

## References:

- [1]. Beecham B.J, Elements of Banking, method of making payments, pp 126
- [2] Dissanayake G.R Banks & Services in Sri Lanka, People's Bank, pp83
- [3] Gamage S.W, Business studies & New Trends, pp136
- [4] Henderson, M.J (1988), Banking Operations, Corporate Customers, pp97
- [5] Ian Sommerville (2004), Software Engineering 7<sup>th</sup> Edition, Chapter 4
- [6] Microsoft Corporation (2002) Programmer's Guide Microsoft Visual Basic
- [7] <http://en.wikipedia.org/wiki/MySQL>
- [8] [http://members.tripod.com/barhoush\\_2/overview.htm](http://members.tripod.com/barhoush_2/overview.htm)
- [9] [www.apache.org](http://www.apache.org)
- [10] [www.atlas.kennesaw.edu/~dbraun/csis4650/A&D/UML\\_tutorial/](http://www.atlas.kennesaw.edu/~dbraun/csis4650/A&D/UML_tutorial/)
- [11] [www.mysql.org](http://www.mysql.org)
- [12] [www.peoplesbank.lk/about](http://www.peoplesbank.lk/about)
- [13] [www.php.net](http://www.php.net)
- [14] [www.slideshare.net/ahirsiddharth/ch4-5742123](http://www.slideshare.net/ahirsiddharth/ch4-5742123)
- [15] [www.slideshare.net/kiran.chkumar/software-testing-techniques-507166](http://www.slideshare.net/kiran.chkumar/software-testing-techniques-507166)
- [16] [www.trainingetc.com/PDF/TE1802eval.pdf](http://www.trainingetc.com/PDF/TE1802eval.pdf)
- [17] [www.uml-forum.com/](http://www.uml-forum.com/)
- [18] [www.w3schools.com/](http://www.w3schools.com/)

## A1. USE CASES

### Head Office

#### A1.1 Access PB web site & Upload files

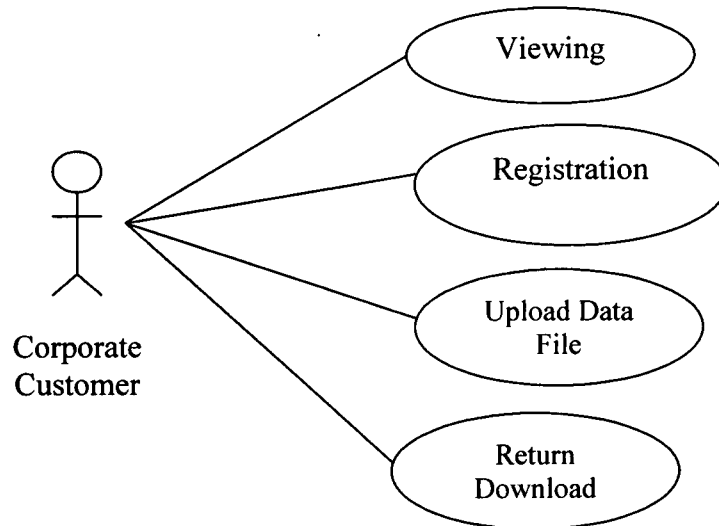


Figure A1.1 Access PB website and Upload Data  
University of Moratuwa, Sri Lanka.  
Electronic Theses & Dissertations  
[www.lib.mrt.ac.lk](http://www.lib.mrt.ac.lk)

#### Brief Description

For this use case initiated, the customer has to be connected to the internet & PB web page.

#### Initial Step-By-Step Description

1. Customer clicks the customers data transfer link on PB home page.
2. The system gives user name & password menu.
3. The user fills the user name & password.
4. System checks user name & password.
5. If login success system present next menu.
6. If login unsuccessful repeat step 2-4.

### A1.2 Corporate Customer logon to data transfer link

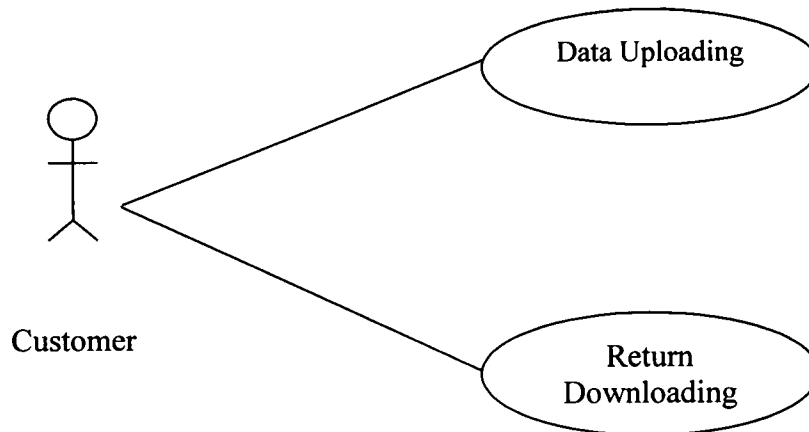


Figure A1.2 Corporate customer login

### A1.3 Data validation & uploading



Customer

Figure A1.3 Data Validation and Uploading

#### Brief Description

After validation process, response message will be displayed. Data can be uploaded after succeed the validation routine. While the uploaded process going on an email message will be send to the relevant branch for funds confirmation.

#### Initial Step-By-Step Description

1. Customer select data upload button
2. Customer select the source & click send file option
3. System gives upload completed message
4. System send email message to the branch for fund confirmation

## A1.4 Return Downloading

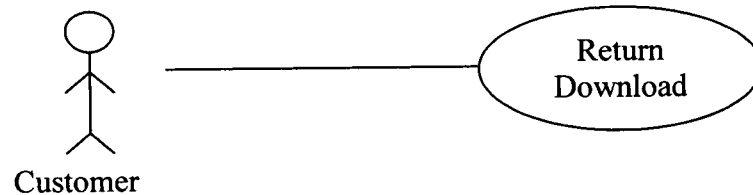


Figure A1.4 Return Downloading

### Brief Description

When the customer uploads the data file to PB HO, they will distribute to their branches or other banks. After comparing them with their existing account details, they will be returned back to us if any differences found. Then P/B H/O will be passed back to the original corporate customer.

### Initial Step-By-Step Description

1. Customer presses the Return downloading button.
2. The system displays the return detail accordance to the date
3. Customer clicks the particular date/s return transactions.
4. System download the return transactions
5. After completing system display the completing message.



University of Moratuwa, Sri Lanka.  
Electronic Theses & Dissertations  
www.lib.mrt.ac.lk

## At Head Office

### A1.5 Data Process

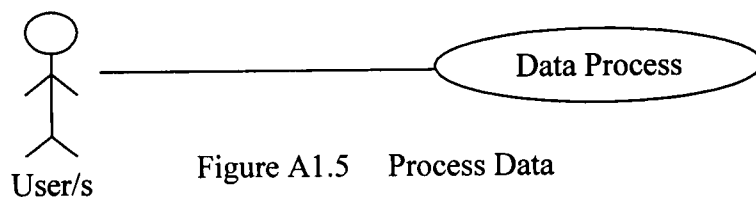


Figure A1.5 Process Data

### Initial Step-By-Step Description

1. User select the Data Process option
2. System displays the data process menu
3. User select the category (Customer data, IBT,OWD,IWD)
4. System get selected category
5. System start to process

## A1.6 Change paying date

### Brief description

Sometimes we have to change the paying dates appearing in the data which are already entered by the customers. User change the paying dates in needful after getting the confirmation by customer, without having Managers authorization users cannot change the paying date.

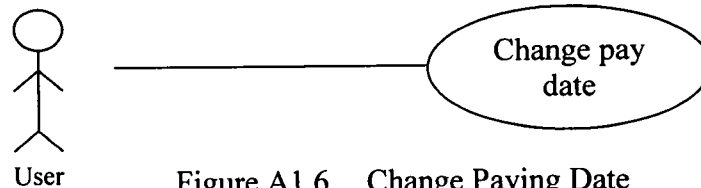


Figure A1.6 Change Paying Date

### Initial Step-By-Step Description

1. User select the change paying date option
2. System asks Customer account number, Current date, Date to be changed details.
3. User gives above details
4. System check the above details
5. System change required date if checking succeeded
6. System gives error message if the checking details unmatched

## A1.7 Batch deletion

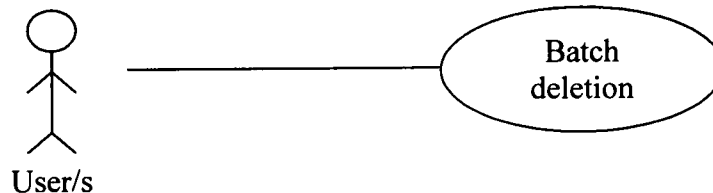


Figure A1.7 Batch Deletion

### Brief description

Sometimes the customers need to stop their salaries. At that time the system has the facility to stop that batch.

### Initial Step-By-Step Description

1. User selects the batch deletion option
2. System display the batch deletion menu
3. User enter the required details

4. System checks the details.
5. System ask Manager's approval
6. Manager gives approval
7. System deletes the batch required

### A1.8 Final file preparation

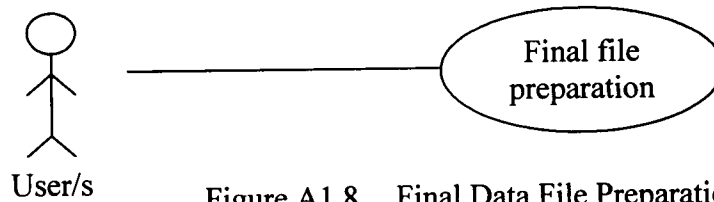


Figure A1.8 Final Data File Preparation

#### Initial Step-By-Step Description

1. User press the final file processing option
2. System asks Manager/Supervisor password
3. Manager/Supervisor gives the password
4. System check the password
5. If user name & password mismatch repeat 2-4
6. System displays the fund balancing report for confirmation
7. Supervisor gives confirmation message
8. System prepares & print the final (Debit account)entries file.
9. system prepares the files for SIBS,PABS and manual branches
10. User copies the SIBS, PABS transferring data files.

### A1.9 Data printing

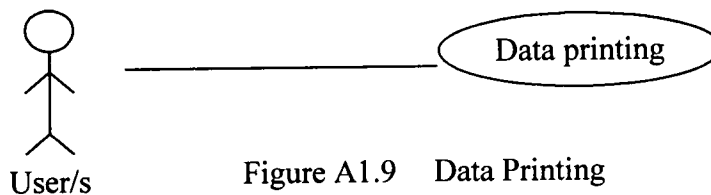


Figure A1.9 Data Printing

#### Initial Step-By-Step Description

1. User select data printing option
2. System gives printing menu
3. User Select printing option
4. System print summary reports for each branch
5. System print individual slip for manual branches
6. System print entries (Credit entries)
7. After printing completed user select backup option
8. System backup IBT file

### A1.10 Transaction Inquiry

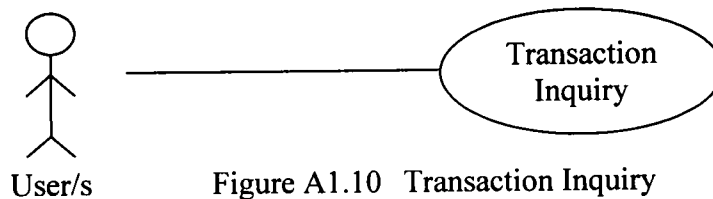


Figure A1.10 Transaction Inquiry

#### Initial Step-By-Step Description

1. User select the Transaction inquiry option
2. System jump to inquiry menu
3. User gives inquiry details
4. System display inquired details

### A1.11 Return Handle

#### Brief Description

Return details of SIBS branch's send by a common file & PABS branch's send as separate files. Return details of manually operated branches send using printed report. It will be entered at head office to the system.

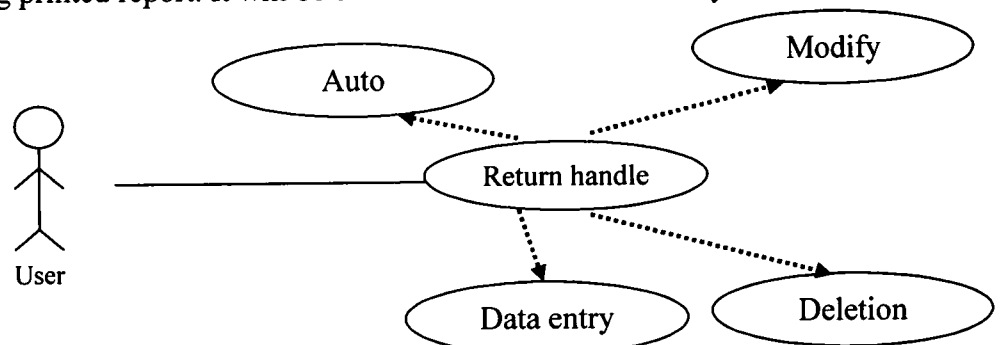


Figure A1.11 Return Handle

### Initial Step-By-Step Description

1. User select the return handle option
2. System gives return handle menu for manual branches
3. User can process enter, modify, delete return details
4. User get a report for manually entered return transactions
5. If return details correct Manager give the authorization to transfer.

### A1.12 Authorization

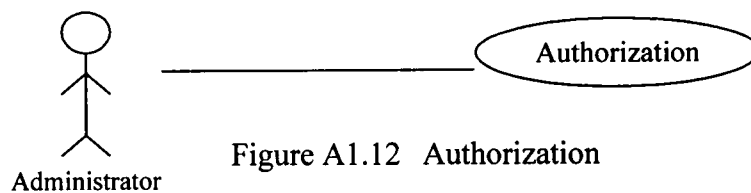


Figure A1.12 Authorization

### Initial Step-By-Step Description

1. System send the message for authorized the data
2. Manager/Administrator gives their user name & password
3. System check User name & password
4. If checking succeeded system approved the data
5. Otherwise repeat 2,3



## A2 Functional requirements

### A2.1 Data validation & uploading

Use Case	Data validation & uploading
Summary	This use case is used to validate the customer data & upload to PB head office.
Actor	Corporate Customer
Related use cases	
Pre condition	System should be available, Corporate customers should have log to the system
Description	<ol style="list-style-type: none"> <li>1. After success login customer click the Validation button</li> <li>2. The system give the option to select source drive</li> <li>3. Customer select drive</li> <li>4. System validate the file</li> <li>5. System give the validation response message</li> <li>6. If any validation exists system automatically return to home page</li> <li>7. If validation errors does not appear system enable upload file option</li> <li>8. Customer click upload button</li> <li>9. System send email to relevant branch for funds confirmation</li> <li>10. System gives upload completed message</li> </ol>
Alternatives	
Post condition	
Reference	

Figure A2.1 Data Validation & Uploading

## A2.2 Return download

Use Case	Return download
Summary	This use case is used by customer to download the return transactions upload by the PB head office
Actor	Corporate Customer
Related use cases	
Pre condition	System should be available, Corporate customers should have log to the system
Description	<ol style="list-style-type: none"> <li>1. After success login customer click the return download button</li> <li>2. System view the return transactions accordance to the date order</li> <li>3. Customer click the appropriate transactions to download</li> <li>4. System download the required transactions</li> <li>5. System gives the download complete message</li> </ol>
Alternatives	
Post condition	
Reference	

Figure A2.2 Return Download

### A2.3 Process Data

Use Case	Data Process
Summary	This use case is used by customer to process to customer data, IWD transactions
Actor	User/Operator
Related use cases	
Pre condition	System should be available, User/Operator should have log to the system
Description	<ol style="list-style-type: none"> <li>1. The user click on data process button</li> <li>2. System display options to customer data, IWD</li> <li>3. The user click the appropriate option</li> <li>4. System get the selected option</li> <li>5. If Option is Customer data               <ol style="list-style-type: none"> <li>5.1.1 User select validate the customer data file</li> <li>5.1.2 System validate &amp; give the validation results</li> <li>5.1.3 If any validation exists, inform to the customer &amp; do not go ahead</li> <li>5.1.4 If validation free user select file transfer option</li> <li>5.1.5 System transferred data to OWD &amp; IBT files</li> <li>5.1.6 User select the backup option</li> <li>5.1.7 System backup the file</li> </ol> </li> <li>6. If option is IWD data do the steps to 5.1.1 to 5.1.7</li> </ol>
Alternatives	
Post condition	
Reference	

Figure A2.3 Process Data

#### A2.4 Change paying date

Use Case	Change pay date
Summary	This use case is used by user/operator to change the payment date of the transactions
Actor	User/Operator, Manager/Supervisor, Administrator
Related use cases	Authorization
Pre condition	System should be available, User has to log on to the system
Description	<ol style="list-style-type: none"> <li>1. User select the change pay date option</li> <li>2. System display the change pay date menu</li> <li>3. User enter the required data</li> <li>4. System check the above entered data</li> <li>5. If does not match entered details, system display the error message &amp; exit from the menu</li> <li>6. If entered data match, system send a message to manager for approval.</li> <li>7. Manager give the approval</li> <li>8. System change the paying dates.</li> <li>9. If manager does not give the approval, system does not change the paying date</li> </ol>
Alternatives	
Post condition	
Reference	

Figure A2.4 Change Paying Date

### A2.5 Batch Deletion

Use Case	Batch Deletion
Summary	This use case is used by user/operator to delete the particular batch send by the customer and with their request
Actor	User/Operator, Manager/Supervisor, Administrator
Related use cases	Authorization
Pre condition	System should be available, User has to log on to the system & there should be a request by customer for stop their particular payment
Description	<ol style="list-style-type: none"> <li>1. User click on the batch deletion option</li> <li>2. System give batch deletion menu</li> <li>3. User enter the corporate customer's account number, sequence number, total amount</li> <li>4. System seek the detail &amp; display the summary</li> <li>5. System ask permission from manager to delete the batch</li> <li>6. Manager give the approval to delete the batch</li> <li>7. System delete the batch &amp; give deleted response</li> </ol>
Alternatives	
Post condition	
Reference	

Figure A2.5 Batch Deletion

## A2.6 Final Processing


Use Case	Final Preparation
Summary	This use case is used by user/operator to prepare the final file to transfer SIBS, PABS & manual branches and prepare debit entries
Actor	User/Operator, Manager/Supervisor, Administrator
Related use cases	Authorization
Pre condition	System should be available; User has to log on to the system. IWD file also processed before final file preparation. If in case IWD file does not exists for particular day, they can prepare final file after getting the confirmation by SLACH
Description	 <p>University of Moratuwa, Sri Lanka. Electronic Theses &amp; Dissertations www.lib.mrt.ac.lk</p> <ol style="list-style-type: none"> <li>1. User click on the final preparation option</li> <li>2. System give final preparation option</li> <li>3. User press the final file processing option</li> <li>4. System asks Manager/Supervisor password</li> <li>5. Manager/Supervisor gives the password</li> <li>6. If user name &amp; password mismatch repeat 2-4</li> <li>7. System displays the fund balancing report for confirmation</li> <li>8. Supervisor gives confirmation message</li> <li>9. System prepares &amp; print the final entries file (Debit account)</li> <li>10. system prepares the files for SIBS,PABS and manual branches</li> </ol> <p>Copy the SIBS, PABS transferring data files</p>
Alternatives	
Post condition	
Reference	

Figure A2.6 Final Processing

### A2.7 Transaction Inquiry

Use Case	Transaction Inquiry
Summary	This use case is used by user/operator to inquire the particular transaction sent to branches or other banks by PB head office
Actor	User/Operator
Related use cases	None
Pre condition	System should be available, User has to log on to the system
Description	<ol style="list-style-type: none"> <li>1. User click on the transaction inquiry option</li> <li>2. System give the transaction inquiry menu</li> <li>3. User enter the required details</li> <li>4. System search the above entered details</li> <li>5. System display sent date, return status (yes or no) Payment date etc</li> </ol>
Alternatives	
Post condition	
Reference	

Figure A2.7 Transaction Inquiry

### A2.8 Return handling


Use Case	Return handling
Summary	This use case is used by user/operator for enter the return transactions to the system. Returns from the SIBS or PABS branches can upload using auto option. For manually operated branches and In case of return files cannot read (SIBS or PABS) by the system use data entry, modify, and delete option.
Actor	User/Operator
Related use cases	None
Pre condition	System should be available; User has to log on to the system. Return transactions should be getting from branches.
 Description	<ol style="list-style-type: none"> <li>1. Use select the Return handle option</li> <li>2. System gives return handle menu</li> <li>3. User can process auto &amp; manual (enter, modify, delete) return transaction details</li> <li>4. After finishing preparing return transactions get the printout</li> <li>5. Manager manually check the transactions</li> <li>6. If entered return transactions are ok, can transfer to the main file</li> </ol>
Alternatives	
Post condition	
Reference	

Figure A2.8 Return Handling



### A2.9 User creation

Use Case	Users creation
Summary	This use case is used by Administrator or Manager to create, modify or delete the users or customer's login facility to the system.
Actor	Administrator or Manager
Related use cases	None
Pre condition	System should be available; Administrator or Manager has to log on to the system. To create customer's login facility the management of the PB has to approve their request.
Description	<ol style="list-style-type: none"> <li>1. Use select the user creation option</li> <li>2. System gives user creation menu</li> <li>3. User or Customer enter their details (user name, password etc)</li> <li>4. Administrator or Manager approved those details</li> <li>5. IF need any modification or deletion about the users or customer's, Manager or Administrator can do that</li> </ol>
Alternatives	
Post condition	
Reference	

Figure A2.9 User Creation

### A2.10 Authorization


Use Case	Authorization
Summary	This use case is used by Administrator or Manager to authorize data process, change pay date, batch deletion or final file preparation done by the users.
Actor	Administrator or Manager
Related use cases	Users
Pre condition	System should be available; Administrator or manager has to log on to the system. System sent message to the manager for authorize the data process, change pay date, batch deletion or final file preparation
Description	 <p>University of Moratuwa, Sri Lanka. Electronic Theses &amp; Dissertations www.lib.mrt.ac.lk</p> <ol style="list-style-type: none"> <li>1. System sent a message to the manager for authorize</li> <li>2. System ring an alarm when the authorization message sent</li> <li>3. Manager enter their authorization code</li> <li>4. System check the authorization code</li> <li>5. If code is ok system continue the process</li> <li>6. Otherwise system give an error message</li> </ol>
Alternatives	
Post condition	
Reference	

Figure A2.10 Authorization

### A3. Database Table's Structures

#### A3.1 Customer data file (Data)

Field Name	Type & Length	Description
Cbnkcd	nchar(04)	Customer's Bank code
Cbred	nchar(03)	Customer's Branch code
Ccacno	nchar(12)	Customer's Account number
Camt	numeric(15,2)	Transaction Amount
Ctrnno	nchar(03)	Transaction Number for particular branch
Ctrncd	nchar(02)	Transaction Code
Ccd	nchar(01)	Transaction type <b>Credit</b> or <b>Debit</b>
Cref	nchar(15)	Reference(Emp no, EPF no etc)
Cparti	nchar(15)	Particulars
Cpdte	datetime	Payment date ("YY/MM/DD")

Table A3.1 Customer Data File

#### A3.2 Data file (IBT / OWD)

Field Name	Type & Length	Description
DBnkcd	nchar(04)	Customer's Bank code
Dbred	nchar(03)	Customer's Branch code
Dcacno	nchar(12)	Customer's account number
Dcname	nchar(20)	Customer's Name
Dtrncd	nchar(02)	Transaction code
Dcd	nchar(01)	Transaction type <b>Credit</b> or <b>Debit</b>
Dtrnno	nchar(03)	Transaction number for particular branch
Damt	numeric(15,2)	Transaction amount
Dref	nchar(15)	Reference(Emp no, EPF no etc)
Dparti	nchar(15)	Particulars
Doacno	nchar(12)	Originating account number
Doacname	nchar(20)	Originating account name
Drtncd	nchar(02)	Return code (if a return record)
Dpdte	datetime	Payment date ("YY/MM/DD")

Table A3.2 Data File (IBT/OWD)

#### A3.3 Transaction Code file

Field Name	Type & Length	Description
Ttrncd	nchar(02)	Transaction Code
Ttrndesc	nchar(15)	Transaction Code description

Table A3.3 Transaction Code File

**A3.4 Return Codes file**

Field Name	Type & Length	Description
Rrtncd	nchar(02)	Return Code
Rrtndesc	nchar(15)	Return Code description

Table A3.4 Return Codes File

**A3.5 Corporate Customer's Detail file**

Field Name	Type & Length	Description
CDbnkcd	nchar(04)	Corp. Customer's Bank code
CDbrcd	nchar(03)	Corp. Customer's Branch code
CDacno	nchar(12)	Corp. Customer's Account number
CDacname	nchar(20)	Corp. Customer's account name
CDadd1	nchar(15)	Address Line1
CDadd2	nchar(15)	Address Line2
CDadd3	nchar(15)	Address Line3
CDtele	nchar(15)	Telephone Number
CDemail	nchar(15)	Email Address
CDcomrte	numeric(5,3)	Commission rate
CDjoinfee	numeric(10,2)	Joining Fee
CDjoindte	datetime	Joining Date
CDcurstat	nchar(01)	Current Status (Active or Suspend etc)

Table A3.5 Corporate Customer's Detail File

**A3.6 Reconciliation File**

Field Name	Type & Length	Description
REbnkcd	nchar(04)	Bank Code
REbrcd	nchar(03)	Branch Code
REcano	nchar(10)	Account Number
REtotamt	numeric(15,2)	Total Amount
REtotibt	nchar(06)	Total IBT
REtotowd	nchar(06)	Total OWD

Table A3.6 Reconciliation File

### A3.7 Return file

Field Name	Type & Length	Description
RTbnkcd	nchar(04)	Bank Code
RTbrcd	nchar(03)	Branch Code
RTacno	nchar(12)	Account number
RTcname	nchar(20)	Account Name
RTtrncd	nchar(03)	Transaction Code
RTrtncd	nchar(02)	Return Code
RTamt	numeric(15,2)	Return Amount
RTsndte	datetime	Send date to branch ("YY/MM/DD")
RTpmtdte	datetime	Payment date ("YY/MM/DD")
RTref	nchar(15)	Reference
RTparti	nchar(15)	Particulars

Table A3.7 Return File

### A3.8 Return Log File

Field Name	Type & Length	Description
RLdte	Datetime	Transaction Date
RLbnkcd	nchar(04)	Bank Code
RLbrcd	nchar(03)	Branch Code
RLtag	nchar()	

Table A3.8 Return Log File

## A4. Sequence Diagrams

### A4.1 Batch Deletion

#### Batch Deletion

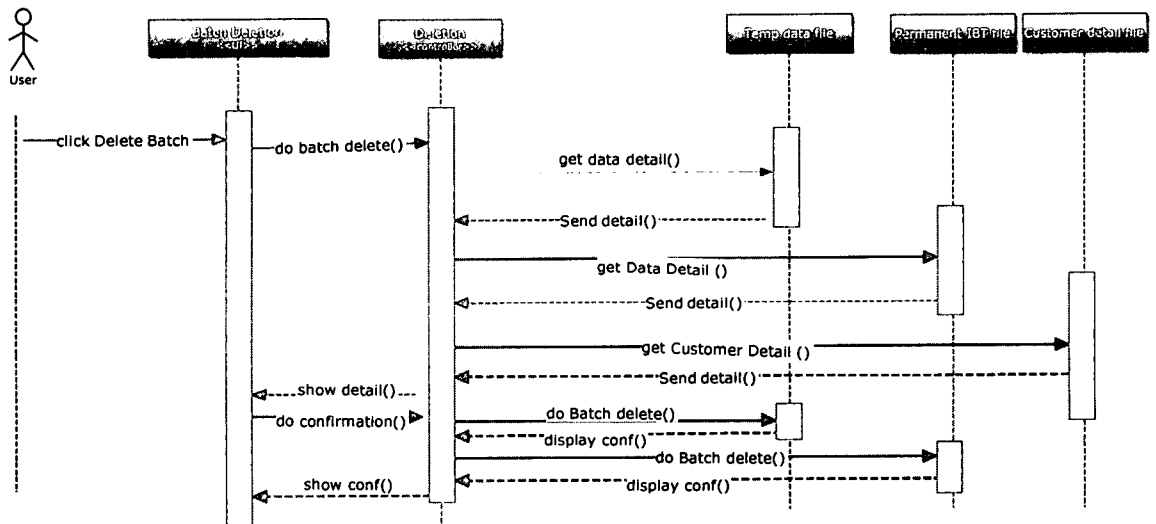


Figure A4.1 Batch Deletion

## A4.2 Change Pay date

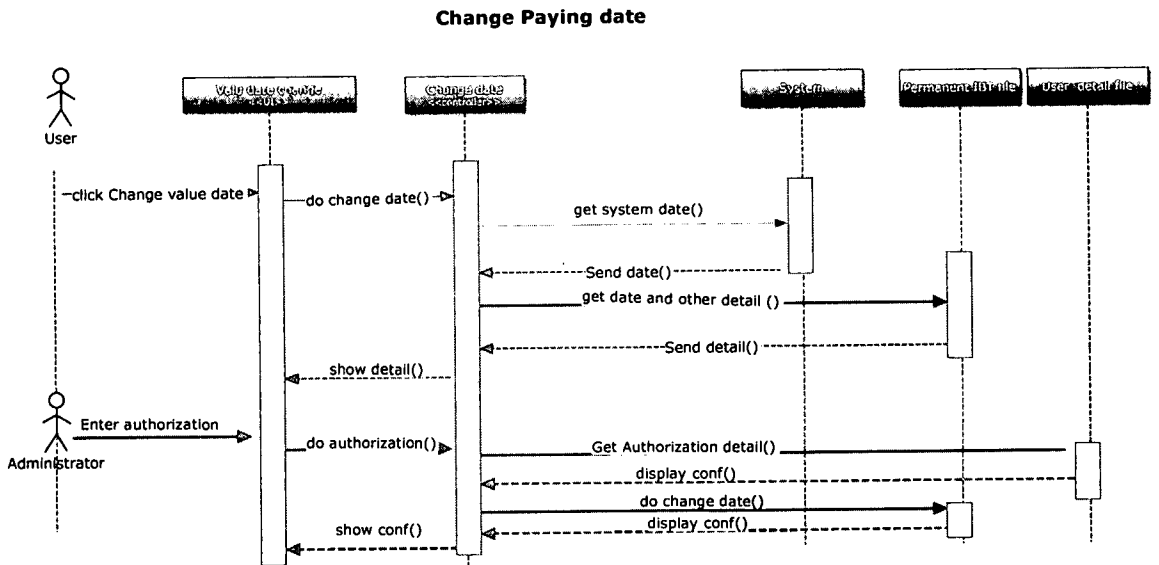


Figure A4.2 Change Pay Date

## A4.3 Check Files & Download

### Check files & Download

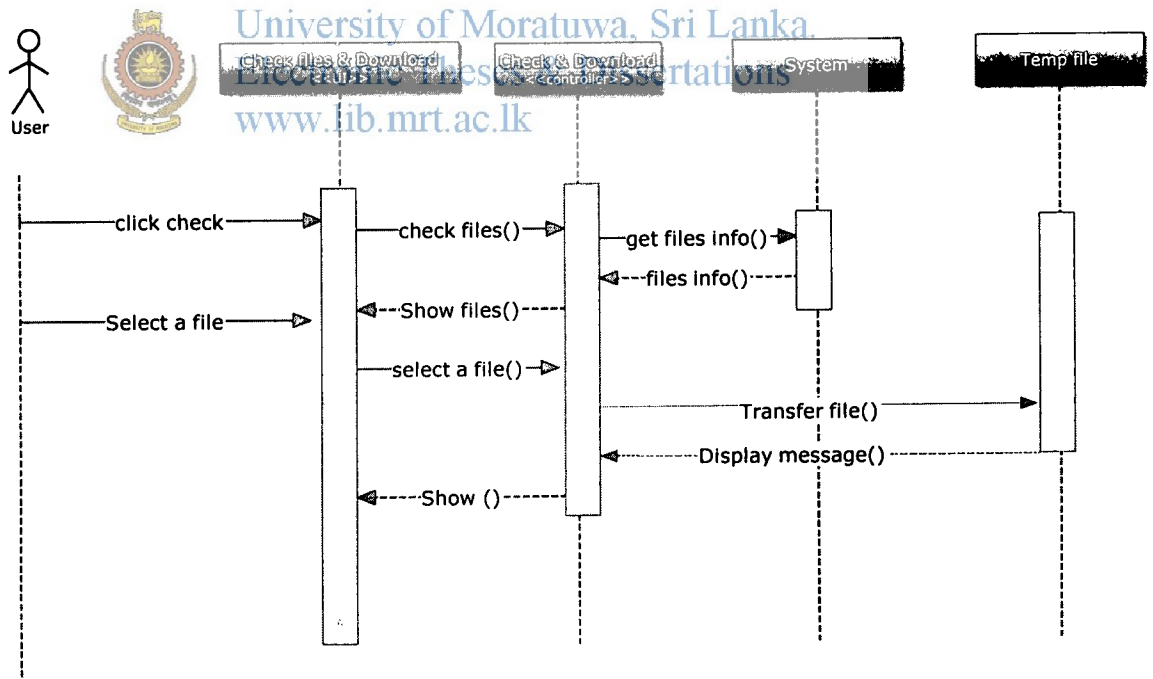


Figure A4.3 Check Files & Download

## A4.4 Data Deletion

### Data Deletion

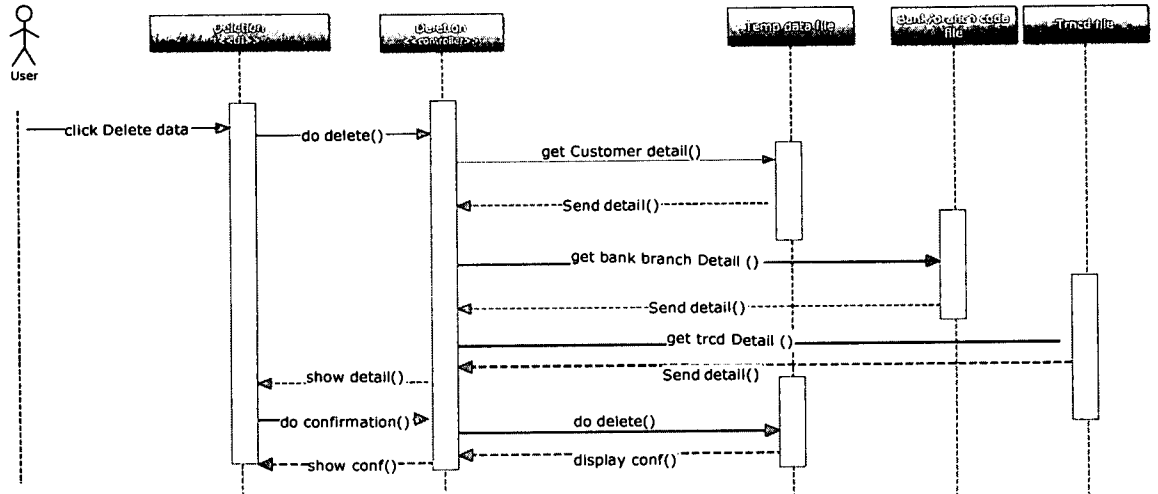


Figure A4.4 Data Deletion

## A4.5 File transfer from temporary file to permanent

### File transfer from temporary to permanent

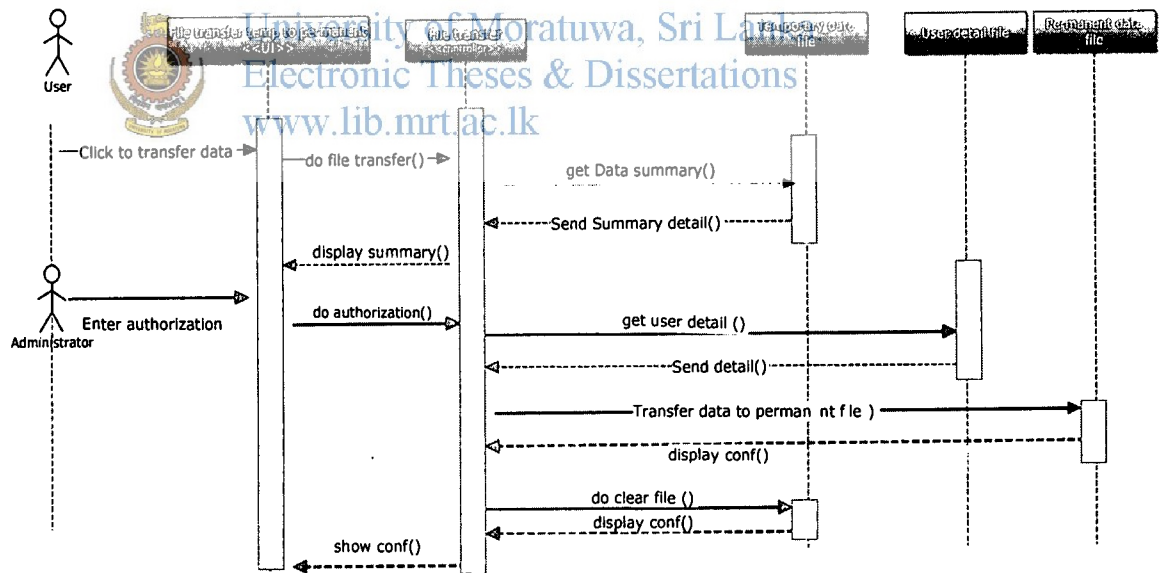


Figure A4.5 Final File transfer

## A4.6 Final Processing

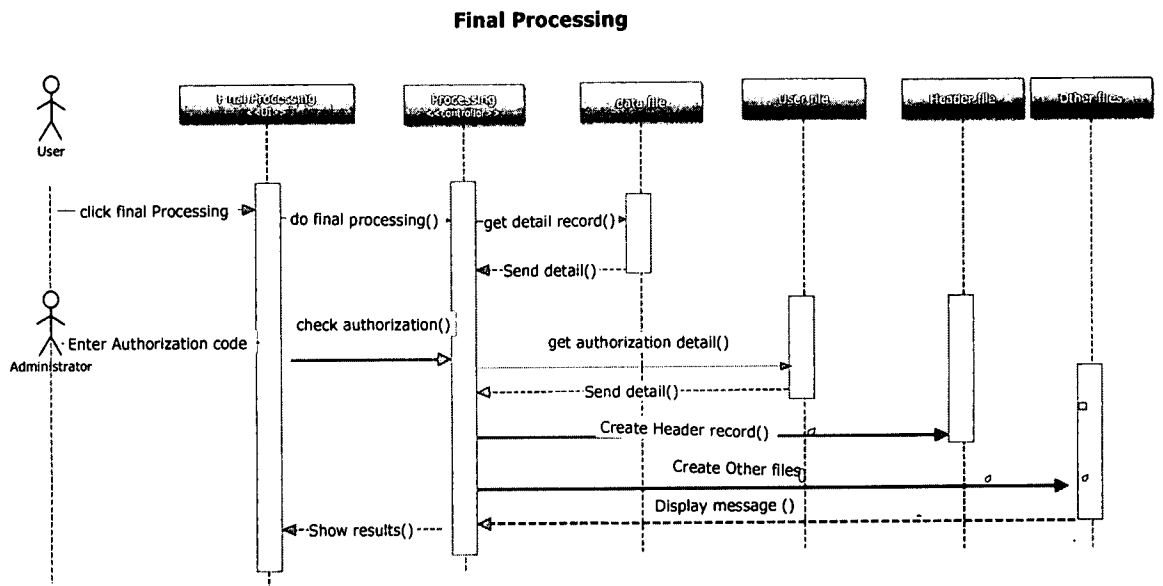


Figure A4.6 Final Processing

## A4.7 Printing Summaries /Vouchers/Entries

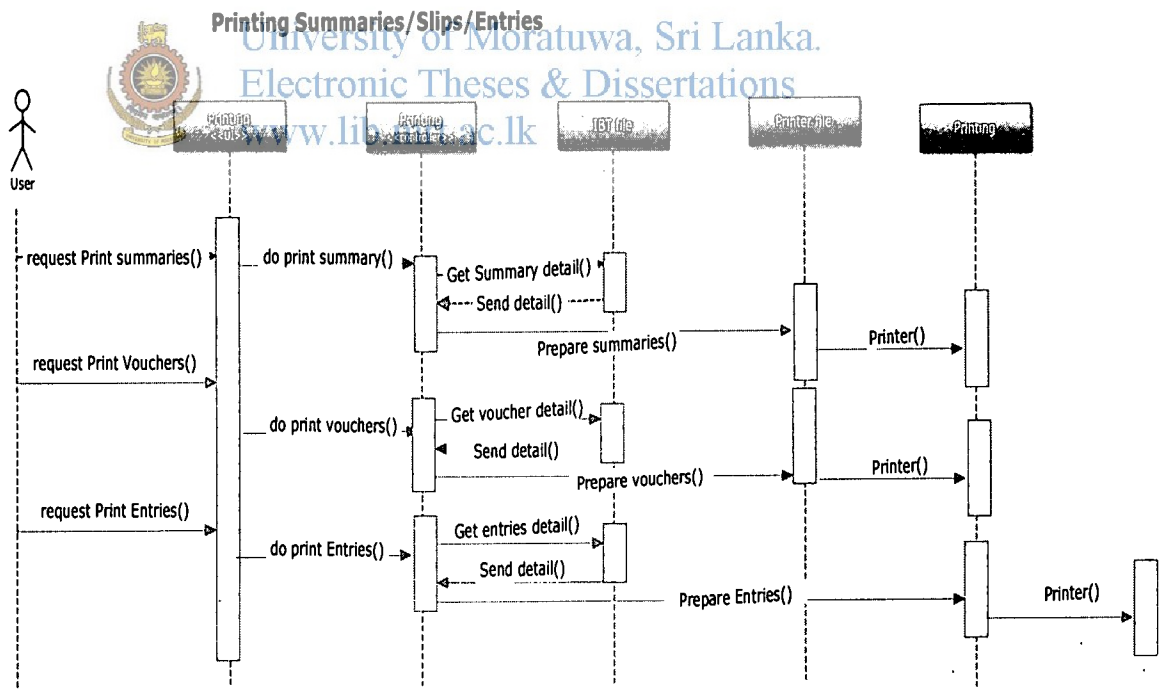


Figure A4.7 Printing Summaries, Entries



## A4.8 Data Transfer to other Interfaces

Data Transfer to other interfaces

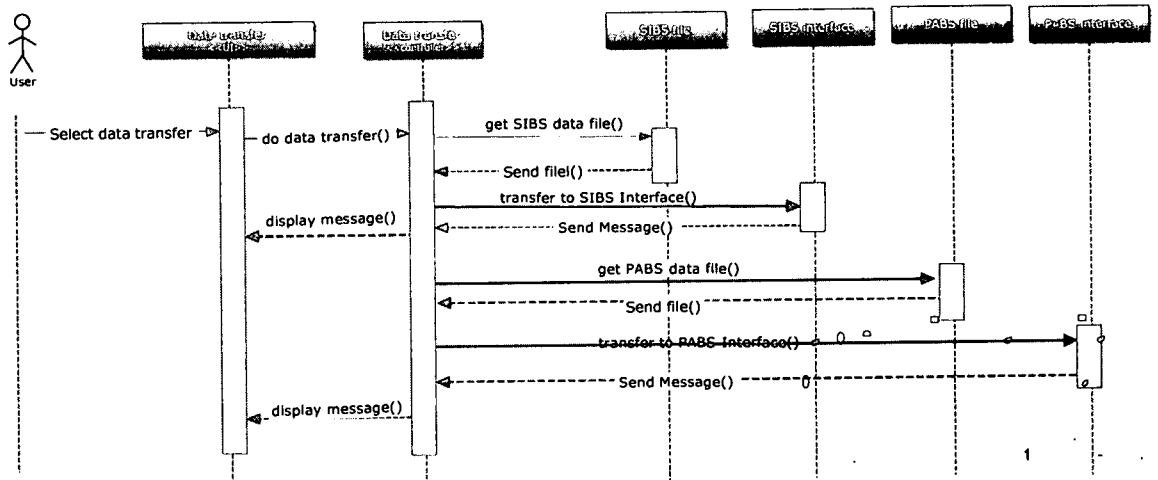


Figure A4.8 Data Transfer to other Interfaces



University of Moratuwa, Sri Lanka.  
Electronic Theses & Dissertations  
[www.lib.mrt.ac.lk](http://www.lib.mrt.ac.lk)

## Appendix B

### B1. Pseudo Codes

#### B1.1 Check files Uploaded by the Customers

Select the Check file option

If the files Exists display the uploaded Files

Else

display Alternate message 'No files to Disply'

Else

Select a File to Download

File transfer to tempory file

If Transferring complete Disply alternate message 'File successfully upload'

Seperate Header file & Detail records

Else

Display Error message 'File Uploaded Error'

#### B1.2 Validation

Enter Data Validation Option

If Files are empty

Display Error message 'File does not exists'

Else

Check Originating account number with Customer's detail file

If Ac.No does not exists Display alternate message

'Not a registered customer'

Clear the header file & detail file

Else

Check Header validation

If Header file validdation fail

Display alternate messages 'Data has been changed'

send a message to customer

Clear the header file & detail file

Else

Start detail record validation

If Error free

Transfer detail file to permanant file

move related customer file to backup folder

### **B1.3. Amend Data**

Enter Bank Code

If incorrect display alternate message 'Invalid Bank code'

Else

Enter Branch code

if incorrect display alternate message 'Invalid Branch code'

Else

Enter Customer file Identification tag

If does not exists display alternate message 'please check the file name'

Else

Enter Account number

If found



Dispaly all related account details

modify the relewant record

Else

Dispaly alternate message 'Record does not exists'

### **B1.4. Delete Data**

Enter Bank code,brnach code,customer file name,account number

If not found display alternate message 'Record does not exists'

Else

Display all the related records

Select the record/records to delete

get administrator's authority to delete

If get authority

delete record/records

Else

Display alternate message 'Record deleted Successfully'

### **B1.5. Delete Customer's Batch file if Necessary**

Enter Customer's file name

seek in temp file

if found

Display total amount, no. of records

get administrator's authority to delete

If authority ok

delete batch

display message 'Batch deleted successfully from temporary file'

Else

Display error message 'Invalid Authorization to delete'

Else

Seek in permanent file

If found

Display total amount, no. of records

get administrator's authority to delete

If get authority

delete batch

display message 'Batch deleted successfully from permanent file'

Else

Display alternate message 'File not found to delete'

### **B1.6. Final Validation**

Click validation

Check bank codes, branch codes with bank code file

if doesn't match write a record in validation file

check transaction codes in transaction code file

if doesn't match write a record in validation file

Check Cr/Dr

if doesn't match write a record in validation file

Check amounts for minuses or any other

if doesn't match write a record in validation file

Check value dates

if doesn't match write a record in validation file

After complete the validation

If record founds in validation file

Print validation report

Else

Display alternate message 'No Errors found'

### **B1.7. Final Process**

Click Final Process option

Get Administrator's authority to process

If Authority ok

Prepare IBTCORE, IBTPABS, OWD, Entries files

Transfer those files to relevant interfaces.

Display alternate messages 'Process completed & transferred to the interfaces'

Else

Display alternate message 'File processing Error'

### **B1.8. Printing Details**

Click Data Printing option

Print relevant branchwise summaries,

Enter Starting Branch no

if finish the print

display alternate message 'Summary printing completed'

Print vouchers (Slips) for manual branch

Enter starting branch number

if finish the print

display alternate message 'Vouchers printing completed'

Print relevant Entries

if finish the print

Display alternate message 'Entries printing completed'

### **B1.9. Batch / Transaction Inquiry**

(If u haven't the batch sending date, first find the batch send date using Batch inquiry)

Select Inquiry option

If batch inquiry

Enter Year,Month,Cooperate customer's name

if not found

display alternate message 'Batch not found'

Else

Display Batch Sending details

If Transaction inquiry

Enter Bank Code,Branch code,Account number,Batch sending date

If found display the details

Else

Display alternate message 'No records found'

### **B1.10. Return upload**

Select upload option (CORE/PABS/Manual)

**If CORE**

Upload return file to return file

if successful display alternate message 'SIBS return file download completed'

Else

Display Error message 'Download Error'

**If PABS**

Uploads return files one by one

if if successful display alternate message 'Successfully downloaded'

Else

Display Error message 'Download Error'

**If Manual**

Enter return details manually one by one

If manual return finish

get printout of manually entered returns.

Check one by one

If any error record found

Modify return detail

if finish modifications

get printout of manually entered returns.

check one by one

modify if any mismatch until correct

If error free manually entered returns

Transfer all the return detail to IBT Permanent file

move return file to backup folder

**B1.11. Customer logon to the web interface**

Customer connect to the Peoples bank web site

Enter Company name, user name and password

Check Company name, user name and password

If Company name incorrect

Display Error message 'Company name Invalid'

Else

If user name Incorrect

Display error message 'Invalid user ID'

Else

if Password mismatch

Display Error message 'Password invalid'

Else

Display File transfer screen

### **B1.12. New User Registration on web**

Select New User Registration

Enter Company Name

Click submit button

If Company Name does not Exist

display Error message 'Company not registered'

Else

Enter User name

click "check availability" button

If User name exists

display alternate message 'User already exists, please use another name'

change user name until it is unique

Else

store user details

display alternate message 'User registration successful'



University of Moratuwa, Sri Lanka.  
Electronic Theses & Dissertations  
[www.lib.mrt.ac.lk](http://www.lib.mrt.ac.lk)



## B2. Screen Shots

### B2.1 Logon Screen

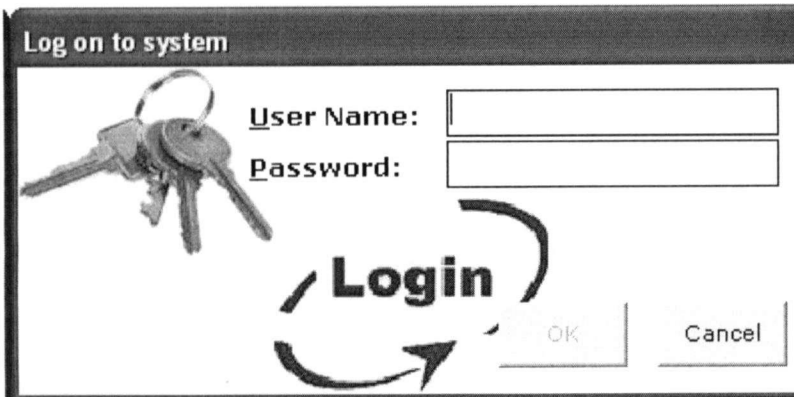


Figure B2.1 Logon Screen

### B2.2 Main Menu



Figure B2.2 Main Menu

### B2.3 New Customer Registration Menu nb

Figure B2.3 New Customer Registration

**B2.4 Customer Data File Processing Menu**



University of Moratuwa, Sri Lanka.  
Electronic Theses & Dissertations  
[www.lib.mrt.ac.lk](http://www.lib.mrt.ac.lk)

**Process Data File**

**Process New File** | **Read / Edit Existing File**

Select data file:

Customer Account No : \_\_\_\_\_ Bank : \_\_\_\_\_  
 Customer Account Name : \_\_\_\_\_ Branch : \_\_\_\_\_  
 Transaction Total : \_\_\_\_\_ No of Records : \_\_\_\_\_

Click browse to select data file

**Transactions on selected file**

Bank	Branch	Account No	Transaction	New Amount	Payment Date
				0.00	19/08/2010

Figure B2.4 Customer Data File Processing menu

### B3. Source Codes

### B3.1 Customer Registration

```
Private Sub Frame1_DragDrop(Source As Control, X As Single, Y As Single)
```

```
End Sub
```

```
Private Sub txtBankCode_KeyPress(KeyAscii As Integer)
```

```
On Error GoTo Err_Handler
```

```
If KeyAscii = 13 Then
```

```
txtBankName.SetFocus
```

```
End If
```

```
Exit Sub
```

```
Err_Handler:
```

```
StrErrMsg = "System Message # " & Str(Err.Number) & " Was Generated By " _  
& Err.Source & vbCrLf & Err.Description
```

```
MsgBox StrErrMsg, vbCritical, strDisplayErrorMessageHeader
```

```
End Sub
```

```
Private Sub txtBankName_KeyPress(KeyAscii As Integer)
```

```
On Error GoTo Err_Handler
```

```
If KeyAscii = 13 Then
```

```
txtAddress1.SetFocus
```

```
End If
```

```
Exit Sub
```

```
Err_Handler:
```

```
StrErrMsg = "System Message # " & Str(Err.Number) & " Was Generated By " _  
& Err.Source & vbCrLf & Err.Description
```

```
MsgBox StrErrMsg, vbCritical, strDisplayErrorMessageHeader
```

```
End Sub
```

```
Private Sub txtAddress1_KeyPress(KeyAscii As Integer)
```

```
On Error GoTo Err_Handler
```

```
If KeyAscii = 13 Then
```

```
txtAddress2.SetFocus
```

```
End If
```

```
Exit Sub
```

```
Err_Handler:
```

```
StrErrMsg = "System Message # " & Str(Err.Number) & " Was Generated By " _  
& Err.Source & vbCrLf & Err.Description
```

```
MsgBox StrErrMsg, vbCritical, strDisplayErrorMessageHeader
```

```
End Sub
```

```
Private Sub txtAddress2_KeyPress(KeyAscii As Integer)
```

```
On Error GoTo Err_Handler
```

```
If KeyAscii = 13 Then
```

```
txtAddress3.SetFocus
```

```
End If
```

```
Exit Sub
```

```

Err_Handler:
    StrErrMsg = "System Message # " & Str(Err.Number) & " Was Generated By " _
        & Err.Source & vbLf & Err.Description
    MsgBox StrErrMsg, vbCritical, strDisplayErrorMessageHeader
End Sub

Private Sub txtAddress3_KeyPress(KeyAscii As Integer)
    On Error GoTo Err_Handler
    If KeyAscii = 13 Then
        txtEmail.SetFocus
    End If
Exit Sub
Err_Handler:
    StrErrMsg = "System Message # " & Str(Err.Number) & " Was Generated By " _
        & Err.Source & vbLf & Err.Description
    MsgBox StrErrMsg, vbCritical, strDisplayErrorMessageHeader
End Sub

Private Sub txtBranchcode_Validate(Cancel As Boolean)
    Dim temprs As New ADO.DB.Recordset
    Dim myobj As PABSDII.BranchData
    Set myobj = New PABSDII.BranchData

    If (txtBranchcode.Text) <> "" Then
        ReturnErrorNo = myobj.selectBranch(txtBranchcode.Text, temprs)
        strReturnErrorDescription = myobj.strErrorDescription
        If ReturnErrorNo = 0 Then
            If Not (temprs.EOF Or temprs.BOF) Then
                If MsgBox("There is a record For your entry," & vbLf & "Do You want To
                    Edit that Record ? ", vbYesNo + vbQuestion, strDisplayErrorMessageHeader) =
                    vbYes Then
                    EditMode
                    temprs.MoveFirst
                    BranchCodeSelected = Iif(IsNull(temprs.Fields("Bbrcd")), "",
                    Trim(temprs.Fields("Bbrcd")))
                    txtBranchcode = Iif(IsNull(temprs.Fields("Bbrcd")), "",
                    Trim(temprs.Fields("Bbrcd")))
                    txtBranchName = Iif(IsNull(temprs.Fields("Bbrname")), "",
                    Trim(temprs.Fields("Bbrname")))
                    txtBankCode = Iif(IsNull(temprs.Fields("Bbnkcd")), "",
                    Trim(temprs.Fields("Bbnkcd")))
                    txtBankName = Iif(IsNull(temprs.Fields("Bbnkdesc")), "",
                    Trim(temprs.Fields("Bbnkdesc")))
                    txtAddress1 = Iif(IsNull(temprs.Fields("Bbradd1")), "",
                    Trim(temprs.Fields("Bbradd1")))
                    txtAddress2 = Iif(IsNull(temprs.Fields("Bbradd2")), "",
                    Trim(temprs.Fields("Bbradd2")))
                End If
            End If
        End If
    End If
End Sub

```

```

        txtAddress3 = IIf(IsNull(temprs.Fields("Bbradd3")), "",
Trim(temprs.Fields("Bbradd3")))
        txtEmail = IIf(IsNull(temprs.Fields("Bbremail")), "",
Trim(temprs.Fields("Bbremail")))
        dtpDataOpen = IIf(IsNull(temprs.Fields("Bdteopn")), Now,
Trim(temprs.Fields("Bdteopn")))
        cmdLookUpBank.Enabled = True
    Else
        Clear
        txtBranchcode.SetFocus
    End If
Else
    addMode
    txtBranchName.SetFocus
End If
End If
End If
End Sub

```

```

Private Sub txtEmail_KeyPress(KeyAscii As Integer)
On Error GoTo Err_Handler
If KeyAscii = 13 Then
    dtpDataOpen.SetFocus
End If
Exit Sub
Err_Handler:
    StrErrMsg = "System Message # " & Str(Err.Number) & " Was Generated By " _
    & Err.Source & vbCrLf & Err.Description
    MsgBox StrErrMsg, vbCritical, strDisplayErrorMessageHeader
End Sub

```

```

Private Sub txtEmail_LostFocus()
    If IsValidEmail(txtEmail.Text) Then
    Else
        DisplayErrorMessage (10002)
        txtEmail.SetFocus
        txtEmail.Text = ""
    End If
End Sub

```

### **B3.2 Customer Data Processing**

```

Private sqlString As String
Public strErrorDescription As String
Public Function insertHeaderRecord(Insertstatus As Integer, strHBankCode As
String, strHBranchCode As String, strHAccNumber As String, dbIHNoofTransaction
As Double, dbIHTotalAmount As Double, dbIHHash As Double, strHBatchSeq As
String, strHYear As String, strHMonth As String, ByRef HeaderNo As Integer) As
Double
On Error GoTo Err_Handler
Dim SHseqno As Integer

If ConnectToDatabase Then
    SHseqno = getNextSeqNo
    If Insertstatus = 1 Then
        sqlString = " insert into customerdatah " & _
            " (CHidno ,CHbnkcd, CHbrcd,CHOacno, CHtottrn, CHtotamt, CHcalc,
CHseqno ,CHsalyear ,CHsalmonth) " & _
            " Values " & _
            " (" & SHseqno & ", " & Trim(strHBankCode) & ", " &
Trim(strHBranchCode) & ", " & Trim(strHAccNumber) & ", " &
dbIHNoofTransaction & ", " & dbIHTotalAmount & ", " & dbIHHash & ", " &
Trim(strHBatchSeq) & ", " & Trim(strHYear) & ", " & Trim(strHMonth) & " )"
    Else
        sqlString = " Update customerdatah set " & _
            " CHbnkcd = " & Trim(strHBankCode) & ", " &
            " CHbrcd = " & Trim(strHBranchCode) & ", " &
            " CHOacno = " & Trim(strHAccNumber) & ", " &
            " CHtottrn = " & dbIHNoofTransaction & ", " &
            " CHtotamt = " & dbIHTotalAmount & ", " &
            " CHcalc = " & dbIHHash & ", " &
            " CHseqno = " & Trim(strHBatchSeq) & " " & _
            " Where Hseqno = " & SHseqno & " "
    End If

    cnn.BeginTrans
        cnn.Execute sqlString
        cnn.CommitTrans
        HeaderNo = SHseqno
    Else
        insertCustomer = 1 'connection failed
    End If

Exit Function
Err_Handler:
    insertCustomer = Err.Number
    strErrorDescription = "System Message # " & Str(Err.Number) & " Was Generated
By " & _

```

```

& Err.Source & vbLf & Err.Description
End Function
Public Function postRecord(HeaderNo As Integer) As Double
On Error GoTo Err_Handler
Dim SHseqno As Integer

If ConnectToDatabase Then
    sqlString = " update customerdatad set CHpost = 1 where CHidno =" &
HeaderNo & ""

    cnn.BeginTrans
    cnn.Execute sqlString
    cnn.CommitTrans
Else
    insertCustomer = 1 'connection failed
End If

Exit Function
Err_Handler:
    insertCustomer = Err.Number
    strErrorDescription = "System Message # " & Str(Err.Number) & " Was Generated
By " _
        & Err.Source & vbLf & Err.Description
End Function

Public Function insertLineRecord(Insertstatus As Integer, CHseqno As Integer,
Clineno As Integer, custBankCode As String, custBranchCode As String,
custAccNumber As String, custTransAmount As Double, cTransNumber As String,
cTransCode As String, cTransType As String, cReference As String, cParticulars As
String, cPaymentdt As Date) As Double
On Error GoTo Err_Handler
If ConnectToDatabase Then
    If Insertstatus = 1 Then
        sqlString = " insert into customerdatad " & _
            " (CHseqno , Clineno ,Cbnkcd, Cbrcd, Ccacno, Camt, Ctrno, Ctrncd,
Ccd, Cref, Cparti, Cpdte ) " & _
            " Values " & _
            "(" & CHseqno & "," & Clineno & "," & Trim(custBankCode) & "," &
Trim(custBranchCode) & "," & Trim(custAccNumber) & "," & custTransAmount
& "," & cTransNumber & "," & cTransCode & "," & Trim(cTransType) & "
," & Trim(cReference) & "," & Trim(cParticulars) & "," & Format(cPaymentdt,
"yyyy/mm/dd") & " )"
    Else
        sqlString = " Update customerdatad set " & _
            " Cbnkcd = " & Trim(custBankCode) & " , " & _
            " Cbrcd = " & Trim(custBranchCode) & " , " & _

```



```

                " Ccagno = " & Trim(custAccNumber) & ", " & _
                " Camt = " & Trim(custTransAmount) & ", " & _
                " Ctrncd = " & Trim(cTransCode) & ", " & _
                " Cpdte = " & Format(cPaymentdt, "yyyy/mm/dd") & " " & _
                " Where CHseqno = " & CHseqno & " And Clineno = " & Clineno & ""
    End If
    cnn.BeginTrans
        cnn.Execute sqlString
    cnn.CommitTrans
Else
    insertCustomer = 1 'connection failed
End If
Exit Function
Err_Handler:
    insertCustomer = Err.Number
    strErrorDescription = "System Message # " & Str(Err.Number) & " Was Generated
By " _
        & Err.Source & vbLf & Err.Description
End Function

Public Function selectAllDataLines(ByRef SelRs As ADODB.Recordset) As Double
On Error GoTo Err_Handler
If ConnectToDatabase Then
    Set rs = New ADODB.Recordset
    sqlString = "select CHseqno, Clineno, Cbnkcd, Bbnkdsc, Cbrcd, Bbrname ,
Ccagno, CDacname , Camt, Ctrncd, Ttrndesc, Gcd, Cref, Cparti,Cpdte " & _
        " From " & _
        " customerdatad as d Inner join bank as b on d.Cbnkcd = b.Bbnkcd " & _
        " inner join bankbranch as br on d.cbrcd = br.Bbrcd " & _
        " inner join corporatecustomer as c on d.Ccagno = c.CDacno " & _
        " inner join transactioncode as tr on d.Ctrncd = tr.Ttrncd "

    OpenADORec sqlString, cnn, rs, "DR"
    Set SelRs = rs
Else
    selectBank = 1 'connection failed
End If
Exit Function
Err_Handler:
    selectBank = Err.Number
    strErrorDescription = "System Message # " & Str(Err.Number) & " Was Generated
By " _
        & Err.Source & vbLf & Err.Description
End Function
Private Function getNextSeqNo() As Integer
On Error GoTo Err_Handler

```

```

If ConnectToDatabase Then
    Set rs = New ADODB.Recordset
    sqlString = "select ifnull(max(CHidno),0) + 1 As nextnum from customerdatah"
    OpenADOREc sqlString, cnn, rs, "DR"
    If rs.RecordCount > 0 Then
        getNextSeqNo = CInt(rs.Fields(0).Value)
    End If

Else
    selectBank_byCode = 1
End If
Exit Function
Err_Handler:
    selectBank_byCode = Err.Number
    strErrorDescription = "System Message # " & Str(Err.Number) & " Was Generated
By " _
        & Err.Source & vbLf & Err.Description
End Function

Public Function DeleteItem(CHseqno As Integer, Clineno As Integer) As Double
On Error GoTo Err_Handler
    If ConnectToDatabase Then
        sqlString = "Delete from customerdatad Where CHseqno = " & CHseqno & "
And Clineno = " & Clineno & ""
        cnn.Execute (sqlString)
    Else
        Delete_Item = 1 ' connection failed
    End If

Exit Function
Err_Handler:
    Delete_Item = Err.Number
    strErrDescription = "System Message # " & Str(Err.Number) & " Was Generated
By " _
        & Err.Source & vbLf & Err.Description

End Function
Public Function DeleteBatch(CHseqno As Integer) As Double
On Error GoTo Err_Handler
    If ConnectToDatabase Then
        sqlString = "Delete from customerdatah Where CHidno = " & CHseqno & " "
        cnn.Execute (sqlString)
        sqlString = ""
        sqlString = "Delete from customerdatad Where CHseqno = " & CHseqno & " "
        cnn.Execute (sqlString)
    Else

```

```

Delete_Item = 1 ' connection failed
End If

Exit Function
Err_Handler:
Delete_Item = Err.Number
strErrDescription = "System Message # " & Str(Err.Number) & " Was Generated
By " _
                & Err.Source & vbLf & Err.Description

End Function

Public Function selectAllDataLinesBySeq(CHseqno As Integer, ByRef SelRs As
ADODB.Recordset) As Double
On Error GoTo Err_Handler
If ConnectToDatabase Then
Set rs = New ADODB.Recordset
sqlString = "select h.CHidno as 'CI No', Clineno as 'Line No', Cbnkcd as 'Bank
Code', Bbnkdesc as 'Bank Name', Cbrcd as 'Branch Code', Bbrname as 'Branch
Name', Ccacno as 'Account No', CDacname as 'Account Name', Camt as 'Amount',
Ctrno, Ctrncd as 'Trn code', Trndesc as 'Trn Name', Ccd as 'Type', Cref as
'Reference', Cparti as Particulars ,Cpdte as 'Payment date' " & _
" From " & _
" customerdata d as d Inner join bank as b on d.Cbnkcd = b.Bbnkcd " & _
" Inner join customerdata h on d.CHseqno = h.CHidno " & _
" inner join bankbranch as br on d.cbrcd = br.Bbrcd " & _
" inner join corporatecustomer as c on h.CHoacno = c.CDacno " & _
" inner join transactioncode as tr on d.Ctrncd = tr.Trncd Where CHseqno = "
& h.CHidno & " "

OpenADOREc sqlString, cnn, rs, "DR"
Set SelRs = rs
Else
selectBank = 1 'connection failed
End If
Exit Function
Err_Handler:
selectBank = Err.Number
strErrorDescription = "System Message # " & Str(Err.Number) & " Was Generated
By " _
                & Err.Source & vbLf & Err.Description

End Function

```

```

Public Function selectBatchNobyCustomer(Tyear As String, Tmonth As String,
Taccount As String, intpost As Integer, ByRef SelRs As ADODB.Recordset) As
Double
On Error GoTo Err_Handler
If ConnectToDatabase Then
    Set rs = New ADODB.Recordset
    sqlString = "select CHseqno from customerdatah where CHsalyear = " & Tyear &
    "" and CHsalmmonth = " & Tmonth & "" and CHOacno = " & Taccount & "" and
    CHpost = " & intpost & ""
    OpenADOREc sqlString, cnn, rs, "DR"
    Set SelRs = rs
Else
    selectBank = 1 'connection failed
End If
Exit Function
Err_Handler:
    selectBank = Err.Number
    strErrorDescription = "System Message # " & Str(Err.Number) & " Was Generated
By " _
        & Err.Source & vbLf & Err.Description
End Function
Public Function checkExistingRecordsStatus(Tyear As String, Tmonth As String,
Taccount As String) As Integer
On Error GoTo Err_Handler
If ConnectToDatabase Then
    Set rs = New ADODB.Recordset
    sqlString = "select CHpost from customerdatah where CHsalyear = " & Tyear & ""
and CHsalmmonth = " & Tmonth & "" and CHOacno = " & Taccount & ""
    OpenADOREc sqlString, cnn, rs, "DR"
    If rs.RecordCount > 0 Then
        checkExistingRecordsStatus = IIf(IsNull(rs.Fields("CHpost")), 2,
Trim(rs.Fields("CHpost")))
    Else
        checkExistingRecordsStatus = 2 "no data
    End If
Else
    selectBank = 1000 'connection failed
End If
Exit Function
Err_Handler:
    selectBank = Err.Number
    strErrorDescription = "System Message # " & Str(Err.Number) & " Was Generated
By " _
        & Err.Source & vbLf & Err.Description
End Function

```

```

Public Function selectSavedHeaderData(Tyear As String, Tmonth As String,
Taccount As String, Tbatch As String, ByRef SelRs As ADODB.Recordset) As
Double
On Error GoTo Err_Handler
If ConnectToDatabase Then
    Set rs = New ADODB.Recordset
    sqlString = "select * from customerdatah where CHsalyear = " & Tyear & " and
CHsalmmonth = " & Tmonth & " and CHOacno = " & Taccount & " and CHseqno ="
& Tbatch & " and CHpost = 0 "
    OpenADOREc sqlString, cnn, rs, "DR"
    Set SelRs = rs
Else
    selectBank = 1 'connection failed
End If
Exit Function
Err_Handler:
    selectBank = Err.Number
    strErrorDescription = "System Message # " & Str(Err.Number) & " Was Generated
By " _
        & Err.Source & vbCrLf & Err.Description
End Function

```

```

Public Function SaveValidationError(strCopAccNo As String, strYear As String,
strMonth As String, intLin As Integer, strBatch As String, strErr As String) As
Double
On Error GoTo Err_Handler
If ConnectToDatabase Then
    sqlString = " insert into validityreport (Vcacno, Vyear, Vmonth, Vlinno, Vseq,
Verr) " & _
        " Values " & _
        "(" & strCopAccNo & ", " & Trim(strYear) & ", " & Trim(strMonth) & ", "
& intLin & ", " & strBatch & " , " & Trim(strErr) & " )"
    cnn.BeginTrans
    cnn.Execute sqlString
    cnn.CommitTrans
Else
    SaveValidationError = 1 'connection failed
End If
Exit Function
Err_Handler:
    SaveValidationError = Err.Number
    strErrorDescription = "System Message # " & Str(Err.Number) & " Was Generated
By " _
        & Err.Source & vbCrLf & Err.Description
End Function

```

