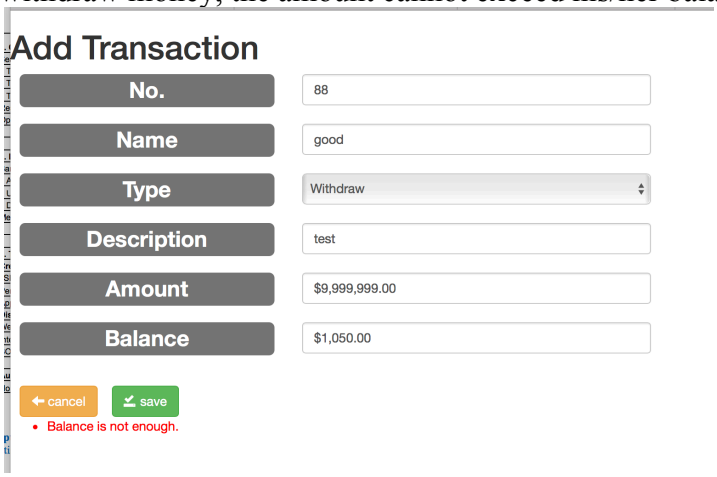


- Public user can create new transaction. I add balance attribute to user entity, and add amount attribute to transaction entity.



Validation:

- when user click save button, the system will do validation and prompt errors on the screen in red text.
- When adding records to database, the system will do validation on each attribute as the assignment required. Such as the format of email.
- When a user withdraw money, the amount cannot exceed his/her balance.



4. Usability Design Review

- Navigation: Each page has good navigation buttons to the other pages, they are colorful and easy to find;
- Familiarity: This web site did not adopt any strange layout, so it is familiar with most people;
- Consistency: This web application can run at different browsers.
- Error Prevention: User inputs are check before store into the database;
- Feedback: When mouse moves over table rows, it will highlight.
- Visual Clarity: Data table makes the web site easy to read and use. Red color on delete button also remains user be careful when clicking. The whole page is clear and tidy.
- Flexibility: Many buttons in a page, so that user can easily go to a page they want. All the pages are connected in graph structure instead of linear.

5. Checklist of site functionality

1. Credit Functionality	✔
Search for Transaction by	✔
Transaction Name,	✔
Transaction No	✔
Transaction Type	✔
Results with tabular format with heading.	✔
Option to view the full details	✔
2. Distinction Functionality	✔
Bank Workers can: View	✔
Add	✔
Update	✔
Delete Users	✔
Members of Public can make Transactions	✔
3. Technical Requirements	✔
Credit	✔
JSF web clients	✔
Persistence API	✔
Application managed entity manager or container managed entity manager.	✔
Distinction	✔
Web client is required	✔
Interaction between clients and database handled by EJBs	✔
BOTH Criteria API and JPQL	✔
Audit	✔
No breaking of copyright	✔

6. User stories

6.1 Public User

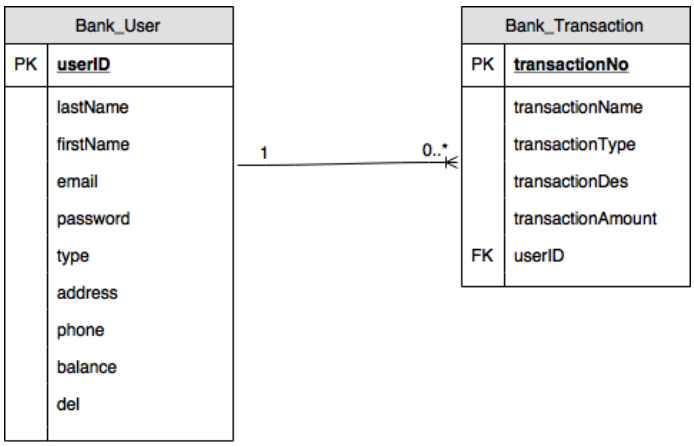
- Login the system via email address and password, then public user main page will show;
- View all transactions data of this user in a table. The user can search, sort and hide columns of this table;
- Create a new transaction;
- View and edit his/her personal information, such as address or phone number;
- Logout the system.

6.2 Bank Worker

- Login the system via email address and password, then bank worker main page will show;

- View all users' data in a table. The administer can search, sort and hide columns of this table;
- Create/delete/edit a user. Use hiding instead of deleting.
- View a user's all transactions, and can search by multiple attributes;
- View all users' transactions, and can search by multiple attributes;
- View and edit his/her personal information, such as address or phone number;
- Logout the system.

7. Entity relation diagram



There are two entities in database, user and transaction. One user has 0 to many transactions and one transaction can only belong to one user. User id is a foreign key in transaction table.
Some key attributes:

- "del" means the user is deleted or not;
- "balance" is implicitly set as 0 when the user is created;
- "type" is limited as "Public" or "Bank Worker";
- "userID" and "email" is unique;

8. Data dictionary

In managed bean file, I set some variables:

- List<BankTransaction> bankTransactionList: All transactions in database;
- BankTransaction currBankTransaction: A transaction to be view;
- List<User> userList: All users in database, only retrieve those do not be deleted (attribute "del" is "false");
- User currUser: A user to be handled, such as view detail, edit and delete;
- List<BankTransaction> currBankTransactionList: All transactions belong to currUser;
- static User loginUser: Keep track of the User who login in the system. When logout, it will be set null.

In entity class files: Most of the attributes are String and int type. And user class has a list of transactions variable while transaction class has a user variable.