CMPE 450
SOFTWARE ENGINEERING
OBJECT DESIGN DOCUMENT

Version 1.0

Project Name:	Stock Follow Up System Project
Submitted to Instructor :	AYŞE BAŞAR BENER
Asistant:	GÜL ÇALIKLI,ONUR GÜNGÖR
Submitted by:	**Design Group 3 Group**

ŞAHİN CEM GEYİK
HÜSEYİN KIRAN
NESLİHAN KUKUT
EMRE YILMAZ

Boğaziçi University, İstanbul
October 19th, 2006

Contents

List of Figures

1 Introduction

The purpose of this document is to summarize the design principles of Stock Follow Up System in detail. This document explains the implemented classes and their methods, table designs of databases and authority layers of different users. The document also provides information for the database system, the environment and the properties. However, it is not the final object design document; it may vary during the development process.

1.1 OBJECT DESIGN TRADE-OFFS

In our approach to design a new Stock Follow Up system for Bosporus University, we tried use what we have, in most efficient way, however we needed to give up on certain things to finish the project in budget

First trade-off is made on time to avoid from any extra cost, since the customer does not offer any money to buy something new. Therefore, the system may work slowly in case many users log into the system at the same time.

Second trade-off is made on space in order to decrease complexity. The system is design to be user friendly. Each user type has a different user-interface, as user rights vary depending on user's authority. After the graphical user interface is implemented, the system may cover much space, but the trade-off saves time. When a user lags in, the system calls the related user-interface instead of recreating a page for the current user depending on user's authority.

These trade-offs made our system fast and user friendly.

1.2 INTERFACE DOCUMENTATION GUIDELINES

In Stock Follow Up design, two packages are used, which are UserInterface and StockFollowUpClasses. The modules of StockFollowUpClasses are used by UserInterfaces.

The code of In Stock Follow Up System is to be written clearly in a certain order to make it understandable to every programmer. The names of classes in packages start with capital letter. In case the name of the class includes more than a word, the first letter of each word is capitalized. For instance, UserInterface consist of two different words "User" and "Interface" and their first letters are capitalized.

The names of the class methods are given in same manner. Their first letters have to be capital.

The names of the parameter used in program are given according to the type of the parameter. An acronym is used preceding the name of the parameter. e.g. txtUserName.

1.3 DEFINITIONS, ACRONYMS, AND ABBREVIATIONS

There are some definitions, acronyms and abbreviations that are frequently used in this odd as followings:

SFUS: Stock Follow Up System
ODD: Object design document
UML: Unified Modeling Language
PR user: Product User

1.4 REFERENCES

Stock Management Software, a similar software
http://www.strikedesigns.co.uk/stock_management_software.php

2

PACKAGES

2.1 PACKAGE DIAGRAM

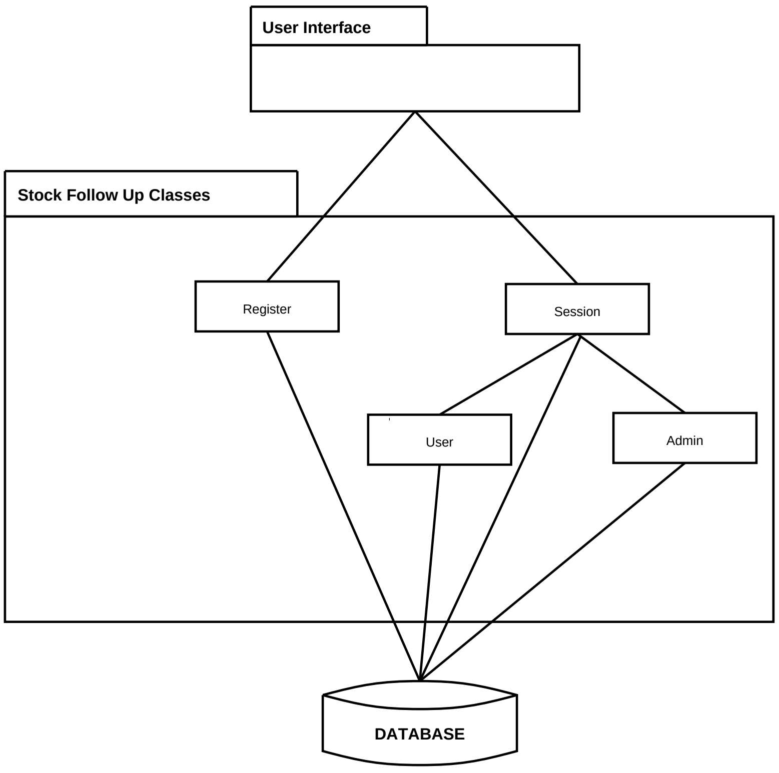

Figure 2.1: PACKAGE DIAGRAM of STOCK FOLLOW UP SYSTEM

2.2 PACKAGE DEFINITIONS

There are two packages in this project. First package is the User Interface package, which is designed for the user interface of the program. The second package is Stock Follow Up Classes package, which includes all classes used for the internal structure of the program.

2.2.1 USER Interface Package

User Interface Package is responsible of the handling of the operations between User and the program. The methods provided by this package are explained in chapter 3

2.2.2 Stock Follow Up Classes Package

This package includes all the classes necessary for the execution of the program. The classes in this package are Session, User, Admin and Register

When a User tries to login to the system, an object of the class Session is created and used. If the user wants to register to the system, then an object of the class Register is created and used. The class Register has access to the database, and a new pending user is created in the database after successful registration. If the user does not want to register, the user can login to the system by giving his user information. Session class has access to the database and using the method CheckPassword (), it checks if given information is valid. If so, login is successful and according to the type of the user, either an object of the class Admin or an object of the class User is created. These both classes have access to the database and their range of operation are determined by their methods. After the user is done with his work, he/she can logout from the system and for this purpose the Terminate() method of the Session class is called.

2.3 CLASS DIAGRAM

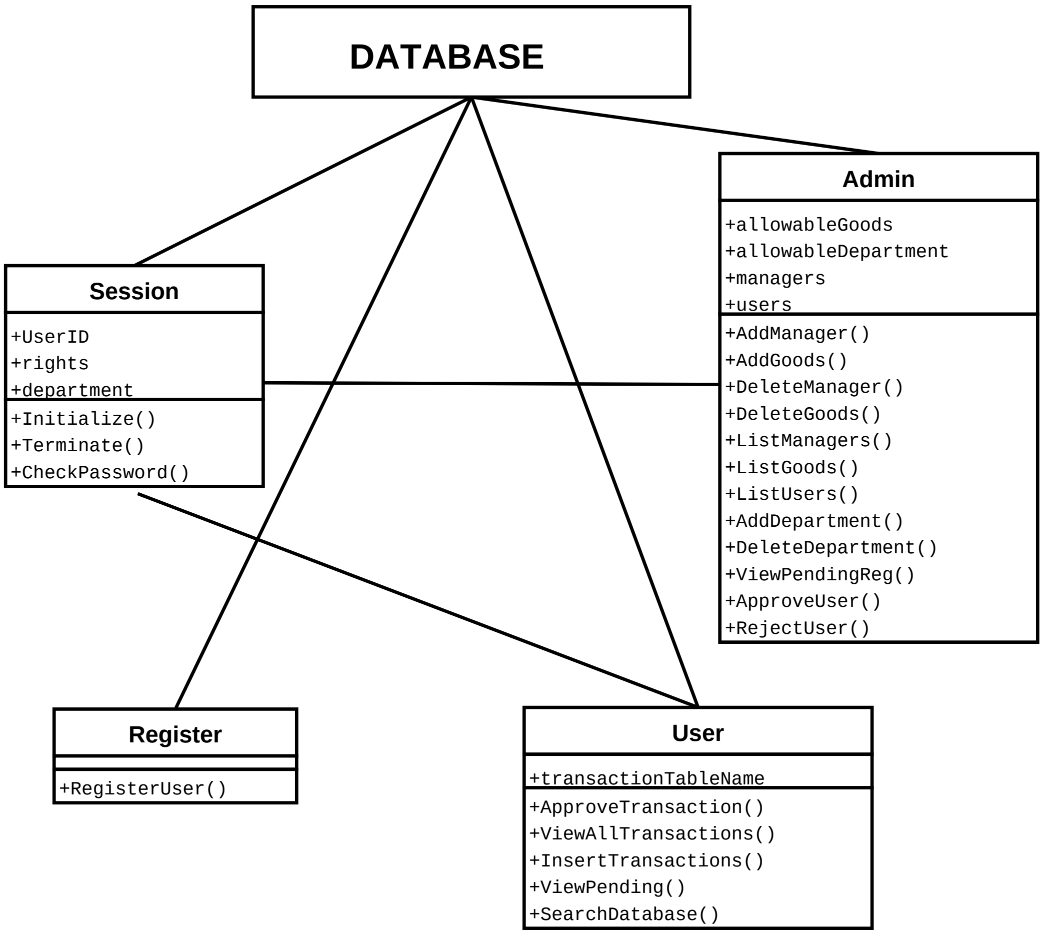

Figure 2.2: CLASS DIAGRAM

2.4 CLASS DEFINITIONS

2.4.1 SESSION Class

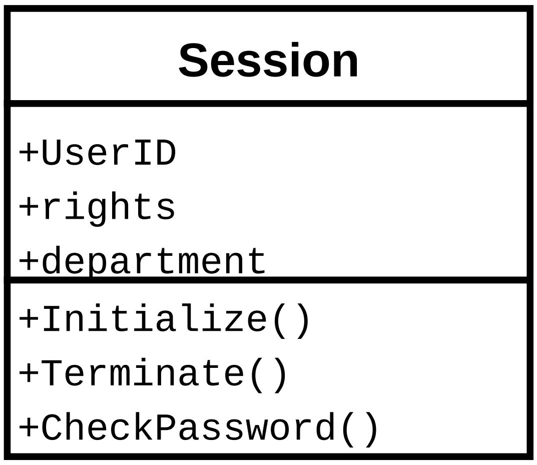

Figure 2.3: SESSION Class

Session class is responsible for checking UserId and password data, entered by user, relying on the data, kept in database and start a new session object.

CheckPassword( ) method access database and checks if the entered password and the password recorded on Password table in database match. If the password and UserId are valid, the module calls Initialize() method and set a value to ”rights” parameter; else the method gives an error message to user in order to tempt user to re-enter his password and username.

Initialize( ) method determines user’s rights, starts a session object for the current user according to ”rights” and ”department” parameters then redirects the current interface to the related user-interface.

Terminate( ) module terminates the session.

2.4.2 REGISTER Class

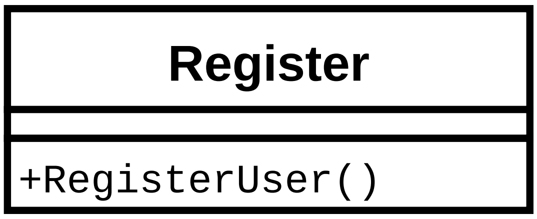

Figure 2.4: REGISTER Class

RegisterUser() method is used to add new users to the system. After user completed and submitted register form, the data, taken from form, will be inserted to database in case the system administer approves the user.

2.4.3 USER Class

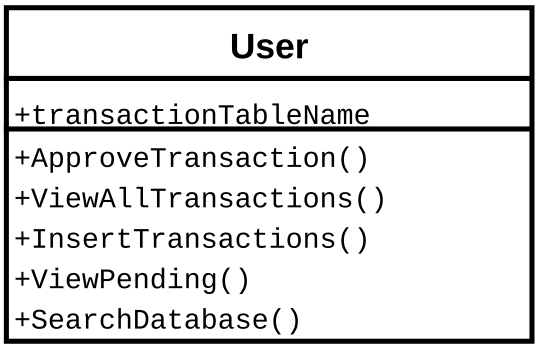

Figure 2.5: USER Class

The User class is a part of Stock Follow Classes package and handles the operations performed by the user of the Stock Follow Up System. There are six different types of user groups. Some methods described below can be accessed by a particular type of group due to the access rights.

The first method is ApproveTransaction, which takes the transactionID as input parameter and makes the approvalStatus of the transaction approved.This method can only be used by the employees of the purchasing department since they have the right to approve the transactions.

The second method is ViewAllTransactions, which returns a list of all transactions done so far. This method can be used by only the rector, employees and the manager of the purchasing department.

The third method is InsertTransactions, which takes the following input parameters:

sellerOfTheGood,
approvalStatus,
typeOfGood,
payingDept,
usingDept,
dateOfOrder,
dateOfBuying,
dateOfApproval,
unitPrice,
totalPrice,
amountOfGood.

This method can be used by the employees of the purchasing department and the employees of the other departments, who are registered to the system.

The fourth method is SearchDatabase .It takes the following parameters: sellerOfGood, approvalStatus, typeOfGood, payingDept, usingDept, dateOfOrder, dateOfBuying, dateOfApproval, lowerPrice, upperPrice. The method generates a query according to the fields of which values are not null and returns the result of the execution of the query generated.

The fifth method is ViewPending .It returns the list of transactions of which approvalStatus value is ”not approved”. Since the class operates on the transactionTable, the name of the table in the database is stored in the class attribute transactionTableName.

2.4.4 ADMIN Class

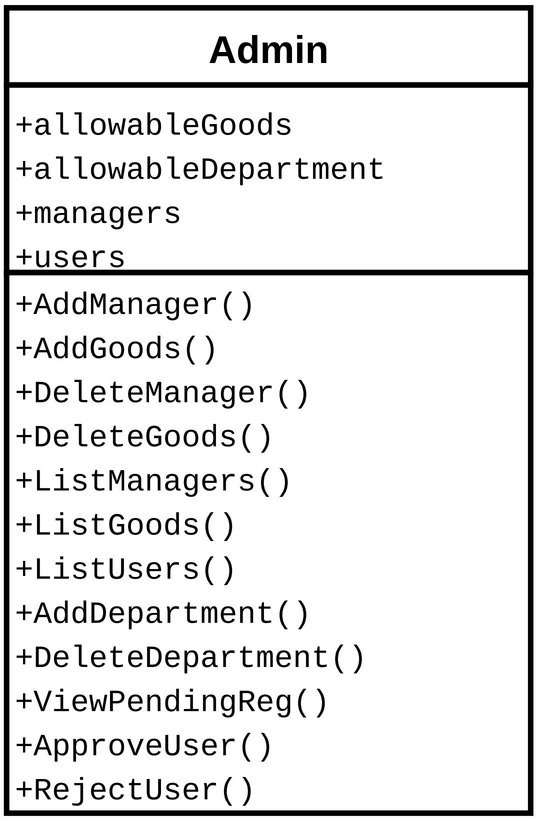

Figure 2.6: ADMIN Class

ADMIN class is the class that contains the methods in order to fulfill the duties of the administrator. The methods are user interface according to the rights of the user. In this case the rights of the user must be administrator rights. The class holds the names of the tables for allowable values,managers,users etc. for the ease of processing.

The method "AddManager()" does what it promises in the name. It gets the department name and userID and adds the user with the ID as the departments manager. The rights of this user will be determined according to this for later logins.

The method "AddGoods()" inserts an allowable value of good to the allowable values table. The method gets the name of the allowable value and inserts it into the table. This way the follow up system will not give an error message when trying to make a transaction relating to that good.

The method "DeleteManager()" does the opposite operation of the "addManager()" method. It gets the ID of the manager of a department and removes the user as a manager.

The method "DeleteGoods()" does the opposite of the method "addGoods()". It gets the name of the good and remove it from the allowable goods table. The transactions regarding this good will be deleted.

The method "ListManagers()" lists the managers of departments in the system. The information is taken from the managers table.

The method "ListGoods()" lists the allowable values of goods in the system. The information is taken from the allowable goods table.

The method "ListUsers()" lists the users in the system and it uses the table of users from the database.

The method "AddDepartment()" adds a new department to the allowable values table for departments in the database. Therefore when a transaction is done for that department, the system will not give an error. This method is mostly used for new departments that are added to the university.

The method "DeleteDepartment()" removes a department that has been closed. The department is deleted from the related table in the database. The users related to this department might be deleted or the department property of these users might be set to null.

The method "ViewPendingReg()" gives the list of the users who registered and waits for the approval of their registration. The information is taken from the related table in the database.

The method "ApproveUser()" gets the ID of the pending registration and adds the user into the users table. This is a crucial duty of the administrator.

The method "RejectUser()" does the opposite of the method above. After the registration is rejected, the pending registration is removed from the pending registrations table.

3

User Interfaces

The purpose of the user interface is to make the Stock Follow Up System user-friendly and provide the ease of use. In order to do that, there is a designed user interface for the user-system interaction.

The methods that are employed by the user interface are as the following.

For the user class: ApproveTransaction, ViewAllTransactions, InsertTransaction, ViewPending and SearchDatabase

For the admin class: AddManager, AddGoods, DeleteManager, ArrangeValues, ApproveUser, ViewPending and DeclineUser.

For the register class: Register.

The details of these methods are explained in the class description section.

Index