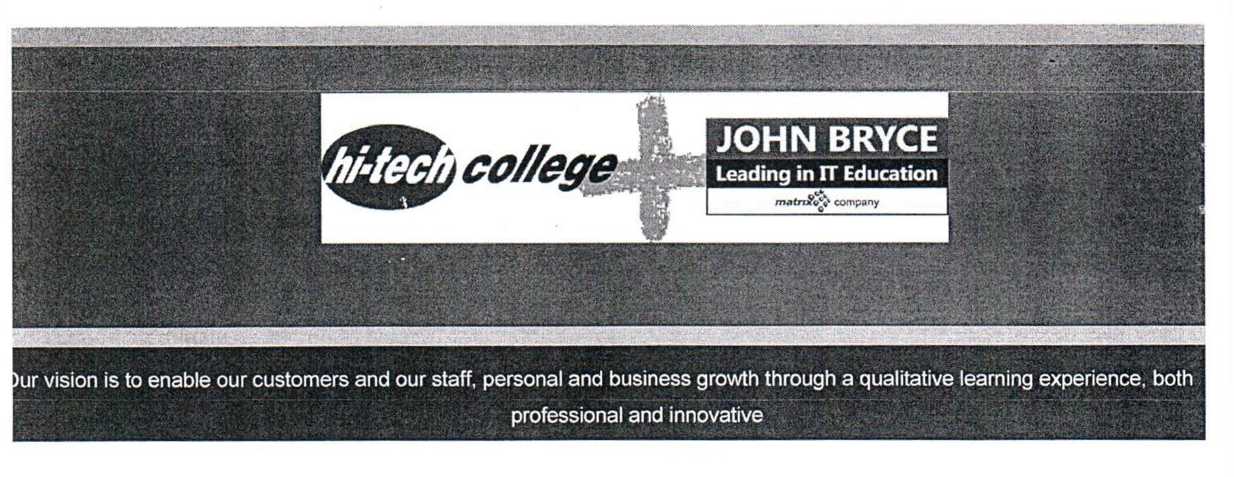
Java & JEE Track On-going Project

**John Bryce Hi-Tech College LTD and Hi-Tech Mediatech College (2002) LTD**

**are part the Matrix IT LTD group**



**Java &JEE Track ־ On-going Project**

Written by Rony Keren

©All rights reserved to John Bryce Hi-Tech College from Matrix group



Java & JEE Track - On-going Project

MBank On-line System

Characterization Document and Phases Instructions

version 5.0

Written by  
Rony Keren  
Java & InternetTeam  
1



Content

[Introduction 3](#bookmark2)

[Description 4](#bookmark3)

[Goals 6](#bookmark5)

[Technologies, Environments and Tools 6](#bookmark6)

[Basic Assumptions 7](#bookmark7)

[System Tables 8](#bookmark9)

[Activity List 10](#bookmark10)

[Activities in Details 10](#bookmark11)

[System Properties Complete List 15](#bookmark15)

[Exception Handling 15](#bookmark16)

[Background Working Thread 15](#bookmark17)

[Architecture & Application Design 16](#bookmark19)

[Implementation 17](#bookmark22)

Phase 1 18

Phase 2 22

Phase 3 23

Phase 4 25

2



Introduction

Dear student,

This on-going project will escort you along the course starting from Basic Java phase, through Desktop and web Ul, ending with business components. The project enforces the usage of most important Java & JEE APIs.

The goal of this project is to allow you practice and gain confidence with the materials delivered in class and in addition to the focused exercises done during lectures. In addition to the fact you get experience as you learn, you also face a "big application" and implement it by working with a team (1-3 people).

As mentioned, the project is made out of several phases. Each phase comes as an extension to the other, and therefore you should stick to the instructions and implement the project according to the specified design. Bonuses are welcomed as long as they don't break the design.

The project phases are synchronized with the course modules. Each phase has a dedicated amount of time to prepare and assign. It is important to stick to the time frame so project phases will be assigned on time rather than get delayed and expand to the next module. This is how you'll keep yourself ready for the new module.

Each phase will be presented and checked by the instructor of the relevant module. The instructor will also be there to support you and solve problems preventing you from progressing while working in the project - for that we have several dedicated meetings which are detailed as part of the course schedule.

We know and believe that this project will assist you to absorb the materials and gain confidence. Both are critical when you are on your new way in this industry. The project is an important added value provided by this track.

Project description, design and phases are all detailed in this document.

Good luck !

Java & JEE Track Team

3

Description



MBank Company provides banking services which are free of brunches. This means that clients get their services through the internet.

In this project you will build the MBank on-iine system from scratch. You'il start with the system core, then you'll build a desktop administration console, add a web module to create the MBank web site and eventually -you'll add an advanced bank logging services.

MBank client services include the following:

* Client can create one current account
* Clients with current accounts can create long and short-term deposits
* Client can deposit and withdraw money into and from their short-term deposit
* Client can pre-open a long-term deposits before its ending date
* Client can view his details

MBank management extension should provide the following:

* Client management
* Client deposit management
* Reports
* Bank activity management

Currently, the system is implemented with clients having a dedicated desktop application installed on their PCs, performing transaction via internet connection. Administrators currently work directly on the DB running pre-defined procedures.

The goal is to provide a limited desktop application for administrator system management and to come up with a new web site that will replace the client desktop applications forcing the bank IT team to install on each client PC and provide support services.

MBank company manager expectations from the new system: CTO - Chief of Technologies Officer

Today, our system is built in a way that cannot be extended. We want to use web protocols instead of proprietary binary protocols so changes and updates can be applied easily. The technologies we currently use are old and the fact that we have to install our client applications physically on client PC is a major challenge. We spend a lot of efforts on

4

supporting clients who face problems during installations and while working with the



application.

CFO - Chief Financial Officer

Lately the number of clients increased dramatically. Therefore, the way we provide services today, which is far from being efficient and lightweight - become a heavy expense. We must iower costs by getting rid of the client support and by working with containers we can purchase with support instead of maintain the entire system independently.

CSO - Chief Sales Officer

We find it very hard to expose our clients to news, notifications & new services we offer.

This is since there is no single point which is constantly viewed by our clients. Each uses its own application, seeing his specific data only - with no marketing infrastructure once or ever. We believe and expect that the new web site will change that and provide a powerful marketing area since all our clients will use it to login.

5



Goals

General goals

* Extra exercising in addition to the focused class and homework exercises
* Allow students to experience a project, with given instructions and time-frame
* Allow students to experience team work, with roles and task assigning
* Ability and confidence to face massive development and use unfamiliar APIs and features
* Experience real life challenges like migrating from one platform to another Project goats
* Simplifying the process of adding new clients
* Creating a modern web based solution
* Make it easier to promote commercial activity through the site
* Use a central system with high connectivity capabilities for remote clients
* Lowering costs by providing services more efficiently and easily

Technologies, environments and tools

* Java SDK - at least J2SE 5.0
* IDE - Eclipse for JEE
* Servers in use -Tomcat for web phase, JBoss for the business components phase
* Database - default is Access that comes as part of your Office installation, but any other DB will do (MySQL, Derbi, Oracle....). For last phase, Hypersonic is used.
* Technologies surly involved in this project;

o J2SE - Collections, Exceptions, Threads, JDBC, Swing o Web - HTML, Jscript, Servlets, JSP, XML, CSS o Business - EJB, JNDI, JMS, JPA, JTA

6



Basic Assumptions

The application will consist of two channels:

Administrative channel - A desktop application based on Swing technology. This application will be installed on each authorized administrator sttion to manage MBank On-line System.

Client channel - A web site that allow all users to view MBank commercial page(s). Registered users can login and maintain their account and deposits.

A note regarding response time and user friendly:

The system must provide real-time data. Data that the system already has - shouldn't be requested from client again. Transactions and interactions with clients must be as short and effective as possible.

Ul must be clear and friendly. Assume that users are not familiar with computers and computing - you must keep this in mind especially when creating Uls. Illegal actions must be prevented. Messages client should get must be notified in a clear way.

7

System Tables



There are 5 tables used by the MBank old system. There are no roles and links between the tables - all is done by the application. The new system will use the same DB schema detailed here:

Table: Clients - holds all MBank client data

* client\_id - long - client unique ID, used as PK as well
* client\_name - String - username
* password - String
* type - String - Enum with the following options: REGULAR, GOLD, PLATINUM (different level of clients get different rates for commission and interests - later)
* address
* email
* phone
* comment

Table: Accounts - holds all client accounts data

* account\_id - long - account unique ID, used as PK as well
* clientjd - long - links between the account and its owner client - each client must have only one account
* balance - double - the current amount of money in the account
* credit-limit - double - the amount of money used as credit (client can never have an amount lower than credit-limit in his account)
* comment

Table: Deposits - holds all client deposits data

* deposit\_id - long - deposit unique ID, used as PK as well
* clientjd - long - links between the deposit and its owner client - each client can have multiple deposits
* balance - double - the current amount of money in the deposit
* type - String - Enum with the following options: SHORT, LONG (clients cannot close short-deposits before their closing date - later)
* estimated balance - long - estimated balance on the day the deposit will end (this is calculated according to the bank current interest rates - later)
* opening\_date - date
* closing\_date - date

8

Table: Activity - this table holds all the activity history data including commission rates for these activities

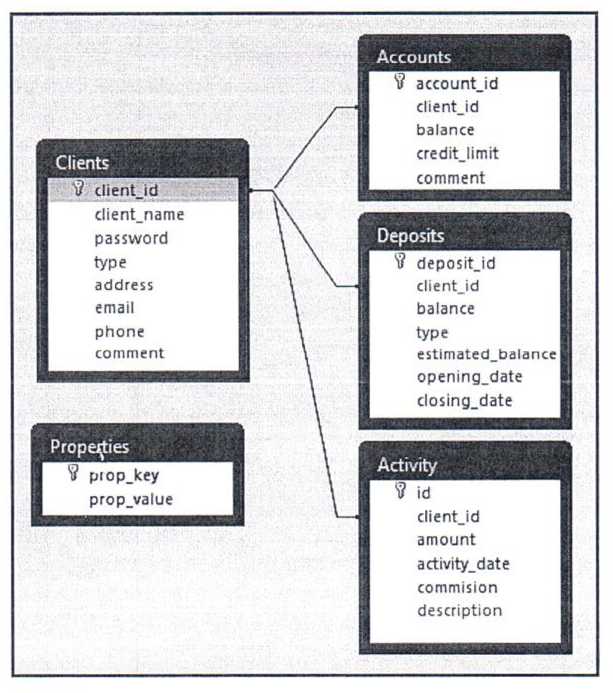


* id - long - used as PK
* client\_id - long - links between the action and its owner client (client activities involving bank commission are detailed later)
* amount - double - the amount of money issued in this transaction
* activity\_date - date - date of the activity
* commission - double - holds the commission rate charged by the bank for this activity (details about commission calculation are given later)
* description - string - a description of the action itself (withdraw, deposit, account closing, deposit ending, deposit pre-opening... all these actions will be detailed later)

Table: Properties - this table holds all system properties. Properties values can be edited by system administrators

* prop\_key - string - holds unique property name, used as PK as well
* prop\_value - string - holds the value of the property

DB Schema Example:



9



Activities List

Here is a complete list of actions that can be done from clients and administrators and must be supported by the system. Activities that are done automatically are implemented as background task (thread). Detailed information about each will be given later

|  |  |  |  |
| --- | --- | --- | --- |
|  | Activity | Done By | Description |
| 1 | Add new client | admin | Adding new client to MBank system |
| 2 | Update client details | admin & client | Updating existing client details |
| 3 | Remove client | admin | Deleting clients with his account & deposits |
| 4 | Create new account | admin | Done when adding new client - & only once |
| 5 | Remove account | admin | Done when deleting new client - & only once |
| 6 | Withdraw From Account | client | Decreasing account balance[[1]](#footnote-2) |
| 7 | Deposit To Account | client | Increasing account balance\* |
| 8 | Create new deposit | client | Create new deposit |
| 9 | Close deposit | automatic | Money is transferred to clients account |
| 10 | Pre-Open deposit | client | Money is transferred after charging a commission\* |
| 11 | View client details | admin & client | Query |
| 12 | View all clients details | admin | Report |
| 13 | View account details | admin & client | Report/Query |
| 14 | View all account details | admin | Report |
| 15 | View client deposits | admin & client | Report |
| 16 | View all deposits | admin | Report |
| 17 | View client activities | admin & client | Report |
| 18 | View all activities | admin | Report |
| 19 | Update system property | admin | Updating values of system properties |
| 20 | View system property | admin & client | Query |

\* These actions affect the activity table and must be documented in it. All of them effects bank balance since a commission is charged for them

Activities in Details

This section details each and one of the activities and allows you to understand how to implement business logic:

1. Add new client

This operation is done by administrators only. When creating new client, a new account must be also created for it. Clients with no account are illegal in the system. When creating new client, it must deposit some amount of money in his new account. That amount of money determines the client type. System properties provide information about how much money is counted as REGULAR client and also for GOLD and PLATINUM. Here are the relevant system properties for that:



1. Update client details

Clients may update personal data that includes: address, email & phone. Administrators can update the client type (REGULAR, GOLD and PLATINUM) and comment.

1. Remove client

Client removal forces the removal of his deposits and account. This operation is done by administrator. If deposits are being stopped before time (pre-opened), which might be the case when client with deposits wants to un-register from MBank system, a commission for deposit pre-opening will be charged for each deposit, before removing it. The commission rate for pre-opening deposit before its time is one of the system properties:

* regular\_deposit\_commission = 1.5% of the deposit balance
* gold\_deposit\_commission = 1.0% of the deposit balance
* platinum\_deposit\_commission = 0.5% of the deposit balance

1. Create new account

This operation is done by administrators as part of the adding new client process. Each client has only one account. When creating new account, the amount of money the client deposits will set both client and account types. Regular client have special commission rates and credits, so does Gold and Platinum clients. Each client account has a credit-limit. Client cannot get to an account balance lower than credit-limit. The credit-limit of the new account will be set according to the following system properties for each client type:

* regular\_credit\_limit=10000$
* gold\_credit\_limit=100000$
* platinum\_credit\_limit=unlimited

1. Remove account

This operation is done by administrators as part of client removal process. This is done after removing all existing client deposits (if any). In case of negative balance, which means that the client owe money to the bank, the money will be charged immediately. Imagine that the client must give cash money to pay the bill and close the account. This means that Activities table is reported due to bank income (as a commission). Therefore, you must insert a new row to the activity table with the following data, just before removing the account:

* client\_id - the id of the client that is about to be removed
* amount - negative balance (current account value)
* activity\_date - today
* commission — the amount of money charged for negative balance

11

• description - "Commission charged due to negative balance account on client removal"



In case of positive balance account, this part is not required - the client doesn't owe anything to the bank so there is no income as part of the removal process and no effect on Activities table.

1. Withdraw From Account

This action occurs when client wants to withdraw money from his account. Withdraw is allowed as long as account balance is greater than credit limit. The bank charges for all withdraw & deposit actions according to the bank commission rate property:

• commission\_rate = 0.5$

Commission is charged from client account.

1. Deposit To Account

This action occurs when client wants to deposit money in his account. The bank charges for all withdraw & deposit actions according to the bank commission rate property. Commission is charged from client account.

1. Create New Deposit

Client may create as many deposits as it likes. The deposit interest is calculated according to the client type daily interest rate system property. Since MBank allows constant interest deposits only, the estimated amount of the deposit can be calculated by the time it is created. Here are the 3 daily interest rates system properties for each client type:

* regular\_daily\_interest=5/365 (0.14~) - means 5% interest per year
* gold\_daily\_interest=7/365 (0.19~) - means 7% interest per year
* platinum\_daily\_interest=8/365 (0.21~) - means 8% interest per year

Another important character of deposit - is the deposit type. Deposit type can be:

* Short - short term deposit, 1 year maximum. Cannot be pre-opened by clients. Clients are forced to wait to its closing date in order to get their money.
* Long - long term deposit, up to 40 years. Can be pre-opened by clients.

12



1. Close Deposit

This operation is done in the background by the daily thread that maintains deposits (will be detailed later). When closing date arrives, the deposit money must be transferred to the client account automatically and the deposit must be silently removed.

1. Pre-Open Deposit

Pre-Open means that client can withdraw before deposit closing date. In this case the bank will charge a pre-open fee as part of the deposit removal process, pre open fee is also a system property:

• pre\_open\_fee=1% of the deposit balance

Note: only long term deposits support close operation. Consider inheritance (LongAccount extends ShortAccount)

1. View Client Details

Both client and administrator can view client details. Administrator may choose different clients

1. View All Clients Details Admin can get a full client report
2. View Account Details

Both client and administrator can view client account details. Administrator may choose different client accounts

1. View All Accounts Details Admin can get a full account report
2. View Client Deposits

Both client and administrator can view client deposits details. Administrator may choose different client deposits

13



1. View All Deposits Details Admin can get a full deposit report
2. View Client Activities

Both client and administrator can view client activities details. Administrator may choose different client activities.

1. View All Activities Details Admin can get a full activities report
2. Update System Property

Admin can update any system property.

1. View System Property

Both admin and client can get the value of a specified property

14



System Properties Complete List

|  |  |  |  |
| --- | --- | --- | --- |
|  | Property name | Property  Value | Description |
| 1 | regular\_deposit\_rate | 10000$ | Regular new client deposit rate |
| 2 | gold\_deposit\_rate | 100000$ | Gold new client deposit rate |
| 3 | platinum\_deposit\_rate | 1000000$ | Platinum new client deposit rate |
| 4 | regular\_deposit\_commission | 1.5% | Regular commission rate for deposit opening |
| 5 | gold\_deposit\_commission | 1.0% | Gold commission rate for deposit opening |
| 6 | platinum\_deposit\_commission | 0.5% | Platinum commission rate for deposit opening |
| 7 | regular\_credit\_limit | 100000$ | Regular account overdraft limit (credit limit) |
| 8 | gold\_credit\_limit | 1000000$ | Gold account overdraft limit (credit limit) |
| 9 | platinum\_credit\_limit | unlimited | Platinum account overdraft limit (credit limit) |
| 10 | commission\_rate | 0.5$ | Commission rate for all withdraws & deposits |
| 11 | regular\_daily\_interest | 5/365 | Regular daily percentage added deposit value |
| 12 | gold \_daily\_interest | 7/365 | Gold daily percentage added deposit value |
| 13 | platinum \_daily\_interest | 8/365 | Platinum daily percentage added deposit value |
| 14 | pre\_open\_fee | 1% | Commission rate for deposit pre-opening |
| 15 | admin\_username | system | Default username for all system administrators |
| 16 | admin\_password | admin | Default password for all system administrators |

Exception handling

In this system you might find 3 different types of exceptions:

1. Unchecked runtime exceptions - these must be prevented by validation or caught by code.
2. Checked exceptions - exceptions that might occur from underlying driver or sub-system must be caught and wrapped with an MBank compliant exception that provides a clear description and data regarding the problem.
3. Logical exceptions - MBank system has its additional logic and restrictions. Logical errors must be also wrapped with MBank exception. For example, withdraw must fail with MBank exception if balance will get lower than credit limit.

Note: printStackTrace() is great for debugging - but not for project assigning....

Background Working Thread

MBank system must keep a background working thread in order to maintain its deposits. This thread will perform a daily check to track deposits with closing dates set for today. These will be silently removed after transferring all the money to the owner client account.

15

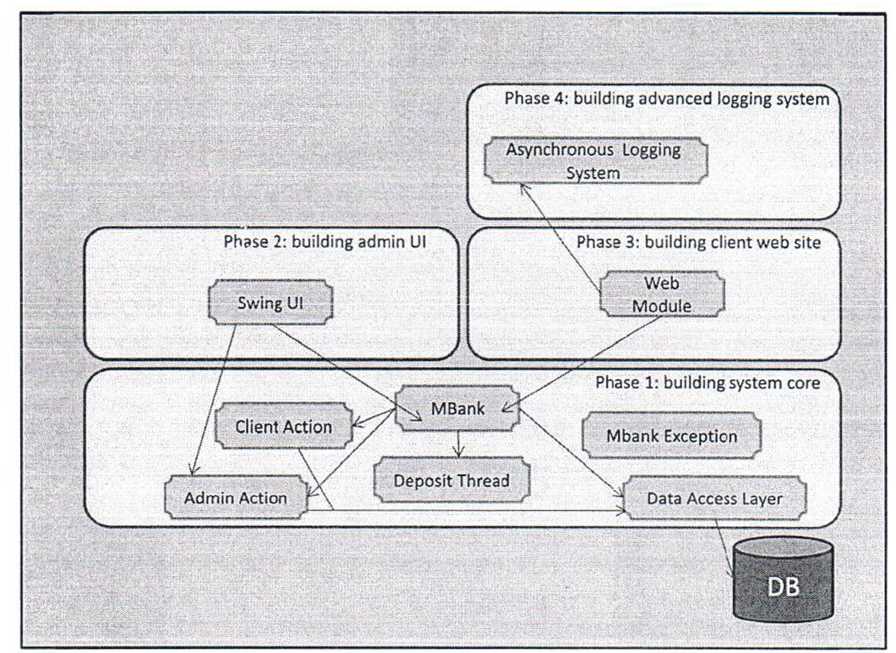


Architecture & Application Design

MBank system consists of several logical layers.

1. System core, built on 100% Java SE
2. Ul for client management that exposes some administrative capabilities
3. Web site for full client functionality
4. Asynchronous logging system

Each layer is a different phase of the project and it must be implemented in that order. Here is the overall look of the application layers:



16



Implementation

Students may develop the project independently or in groups of 2-3 members. Each team will choose one member to be the team leader. Team leaders are responsible for scheduling, giving tasks to each member equally, administrate code integration from different members and problem reporting.

Each phase has an assignment date. Do your best to assign your project in that day and stick to the time frame enforced during the track. Time management is very important. But more than that, the track is moving on. By the time you implement your project you learn new things in class. You must do your best to be ready with previous phase when the next phase is presented.

Good Luck !

17



Phase 1 - Building System Core

In this phase you will:

* Create a database access layer to work with a chosen DB
* Create and use system properties
* Implement administration actions
* Implement client actions
* Create a background deposit maintainer
* Build an MBank singleton to load the system
* Allow login for registered clients and administrators
* Define non-UI main application to test system core

Main system components

1. Database access layer

A layer isolates underlying DB from Java code. It is done by assigning objects (beans) + command to be translated in to SQL statements - and vice versa. This layer will be defined by interfaces and then will be implemented to work with DB (the interface layer provides future ability to extend storage types like files for example).

Each table should have dedicated DB managers, for example:

Client table will have the following classes:

* Client bean to be used as value object
* ClientManager interface that defines methods to insert, update, delete and query Client beans from and to a general storage
* ClientDBManager class that implements ClientManager interface to work with a DB and a DB connection

Note:

DB Connection should be used carefully. Eventually the system will work in a web multithreaded environment. Since connections are not thread safe we must make sure they are secured while handed to a client. In order to protect connections you can do one of the following:

* Create a static connection pool - Create a Set collection and populate it with a specific number of connections (you may define the default number of connections as system property). Any client which logs in or any thread that needs a connection - simply "borrows" it from the pool. When the client logs out and when the thread goes to sleep - the connection is returned to the pool. In that case you should

18

handle a situation in which all connections are consumed - client should be notified by connection exception.



• 2. Provide each client a new connection. Give the thread a new connection every

time it wakes up. Easier to implement but might be risky in an intensive environments (you shouldn't be bothered by that now...)

1. Action objects

Action objects define all system activities. Some activities are common to clients and administrators but most are not. Action super class may define all common operations and then be extended by ClientAction and AdminAction classes.

When client login through MBank - it should get a ClientAction with a working data access layer and with a dedicated connection. Same should happen with administrators getting AdminAction objects.

ActionObjects must provide an 'exit' or 'logout' operation to close/return connection before leaving the system.

1. MBank singleton

This is the system engine. Mbank is responsible for:

* Launching deposit maintenance thread
* Allow client to login with a given username and password and create a working ClientAction object in result.
* Allow administrators to login with a given username and password and create a working AdminAction object in result - admin username and password are defined in system properties (defaults: username="system", password="admin").
* In case of using a connection pool - MBank must create it on startup and destroy it on shut down.

1. Background Deposit Thread

This thread is started when system is launched. Deposit thread is a background task that tracks deposits with closing date set for today and closes it. Note that this thread sleeps most of the time and therefore must create/borrow a new connection every time it wakes up much like it must close/return it before going back to sleep.

19

Assignment



This phase requires no Ul based clients. Therefore, input should be done via command prompt. You may use java.util.Scanner. Classes must be arranged in appropriate packages.

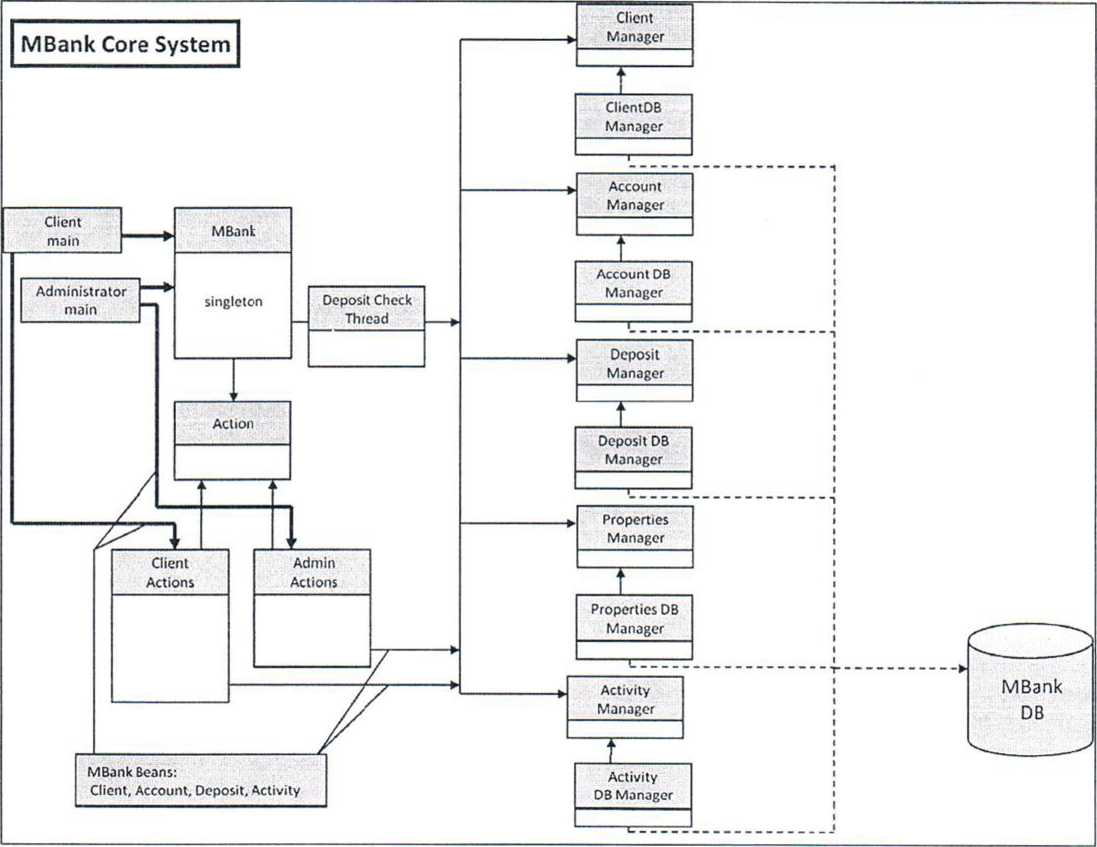
In this phase no user input validation is required. This is because the "real" clients of MBank will be Swing and Web based applications. The client application you provide in this phase are mainly for testing your system core.

Assignment files:

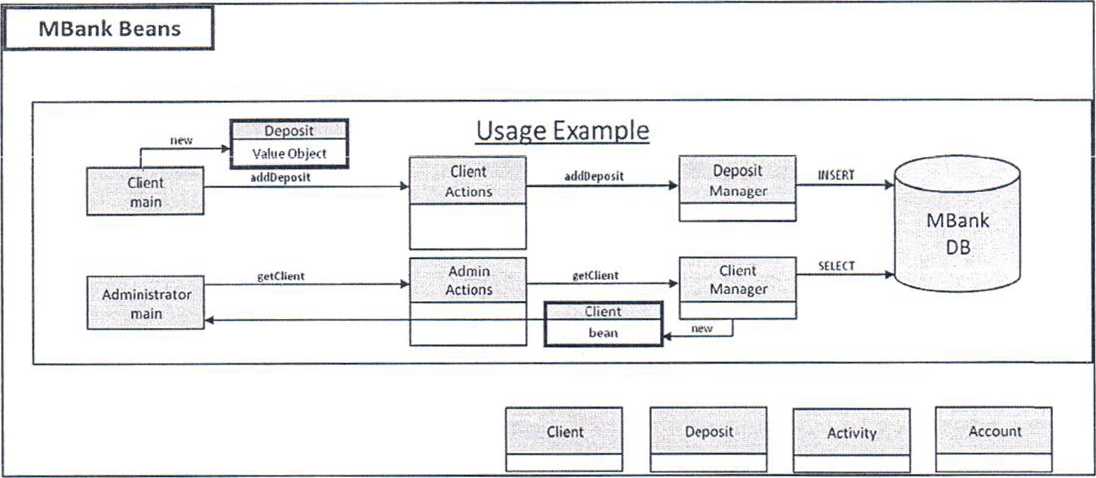
* Working eclipse project **י** DB files
* Readme document with details regarding users, db configurations, how to use, extras (if any) & important notes

20

System Diagram



System Beans Examples



Consider inheritance on Account (LongAccount that extends ShortAccount)

21

Phase 2 - Building Administration Desktop Application



In this phase you will:

* Create a desktop management application using Swing technology
* Expose some administrative capability via Ul

The administration application will allow the following:

* Login as an administrator
* View and change system properties values
* Add new client, with a new account
* View client details

Important notes:

* Ul must be user friendly
* All inputs must be validated
* All exceptions and notifications must be presented nicely to the client (consider working with JOptionPane)
* All lists (properties and clients list) must be presented in a JTable
* All listeners with significant amount of code should be separated from Ul code
* The application must allow the user to go back to main menu and also to log out
* All your Swing development will be packed in a separate package hierarchy
* All the data presented to the user must be 'real' and synchronized with the DB

22

Phase 3 - Creating MBank Web Site for Clients



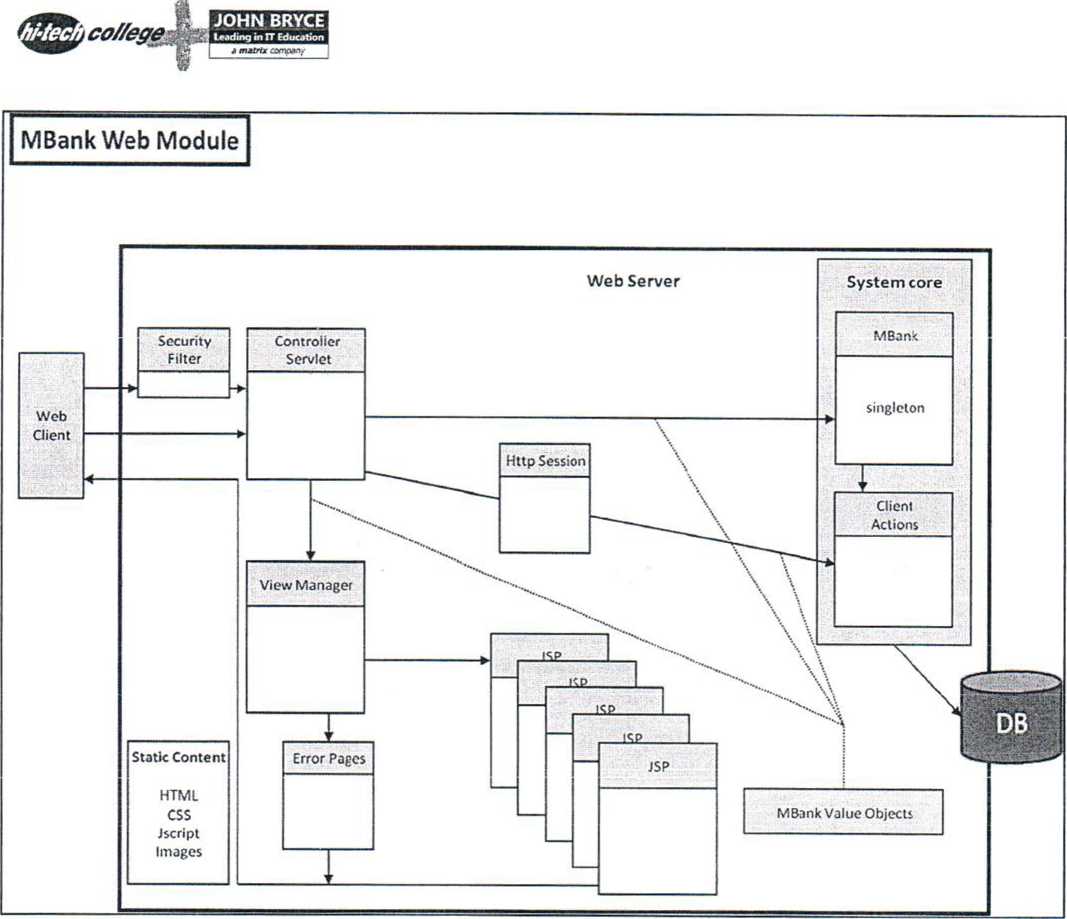
In this phase you will:

* Implement a web module using MVC model 2 architecture - Mbank web site
* Allow clients to login
* Expose all client activities are available through the web site (including deposits pre- opening)
* Allow clients to view personal data, account data and deposits List of main web site pages:
* Welcome page that provides some commercial information and a link to login page
* login page
* error page to present MBank errors to the client
* withdraw and deposit to & from account (this process must present the commission charged by the bank so client can view and approve before actually depositing or withdrawing)
* view client personal details
* update personal data (address, email & phone)
* view account details
* view all deposits
* create new deposit
* pre-open deposit (this process must present the commission charged by the bank so client can view and approve before actually pre-opening the deposit)
* view client activities
* view chosen system property value

Important notes:

* Ul must be user friendly. The site look must be consistent, using menu area, logo area and main page area
* All user form input must be validated with Jscript before submit
* All exception and notification must be presented nicely to the client. You may use alerts, DHTML and error pages
* All lists (properties and clients list) must be presented in a tabular view
* The web site must allow users to go back to main menu and also to log out
* All the data presented to the user must be 'real' and synchronized with the DB
* Links that point to static pages shouldn't go through a controller
* Verify that clients with no active sessions (hackers or clients with time-out sessions) will be redirected back to welcome page. This can be easily done via Servlet filter

23



**24**

Phase 4 - Creating MBank Asynchronous Log System



In this phase you will:

Implement an asynchronous logging system to log all web activity on site. The system will use JMS for asynchronous messaging, JPA and EJB3 technologies to both simplify the development process and also to face heavy loads since unlike the web site, this logging system will be used widely by the bank, even for internal purposes.

At the beginning you must migrate your Mbank web site from Tomcat to JBoss

Log Entity Bean

* Each instance is a Log content consist of:
* Id - unique value used as PK
* Log\_date
* Client\_id
* Description

Entities are created by the controller and passed to the business tier.

Log flow

* Controller servlet issues a Log message every time it is servicing. Log message is an Entity bean defined in a preset persistent unit
* The log goes through some synchronizer (Log Delegator) that hold an EJB stub (LogSLBean) - explained later
* The bean that receives a log message from the controller -generates a JMS Message containing log data and puts it in a JMS queue
* A message driven bean consumes log messages asynchronously from that queue and delegates it to another stateless bean for processing (StoreLogsSLBean)
* The stateless bean uses JPA to store the log message

Persistence unit

Since you're using JBoss as our application server, we can use its default setting as follows:

* Database - JB oss embedded Hypersonic
* Data source JNDI name - "java:/DefaultDS"

Messaging services

You may use JBoss default setting for JMS as well:

* Destination JNDI Name: "queue/testQueue"
* JMS Connection Factory JNDI Name: "ConnectionFactory"

25



Regarding Servlet synchronizer

Controller servlet experience multithreaded requests. If it holds a stub to a stateless EJB, this stub is a non thread-safe connection. Therefore, all logs must be synchronized before being sent to the business tier. The fact that stateless beans are pooled and the business tier handles each log message asynchronously helps in keeping this synchronization lock as short as possible. All servlet logs must go through the synchronizer which holds a stub to the bean. The synchronizer is referred as 'LogDelegator' and should be implemented as a regular Java class loaded by the Controller init() method.

26

1. regular\_deposit\_rate = 10000$ new account balance

   gold\_deposit\_rate = 100000$ new account balance

   platinum\_deposit\_rate = 1000000$ new account balance

   Each client must use a unique username-password combination.

   10 [↑](#footnote-ref-2)