**Financial Control System (FCS)**

**Accounts Payable Project.**

**Programmer 's Manual**

|  |  |  |
| --- | --- | --- |
| Filing Reference : | ISS/FCS2/TS.3/1 | |
| Document Title : | Programmer's Manual | |
| Version : | 1.0 | |
| Prepared By : | Low Siew Heng | |
| Date Created : | Friday, March 18, 2011 | |
| Approved By : |  |  |
|  | Howard Russon | Date |
|  | Project Manager |  |
| Authorised By : |  |  |
|  | Mike Fenton | Date |
|  | Project Director |  |
| **1993 ISS.**  **The information contained in this document is the property of ISS. The contents must not be reproduced, wholly or in part, for purposes other than for which it has been supplied, without the prior permission of ISS, or, if it has been furnished under contract to another party, as expressly authorised under that contract. ISS shall not be liable for any errors or omissions.** | ISS | |

Document Reference : **ISS/FCS2/TS.3/1**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Version | Date | Author | Authorised by | Date |
| 1.0 | 18/03/11 | Low Siew Heng |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

**Financial Control System (FCS)**

**Accounts Payable Project.**

**Programmer's Manual**

**Distribution :**

|  |  |  |
| --- | --- | --- |
| **Name** | **Department** | **Organisation** |
| Mike Fenton |  |  |
| Howard Russon |  |  |
| Lim Swee Cheang |  |  |
| Low Siew Heng |  |  |
| Frank Jenkins |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

**Table of Contents.**

# 1. INTRODUCTION.

The Institute of Systems Science (ISS) requires the development of a Financial Control System (FCS). This will involve the computerisation of the two major functions of the current manual system:

1. Invoice/Accounts Receivable; and

2. Accounts Payable.

The development is required to be undertaken in two stages. The first stage, to develop the Invoice/Accounts Receivable module, has been completed. This project is required to undertake Stage 2 of the development, to develop the Accounts Payable module of the system. In addition, the project is required to undertake the enhancement of the existing module (Invoice / Accounts Receivable) to meet the new requirements.

## 1.1 Purpose.

The purpose of this document is to describe the system design architectures so as to enable maintenance programmers to understand the Accounts Payable module of the FCS.

## 1.2 Audience.

The intended readers of this manual are the programmers, who are responsible for maintaining this system. This document should provide all the necessary informations.

## 1.3 Organisation.

The System Overview is described in Section 2. The data base design and the client software are specified in Section 3 and 4 respectively. Section 5 describes the data base interface software design. Finally, the configuration information is defined in Section 6.

## 1.4 References.

To fully understand the background to this project, the reader should also be familiar with:

1. FCS Accounts Payable Project Plan, reference ISS/FCS2/MP.1/1, version 1.0, dated 9 July 1993.

2. FCS Accounts Payable Quality Plan, reference ISS/FCS2/MQ.1/1, version 1.0, dated 3 March 1994.

3. FCS Accounts Payable User Requirement Specification, reference ISS/FCS2/TS.1/1, version 1.0, dated 14 October 1993.

4. FCS Accounts Payable System Specification, reference ISS/FCS2/TS.2/1, version 1.0, dated 4 May 1994.

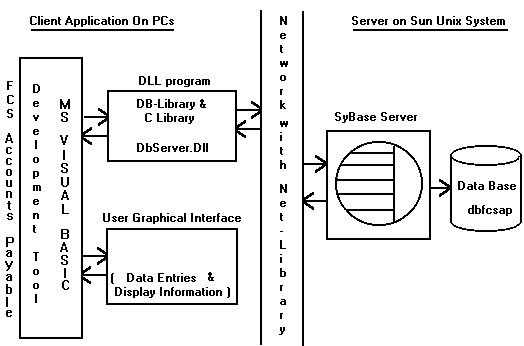
5. FCS Accounts Payable Project, User's Manual, reference ISS/FCS2/TU.1/1, version 1.0, dated 29 December 1994.

6. FCS Invoices/Accounts Recievable Project, Programmer's Manual, reference ISS/FCS/TU.3/1, version 1.0, dated 29 December 1994.

# 2. SYSTEM OVERVIEW.

FCS Accounts Payable Project is a MS Windows application developed by using the development tool MS VISUAL BASIC. The database is implemented using the SyBase SQL Server. A DLL (dynamic link libraries) program is written in C language, utilising the DB-Library that is provided by the SyBase open client server software. In order to access to SyBase SQL server, this DLL program is called from the MS VISUAL BASIC program. The architecture of the FCS Accounts Payable system operating in the Client/Server mode is shown in Figure 2.1.

Figure 2.1: The Client/Server Architecture Implemented in FCS A/P



# 3. DATA BASE DESIGN.

The database of the FCS program is named '***dbfcsap***'. The space allocated to dbfcsap is 20 MB for the data storage and 10 MB for the transactions log. The database (dbfcsap) consists of 23 tables. There is a working table among these tables.

3.1 Cost Centre Table --- tblCostC

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Column No. | Column Name | Description | Column  Type | Length |
| 1 |  | Cost Centre A/C code | char | 20 |
| 2 |  | Description | varchar | 40 |
| 3 |  | Cost centre type  (T-ATA, N-RND, A-Apple-ISS, O-Others) | char | 1 |
| 4 |  | Abbr. description | varchar | 20 |
| 5 |  | Special Account | char | 2 |

3.2 Vote Table. -- tblVote

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Column No. | Column Name | Description | Column  Type | Length |
| 1 |  | Vote number | varchar | 10 |
| 2 |  | Cost Centre A/C code | char | 20 |
| 3 |  | Vote description | varchar | 40 |
| 4 |  | Vote type | char | 1 |
| 5 |  | Period | tinyint |  |
| 6 |  | Financial year | tinyint |  |

3.3 Currency Table --- tblCurrency

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Column No. | Column Name | Description | Column  Type | Length |
| 1 | Currency | Currency Code | char | 3 |
| 2 | CurrDesc | Description | varchar | 15 |
| 3 | ExchRate | Exchange Rate | float | (4, 6) |
| 4 | ExchDM | Division / Multiplication | char | 1 |
| 5 | ExchUnit | Basic unit | TinyInt | 3 |
| 6 | Sybase user ID | Sybase user ID | varchar | 10 |
| 7 |  | Date updated | datetime |  |

3.4 Professional Societies Table --- tblProfSoc

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Column No. | Column Name | Description | Column  Type | Length |
| 1 |  | Society code | varchar | 10 |
| 2 |  | Society name | varchar | 40 |

3.5 Control Table --- tblCtrl

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Column No. | Column Name | Description | Column  Type | Length |
| 1 |  | Financial year | numeric | (2, 0) |
| 2 |  | Last generated invoice reference number | numeric | (6, 0) |

3.6 Purchase Orders Table ---- tblPO

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Column No. | Column Name | Description | Column  Type | Length |
| 1 |  | P/O number | char | 10 |
| 2 |  | Supplier code | char | 6 |
| 3 |  | Date of P/O | datetime |  |
| 4 |  | PRF number | varchar | 10 |
| 5 |  | Currency of the P/O amount | char | 3 |
| 6 |  | P/O amount | money |  |
| 7 |  | P/O amount in S$ equivalent | money |  |
| 8 |  | Exchange rate | float |  |
| 9 |  | Paid amount. | money |  |
| 10 |  | Cancelled P/O amount | money |  |
| 11 |  | Cancelled P/O amount in S$ | money |  |
| 12 |  | Reference number | varchar | 10 |
| 13 |  | Description 1 | varchar | 40 |
| 14 |  | Description 2 | varchar | 40 |
| 15 |  | Pre-payment status (Y/N) | char | 1 |
| 16 |  | Cheque number | varchar | 20 |
| 17 |  | Delivery status | char | 1 |
| 18 |  | Date of payment | datetime |  |
| 19 |  | Date of P/O item cancelled | datetime |  |
| 20 |  | Financial year | numeric | (2, 0) |
| 21 |  | Financial month | numeric | (2, 0) |
| 22 |  | Exchange differences | money |  |
| 23 |  | Amount adjusted | money |  |
| 24 |  | Amount adjusted in S$ | money |  |

3.7 Received Invoices Table ---- tblInvRecd

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Column No. | Column Name | Description | Column  Type | Length |
| 1 |  | Invoice number | char | 15 |
| 2 |  | P/O number | varchar | 10 |
| 3 |  | D/O number | varchar | 15 |
| 4 |  | Reference number | varchar | 10 |
| 5 |  | Date of invoice | datetime |  |
| 6 |  | Supplier code | char | 6 |
| 7 |  | Currency of the invoice amount | char | 3 |
| 8 |  | Invoice amount | money |  |
| 9 |  | Invoice amount (S$ equivalent) | money |  |
| 10 |  | Exchange rate | float |  |
| 11 |  | Date sent to Bursar's office | datetime |  |
| 12 |  | Amount paid | money |  |
| 13 |  | Exchange differences | money |  |
| 14 |  | Date of payment | datetime |  |
| 15 |  | Remarks 1 | varchar | 40 |
| 16 |  | Remarks 2 | varchar | 40 |
| 17 |  | Amount adjusted | money |  |
| 18 |  | Amount adjusted in S$ | money |  |
| 19 |  | Invoice status (Final) | char | 1 |
| 20 |  | Financial year | numeric | (2, 0) |
| 21 |  | Financial month | numeric | (2, 0) |
| 22 |  | Invoice type | char | 1 |

3.8 Credit/Debit Notes Table --- tblCDNote

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Column No. | Column Name | Description | Column  Type | Length |
| 1 |  | Credit/Debit Note number | varchar | 15 |
| 2 |  | Type of Credit/Debit note  (CN-credit note DN-debit note) | char | 2 |
| 3 |  | Invoice number | varchar | 15 |
| 4 |  | P/O number | varchar | 10 |
| 5 |  | Date of credit/debit note | datetime |  |
| 6 |  | Supplier code | char | 6 |
| 7 |  | Currency | char | 3 |
| 8 |  | Amount | money |  |
| 9 |  | Amount in S$ | money |  |
| 10 |  | Exchange rate | float |  |
| 11 |  | Remarks 1 | varchar | 40 |
| 12 |  | Remarks 2 | varchar | 40 |

3.9 Vote Commitment Table --- tblVoteCommit

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Column No. | Column Name | Description | Column  Type | Length |
| 1 |  | Cost centre A/C numberr | varchar | 20 |
| 2 |  | Vote number | varchar | 10 |
| 3 |  | Table type | char | 2 |
| 4 |  | Form number | varchar | 15 |
| 5 |  | Supplier code | char | 6 |
| 6 |  | Reference number | varchar | 10 |
| 7 |  | Date | datetime |  |
| 8 |  | Amount | money |  |
| 9 |  | Amount in S$ | money |  |
| 10 |  | Percentage of vote commitment | numeric | (5, 2) |
| 11 |  | Currency | char | 3 |
| 12 |  | Exchange rate | float |  |
| 13 |  | Cancelled amount | money |  |
| 14 |  | Cancelled amount in S$ | money |  |
| 15 |  | Amount paid | money |  |
| 16 |  | Exchange differences | money |  |
| 17 |  | Financial year | numeric | (2, 0) |
| 18 |  | Financial month | numeric | (2, 0) |
| 19 |  | P/O or Invoice number | varchar | 15 |

3.10 Cancelled P/O Table --- tblCancelPO

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Column No. | Column Name | Description | Column  Type | Length |
| 1 |  | P/O number | varchar | 10 |
| 2 |  | Reference number | varchar | 10 |
| 3 |  | Date of P/O item cancelled | datetime |  |
| 4 |  | Currency | char | 3 |
| 5 |  | Amount | money |  |
| 6 |  | Amount in S$ | money |  |
| 7 |  | Exchange rate | float |  |
| 8 |  | Remarks 1 | varchar | 40 |
| 9 |  | Remarks 2 | varchar | 40 |

3.11 Cancelled Vote Commitment of P/O Table --- tblCancelVote

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Column No. | Column Name | Description | Column  Type | Length |
| 1 |  | Cost centre A/C numberr | varchar | 20 |
| 2 |  | Vote number | varchar | 10 |
| 3 |  | Reference number | varchar | 10 |
| 4 |  | Currency | char | 3 |
| 5 |  | Amount | money |  |
| 6 |  | Amount in S$ | money |  |
| 7 |  | Percentage of vote commitment | numeric | (5, 2) |
| 8 |  | Exchange rate | float |  |

3.12 Claims\Expense Payments Table --- tblClaim

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Column No. | Column Name | Description | Column  Type | Length |
| 1 |  | Claim reference number | varchar | 15 |
| 2 |  | Type of claim | char | 1 |
| 3 |  | Date | datetime |  |
| 4 |  | Currency | char | 3 |
| 5 |  | Claim amount | money |  |
| 6 |  | Claim amount in S$ | money |  |
| 7 |  | Exchange rate | float |  |
| 8 |  | Amount paid | money |  |
| 9 |  | Exchange differences | money |  |
| 10 |  | Date of payment | datetime |  |
| 11 |  | Remarks 1 | varchar | 40 |
| 12 |  | Remarks 2 | varchar | 40 |
| 13 |  | Financial year | numeric | (2, 0) |
| 14 |  | Financial month | numeric | (2, 0) |

3.13 Claim for Professional Societies Subscription Table --- tblSubscription

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Column No. | Column Name | Description | Column  Type | Length |
| 1 |  | Staff ID | numeric | (6, 0) |
| 2 |  | Claim reference number | varchar | 10 |
| 3 |  | Professional Society code | varchar | 10 |
| 4 |  | Membership period | smallint |  |
| 5 |  | Date | datetime |  |
| 6 |  | Amount | money |  |
| 7 |  | Subscription fees | money |  |
| 8 |  | Currency | char | 3 |
| 9 |  | Remarks 1 | varchar | 40 |
| 10 |  | Remarks 2 | varchar | 40 |
| 11 |  | Financial year | numeric | (2, 0) |
| 12 |  | Financial month | numeric | (2, 0) |

3.14 Committed Vote of Subscription claims Table --- tblSubsVote

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Column No. | Column Name | Description | Column  Type | Length |
| 1 |  | Staff ID | numeric | (6, 0) |
| 2 |  | Claim reference number | varchar | 10 |
| 3 |  | Professional Society code | varchar | 10 |
| 4 |  | Membership period | smallint |  |
| 5 |  | Cost centre A/C numberr | varchar | 20 |
| 6 |  | Vote number | varchar | 10 |
| 7 |  | Amount | money |  |
| 8 |  | Vote description | varchar | 40 |

3.15 Staff Travel Table --- tblTravel

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Column No. | Column Name | Description | Column  Type | Length |
| 1 |  | Staff ID | numeric | (6, 0) |
| 2 |  | Claim reference number | varchar | 10 |
| 3 |  | Date | datetime |  |
| 4 |  | Event 1 | varchar | 40 |
| 5 |  | Event 2 | varchar | 40 |
| 6 |  | Event 3 | varchar | 40 |
| 7 |  | Event 4 | varchar | 40 |
| 8 |  | Date of event | varchar | 20 |
| 9 |  | Type of Travel (L-Local, O-Overseas) | char | 1 |
| 10 |  | Budgeted expenses | money |  |
| 11 |  | Air fare | money |  |
| 12 |  | Advance allowance | money |  |
| 13 |  | Final allowance | money |  |
| 14 |  | Registration fees | money |  |
| 15 |  | Registration fees in S$ | money |  |
| 16 |  | Currency | char | 3 |
| 17 |  | Exchange differences | money |  |
| 18 |  | Date of payment | datetime |  |
| 19 |  | Record update status (U-updated) | char | 1 |
| 20 |  | Financial year | numeric | (2, 0) |
| 21 |  | Financial month | numeric | (2, 0) |

3.16 Staff Travel Table --- tblTravelVote

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Column No. | Column Name | Description | Column  Type | Length |
| 1 |  | Staff ID | numeric | (6, 0) |
| 2 |  | Claim reference number | varchar | 10 |
| 3 |  | Cost centre A/C numberr | varchar | 20 |
| 4 |  | Vote number | varchar | 10 |
| 5 |  | Budgeted expenses | money |  |
| 6 |  | Advance allowance | money |  |
| 7 |  | Final allowance | money |  |
| 8 |  | Air fare | money |  |
| 9 |  | Registration fees | money |  |
| 10 |  | Registration fees in S$ | money |  |
| 11 |  | Currency | char | 3 |

3.17 Daily Journal Adjustment Table --- tblJAdj

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Column No. | Column Name | Description | Column  Type | Length |
| 1 |  | Journal reference number | varchar | 10 |
| 2 |  | Type of Adjustment | char | 1 |
| 3 |  | Date | datetime |  |
| 4 |  | Cost centre A/C numberr | varchar | 20 |
| 5 |  | Vote number | varchar | 10 |
| 6 |  | Financial year | numeric | (2, 0) |
| 7 |  | Financial month | numeric | (2, 0) |
| 8 |  | Amount paid | money |  |
| 9 |  | Amount received | money |  |
| 10 |  | Remarks 1 | varchar | 40 |
| 11 |  | Remarks 2 | varchar | 40 |
| 12 |  | Sequence number | smallint |  |

3.18 Updated Journal Adjustment Table --- tblJAdjM

The structure is same as the table (tblJAdj).

3.19 Daily Petty Cash Table --- tblPettyCash

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Column No. | Column Name | Description | Column  Type | Length |
| 1 |  | Voucher number | char | 10 |
| 2 |  | Type of transaction | char | 1 |
| 3 |  | Sequence number | smallint |  |
| 4 |  | Date | datetime |  |
| 5 |  | Cash balance | money |  |
| 6 |  | Amount | money |  |
| 7 |  | Amount paid | money |  |
| 8 |  | GST amount | money |  |
| 9 |  | Date of payment | datetime |  |
| 10 |  | Remarks 1 | varchar | 40 |
| 11 |  | Remarks 2 | varchar | 40 |
| 12 |  | Financial year | numeric | (2, 0) |
| 13 |  | Financial month | numeric | (2, 0) |

3.20 Archived Petty Cash Table --- tblPettyM

The structure is same as the table (tblPettyCash).

3.21 User Function Table -- tblFunction

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Column No. | Column Name | Description | Column  Type | Length |
| 1 |  | Sybase user ID | varchar | 8 |
| 2 |  | Menu name | varchar | 10 |
| 3 |  | Item name | varchar | 10 |
| 4 |  | Item enabled | varchar | 2 |
| 5 |  | Item disabled | varchar | 2 |

3.22 BudgetTable --- tblBudget

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Column No. | Column Name | Description | Column  Type | Length |
| 1 |  | Vote number | varchar | 10 |
| 2 |  | Cost centre A/C numberr | varchar | 20 |
| 3 |  | Financial year | numeric | (2, 0) |
| 4 |  | Budgeted amount | money |  |
| 5 |  | Balance B/F | money |  |
| 6 |  | Balance C/F | money |  |
| 7 |  | Amount Committed | money |  |
| 8 |  | Date updated | datetime |  |
| 9 |  | Reference number (Reserved) | int |  |

3.23 Purchase Orders Table ---- tblVoteFunds

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Column No. | Column Name | Description | Column  Type | Length |
| 1 |  | Vote number | varchar | 10 |
| 2 |  | Cost centre A/C numberr | varchar | 20 |
| 3 |  | Financial year | numeric | (2, 0) |
| 4 |  | Budgeted amount | money |  |
| 5 |  | Balance B/F | money |  |
| 6 |  | Balance C/F | money |  |
| 7 |  | Amount Committed in Jan | money |  |
| 8 |  | Amount Committed in Feb | money |  |
| 9 |  | Amount Committed in March | money |  |
| 10 |  | Amount Committed in April | money |  |
| 11 |  | Amount Committed in May | money |  |
| 12 |  | Amount Committed in June | money |  |
| 13 |  | Amount Committed in July | money |  |
| 14 |  | Amount Committed in Aug | money |  |
| 15 |  | Amount Committed in Sept | money |  |
| 16 |  | Amount Committed in Oct | money |  |
| 17 |  | Amount Committed in Nov | money |  |
| 18 |  | Amount Committed in Dec | money |  |
| 19 |  | Amount Paid in Jan | money |  |
| 20 |  | Amount Paid in Feb | money |  |
| 21 |  | Amount Paid in March | money |  |
| 22 |  | Amount Paid in April | money |  |
| 23 |  | Amount Paid in May | money |  |
| 24 |  | Amount Paid in June | money |  |
| 25 |  | Amount Paid in July | money |  |
| 26 |  | Amount Paid in Aug | money |  |
| 27 |  | Amount Paid in Sept | money |  |
| 28 |  | Amount Paid in Oct | money |  |
| 29 |  | Amount Paid in Nov | money |  |
| 30 |  | Amount Paid in Dec | money |  |
| 31 |  | Amount Received in Jan | money |  |
| 32 |  | Amount Received in Feb | money |  |
| 33 |  | Amount Receivedd in March | money |  |
| 34 |  | Amount Received in April | money |  |
| 35 |  | Amount Received in May | money |  |
| 36 |  | Amount Received in June | money |  |
| 37 |  | Amount Received in July | money |  |
| 38 |  | Amount Received in Aug | money |  |
| 39 |  | Amount Received in Sept | money |  |
| 40 |  | Amount Received in Oct | money |  |
| 41 |  | Amount Received in Nov | money |  |
| 42 |  | Amount Received in Dec | money |  |
| 43 |  | Amount Adjusted in Jan | money |  |
| 44 |  | Amount Adjusted in Feb | money |  |
| 45 |  | Amount Adjustedd in March | money |  |
| 46 |  | Amount Adjusted in April | money |  |
| 47 |  | Amount Adjusted in May | money |  |
| 48 |  | Amount Adjusted in June | money |  |
| 49 |  | Amount Adjusted in July | money |  |
| 50 |  | Amount Adjusted in Aug | money |  |
| 51 |  | Amount Adjusted in Sept | money |  |
| 52 |  | Amount Adjusted in Oct | money |  |
| 53 |  | Amount Adjusted in Nov | money |  |
| 54 |  | Amount Adjusted in Dec | money |  |
| 55 |  | Exchange differences | money |  |

3.24 Tables of Cancelled Records.

The cancelled records of P/O, invoice and vote Received are stored in the following six tables.

Table Remarks

delPO It is used to store the detailed records of the cancelled invoices. The data structure is the same as the table tblPO (refer to Section 3.5).

delInvRecd It is used to store the cancelled invoices. The data structure is the same as the table tblInvoice (refer to Section 3.4) except that there are two extra columns: **UserId** and **DelDate**.

delCDNote It is used to store the detailed records of the cancelled invoices. The data structure is the same as the table tblInvDetail (refer to Section 3.5).

delVoteCommit It is used to store the cancelled remittance records. The data structure is the same as the table tblRemitt (refer to Section 3.6) except that there are two extra columns: **UserId** and **DelDate**.

**Note : UserId** is the user id of the staff who deletes the records. **DelDate** is the date on which the records deleted. The data types of **UserId** and **DelDate** are **varchar(10)** and **datetime** respectively.

3.25 Working Table -- tblVCR

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Column No. | Column Name | Description | Column  Type | Length |
| 1 |  | Cost centre A/C numberr | varchar | 20 |
| 2 |  | Vote number | varchar | 10 |
| 3 |  | Table type | char | 2 |
| 4 |  | Form number | varchar | 15 |
| 5 |  | Reference number | varchar | 10 |
| 6 |  | Date | datetime |  |
| 7 |  | Amount (in S$) | money |  |
| 8 |  | Cancelled amount in S$ | money |  |
| 9 |  | P/O or Invoice number | varchar | 15 |
| 10 |  | Supplier code | char | 6 |
| 11 |  | Budgeted amount | money |  |
| 12 |  | Balance B/F | money |  |
| 13 |  | Vote description | varchar | 40 |
| 14 |  | Cost centre description | varchar | 40 |
| 15 |  | Supplier's name | varchar | 40 |

This working table is used to generate the Vote Control Register report.

3.26 Supplier Table -- Supplier

This table is one of the table in database icsdb that is created for the Inventory Control System.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Column No. | Column Name | Description | Column  Type | Length |
| 1 |  | Supplier name | varchar | 40 |
| 2 |  | Address 1 | varchar | 40 |
| 3 |  | Address 2 | varchar | 40 |
| 4 |  | Address 3 | varchar | 40 |
| 5 |  | Telephone | varchar | 15 |
| 6 |  | Extention | varchar | 4 |
| 7 |  | Fax number | varchar | 15 |
| 8 |  | Supplier code | char | 6 |

3.27 Staff Information Table -- tblStaffInfo

This table is one of the table in database personneldb that is created for the Staff Information System.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Column No. | Column Name | Description | Column  Type | Length |
| 1 |  | Staff ID |  | 10 |
| 2 |  | Staff Name |  |  |
| 3 |  | Email ID |  |  |
| 4 |  | Nick name |  |  |
| 5 |  | Date Joined |  |  |
| 6 |  | Designation ID |  |  |
| 7 |  | Title ID |  |  |
| 8 |  | Group ID |  |  |
| 9 |  | Location ID |  |  |
| 10 |  | Telephone Extention |  |  |
| 11 |  | Section ID |  |  |
| 12 |  | Sybase User ID |  |  |
| 13 |  | Qualification |  |  |

3.28 Program Information Table -- tblProgramInfo

This table is one of the table in database personneldb that is created for the Staff Information System.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Column No. | Column Name | Description | Column  Type | Length |
| 1 | colProgramID | Program ID | smallint |  |
| 2 | colProgramName | Program Name | varchar | 80 |
| 3 | colGroupID | Group ID | smallint |  |
| 4 | colCCNo | Cost Centre A/C No. | varchar | 20 |
| 5 | colPubInfo | Publication Information | tinyint |  |

# 4. CLIENT SOFTWARE DESIGN.

The methods defined in each form are described in this section.

4.1 AboutBox Form.

The form displays the information about system resources and the available memory.

4.1.1 Procedures :

1. Form\_Load().

To load the form and display the system resources information in the form.

2. cmdOK\_Click().

To close the dialog box ( form).

4.2 Budget Form.

The form defines all the functions to be used to update the budget account for a financial year.

4.2.1 Procedures :

1. cmdCancel\_Click().

To perform procedure **ResetField** (see item 12) to reset the items shown in the dialog box.

2. cmdOK\_Click().

To perform procedures **SaveBudget** (see item 13) or **DeleteBudget** (see item 3) based on the selected menu item from the top of the dialog box.

3. DeleteBudget().

To delete a record from the database table when the user is confirmed to delete the record.

4. Form\_Activate().

To load the Cost Centre table.

5. Form\_Load().

To load the form and perform procedure **ResetField** (see item 12).

6. GetBudget().

To retrieve a record from the database table tblVoteFunds and display the detailed information in the dialog box.

7. lstCostC\_Click().

To display a list of votes of a selected cost centre in the Vote list box.

8. lstVoteList\_Click().

To display the vote description of a selected vote. If the selected menu item is **'Add'**, the procedure **GetBudget** (see item 6) is performed.

9. MenuItem(Index As Integer).

To perform procedure **ResetField** (see item 12) based on the selected menu item from the top of the dialog box.

10. mnuMItem\_Click(Integer As Integer).

To perform procedure **MenuItem** (see Item 9) based on the selected menu item.

11. NewEntry().

To check whether the entry already exists.

12. ResetField().

To reset the content in each entry.

13. SaveBudget().

To update the database table tblVoteFunds.

14. txtBudget\_LostFocus().

To validate the entered data in the budget text box after loosing the focus of this text box.

15. txtFinYr\_KeyPress(KeyAscii As Integer).

To set the cursor at the budget text box once user pressed the **'Enter'** key.

16. txtFinYr\_LostFocus().

To validate the entered data in the financial year text box after loosing the focus. The procedure **GetBudget** (see item 6) is performed when the selected menu item is **'Add'**.

17. UpdBudger() As Integer.

To update the database table tblBudget.

4.3 BudgetOpt Form.

The form is used for user to select the necessary conditions for viewing or printing the budget account report.

4.3.1 Procedures :

1. chkVote\_Click(Value As Integer).

To set the value of the global variable ***gszSelection*** based on the status of the **'All vote'** check box.

2. cmdCancel\_Click().

To close the dialog box.

3. cmdOK\_Click().

To set the value of global variables ***gnFinYr*** and ***gnOption***. To fill the selected votes in the vote list box of the ReportFrm (see Section 4.24) for printing the budget account report.

4. Form\_Load().

To load the form.

5. optCostC\_Click(Index As Integer, Value As Integer).

To load the database table tblCostC in the vote list box based on the selected option button of the cost centre type.

4.4 CancelPO Form.

The form is used for user to cancel an item from the purchase order.

4.4.1 Procedures :

1. CanCancel(Index As Integer).

To check whether the user wishes to update the record before exiting the function. If it is, perform procedure **SaveCancelPO** (see item 20).

2. cmdCancel\_Click().

To perform the procedure **CanCancel** (see item 1).

3. cmdDelete\_Click().

To remove the selected vote commitment from the list of cancelled vote commitment when the **'Delete'** button is pressed.

4. cmdList\_Click().

To load the DateFrm form for user to select a range of records to be retrieved. Performs procedure **LoadRec** (see item 24 in Section 4.43.1) to load the database table tblCancelPO.

5. cmdOK\_Click().

To perform procedure **SaveCancelPO** (see item 20), **UpdCancelPO** (see item 29) or **DelCancelPO** (see item 7) based on the selected menu item.

6. cmdSelect\_Click().

To close the dialog box.

7. DelCancelPO().

Perform procedure **DelCancelVote** (see item 8) and delete record in the database table tblCancelPO.

8. DelCancelVote() As Integer.

To update the database tables tblPO, tblVoteCommit, tblVoteFunds and tblBudget as well as delete cancelVote when the selected menu item is **Delete**.

9. DispAdjFunds().

To perform procedure **AdjustedFunds** (see Section 4.37.1 item 1) to compute the vote commitment based on the entered amount to be cancelled. Displays the adjusted vote commitment in the funds commitment list box again.

10. DispCancelF().

To retrieve the cancelled vote commitment from the database table tblCancelVote and display the records in the cancelled votes list box.

11. Form\_Load().

To load the form and perform procedure **ResetField** (see item 19).

12. GetCancel(listText As String).

To retrieve the cancelled P/O record from the database tables tblCancelPO and tblPO. Displays the detailed information in the dialog box.

13. GetPO(listText As String).

To retrieve the P/O record from the database table tblPO. Displays the detailed information in the dialog box.

14. lstFunds\_Click().

To display the detailed information of the selected vote commitment in the text boxes and set the **'Select'** button to be visible.

15. lstPO\_Click().

To perform procedure **GetPO** (see item 13) or **GetCancel** (see item 12) based on the selected menu item.

16. lstVoteCl\_Click().

To set the **'Delete'** button to be visible when the selected menu item is either **Add** or **Edit**.

17. MenuItem(Index As Integer).

To load the form.

18. mnuMItem\_Click(Index As String).

To perform procedure **MenuItem** (see item 17).

19. ResetField().

To reset each entry in the dialog box.

20. SaveCancelPO().

Performs procedure **validEntry** (see item 31). If the entries are valid, performs procedure **SaveCancelVote** (see item 21) and inserts record in the database table tblCancelPO.

21. SaveCancelVote() As Integer.

To update the database tables tblPO, tblVoteCommit, tblVoteFunds, tblBudget and cancelVote when the selected menu item is **Add**.

22. txtAmt\_LostFocus().

To perform procedure **CalAmount** (see Section 4.37.1 item 2) when this entry is valid.

23. txtPercent\_LostFocus().

To set the **'Select'** button to be visible when the entered percent is greater than 0.

24. txtRemark\_KeyPress(Index As Integer, KeyAscii As Integer).

To reset the cursor at other text box after pressing the **'Enter'** key.

25. txtRemark\_LostFocus(Index As Integer).

To convert the character **"'"** to **"`"** of the remarks entered in the text box by using the procedure **ConvertString** (see Section 4.34.1 item 7).

26. txtSAmt\_LostFocus().

To validate the entry of the cancelled amount in S$ equivalent.

27. txtVoteCl\_LostFocus().

To validate the entry of the cancelled amount of a vote and compute the cancelled amount in S$ equivalent amount.

28. UpdBudget(voteAC As String, voteNo As String, szStr As String) As Integer.

To update the database table tblBudget.

29. UpdCancelPO().

Perform procedure **validEntry** (see item 31). If the entries are valid, perform procedure **UpdCancelVote** (see item 30) and update the database table tblCancelPO.

30. UpdCancelVote().

To update the database tables tblPO, tblVoteCommit, tblVoteFunds, tblBudget and cancelVote.

31. validEntry().

To validate the data entered in the text boxes.

32. VisibleItem().

To set some of the item to be visible or invisible based on the selected menu item.

4.5 ClaimFrm Form.

The form is used for user to input/update the claims/expenses payment that are to be reimbursed to the staff member.

4.5.1 Procedures :

1. CanCancel(Index As Integer).

To check whether the user wishes to update the record before exiting the function. If it is, perform procedure **SaveClaim** (see item 31) or **UpdateClaim** (see item 40) based on the selected menu item.

2. CheckEntry().

To perform procedure **validFunds** (see Section 4.34.1 item 46).

3. cmdAdd\_Click().

To perform procedure **CheckFund** (see Section 4.37.1 item 5) and insert the vote commitment into the funds committed list box. It also performs procedure **ResetAddCmd** (see Section 4.37.1 item 10) to reset the **'Add'** and **'Delete'** buttons.

4. cmdCancel\_Click().

To perform the procedure **CanCancel** (see item 1).

5. cmdDelete\_Click().

To remove the selected vote commitment from the funds committed list box when the **'Delete'** button is pressed.

6. cmdList\_Click().

To load the DateFrm form for user to select a range of records to be retrieved. Perform procedure **LoadRec** (see Section 4.43.1 item 24) to load the database table tblClaim.

7. cmdOK\_Click().

To perform procedure **SaveClaim** (see item 20), **UpdateClaim** (see item 40) or **DeleteClaim** (see item 8) based on the selected menu item.

8. DeleteClaim().

To perform procedures **WrtPrevBudget** (see Section 4.36.1 item 16) and **updateVoteFunds** (see Section 4.37.1 item 13). Delete record in the database table tblClaim.

9. DisableItem().

To disable some entries in the dialog box.

10. EnableItem().

To perform procedure **LoadCurrency** (see Section 4.34.1 item 22) and **LoadCostC** (see Section 4.34.1 item 21) to load the database tables tblCurrency and tblCostC respectively when the value of variable ***listLoaded*** is False. Disable some disabled entries in the dialog box.

11. Form\_Activate().

To set the value of variable ***listLoaded*** to False.

12. Form\_Load().

To load the form and perform procedure **ResetField** (see item 19).

13. GetClaim(listText As String).

To retrieve the claims record from the database tables tblClaim and display the detailed information in the dialog box.

14. lstClaimRef\_Click().

To perform procedure **GetClaim** (see item 13) when a record is selected from the Claims list box.

15. lstCostC\_Click().

To display a list of votes of a selected cost centre in the Vote list box.

16. lstCurrList\_Click().

To perform procedure **Get\_ExchRate** (see Section 4.34.1 item 12) to get the exchange rate of a selected currency.

17. lstFunds\_Click().

To set the **'Delete'** button to be visible and enabled.

18. lstVoteList\_Click().

To perform procedure **GetPercent** (see Section 4.37.1 item 9) to compute the percentage and set the **'Delete'** button to be disabled.

19. MenuItem(Index As Integer).

To perform procedures **EnableItem** (see item 10) and **DisableItem** (see item 9) based on the selected menu item.

20. mnuMItem\_Click(Index As String).

To perform procedure **MenuItem** (see item 19).

21. ResetField().

To reset each entry in the dialog box.

22. SaveClaim().

To perform procedure **CheckEntry** (see item 2). If the entries are valid, perform procedure **SaveVoteAC** (see Section 4.37.1 item 12) and insert record in the database table tblClaim.

23. txtAmt\_KeyPress(KeyAscii As Integer).

To set the cursor at cost centre list box.

24. txtAmt\_LostFocus().

To perform procedure **CalAmount** (see Section 4.37.1 item 2) when this entry is valid.

25. txtCommit\_GotFocus().

To set the **'Add'** button to be visible and **'OK'**, **'Delete'** buttons to be invisible.

26. txtCommit\_LostFocus().

To perform procedure **CalPercent** (see Section 4.37.1 item 4) if this entry is valid.

27. txtDate\_KeyPress(KeyAscii As Integer).

To set the cursor at the **Amount** text box.

28. txtDate\_LostFocus().

To perform procedure **DispFinYrMth** (see Section 4.34.1 item 9) when the date is valid.

29. txtExch\_KeyPress(KeyAscii As Integer).

To reset the cursor at other text box after pressing the **'Enter'** key.

30. txtExch\_LostFocus().

To perform procedure **CalAmount** (see Section 4.37.1 item 2) when this entry is valid.

31. txtFinMth\_LostFocus().

To perform procedure **CheckFinMth** (see Section 4.34.1 item 3).

32. txtFinYr\_KeyPress(KeyAscii As Integer).

To set the cursor at the financial month text box.

33. txtFinYr\_LostFocus().

To perform procedure **CheckFinMth** (see Section 4.34.1 item 3).

34. txtPercent\_GotFocus().

To set the **'Delete'** button to be invisible and **'Add'** button to be visible.

35. txtPercent\_LostFocus().

To perform procedure **GetPercent** (see Section 4.37.1 item 9).

36. txtRemark\_KeyPress(KeyAscii As Integer).

To set the cursor at another remark text box.

37. txtRemark\_LostFocus(Index As Integer).

To convert the character **"'"** to **"`"** of the remarks entered in the text box by using the procedure **ConvertString** (see Section 4.34.1 item 7).

38. txtSAmt\_LostFocus().

To validate the entry of the claim amount in S$ equivalent.

39. txtSFunds\_LostFocus().

To validate the entry of the funds committed amount in S$ equivalent.

40. UpdateClaim().

To performs procedure **CheckEntry** (see item 2). If the entries are valid, perform procedure **SaveVoteAC** (see Section 4.37.1 item 12), **updateVoteFunds** (see Section 4.37.1 item 13) and update record in the database table tblClaim.

4.6 CostCFrm Form.

The form is used for maintaining the cost centres of the accounts.

4.6.1 Procedures :

1. AddNewVote() As Integer).

To insert a vote record that the vote number is the same as cost centre account, in the database table tblVote. This procedure will be performed when the **'Special'** check box is checked.

2. CanCancel(Index As Integer).

To check whether the user wishes to update the record before exiting the function. If it is, perform procedure **SaveTable** (see item 19).

3. ChkSpec\_Click(Value As Integer).

To set the value of variable ***szSpec*** based on the selection of the Special check box.

4. cmdCancel\_Click().

To perform the procedures **CanCancel** (see item 2) and **ResetField** (see item 18).

5. cmdOK\_Click().

To perform procedure **SaveTable** (see item 19) or **DeleteTable** (see item 6) based on the selected menu item.

6. DeleteTable().

To delete record in the database table tblCostC.

7. DisableItem().

To disable some entries in the dialog box.

8. EnableItem().

To enable some disabled entries in the dialog box.

9. Form\_Activate().

To perform procedure **LoadCostC** (see Section 4.34.1 item 21) to load the records from database table tblCostC.

10. Form\_Load().

To load the form and perform procedure **MenuItem** (see item 14).

11. GetAll(szAcNo As String).

To retrieve a selected cost centre record from the database table tblCostC and display the detailed information in the dialog box.

12. GetList().

To load the cost centre records from the database table tblCostC and display them in the cost centre list box.

13. lstCostC\_DblClick().

To perform procedure **GetAll** (see item 11) to retrieve the selected cost centre record.

14. MenuItem(Index As Integer).

To perform procedures **EnableItem** (see item 8) or **DisableItem** (see item 7) based on the selected menu item.

15. mnuMItem\_Click(Index As String).

To perform procedure **MenuItem** (see item 14).

16. NewEntry() As Integer.

To check whether the entered cost centre account already exists in the database table tblCostC.

17. OptCCType\_Click(Index As Integer, Value As Integer).

To set the value of variable ***szCCType*** based on the selection of the Cost Centre type.

18. ResetField().

To reset each entry in the dialog box.

19. SaveTable().

To perform procedures **validEntry** (see item 22) and **NewEntry** (see item 16). If the entries are valid, inserts record in the database table tblCostC. It also performs procedure **AddNewVote** (see item 1) if the special check box is checked.

20. txtAbbr\_LostFocus().

To convert the character **"'"** to **"`"** of the remarks entered in the text box by using the procedure **ConvertString** (see Section 4.34.1 item 7).

21. txtCDesc\_LostFocus().

To convert the character **"'"** to **"`"** of the remarks entered in the text box by using the procedure **ConvertString** (see Section 4.34.1 item 7).

22. valid Entry().

To validate the entries in the dialog box.

4.7 CredInv Form.

This form is used for user to input the invoices received from suppliers. Some received invoices may have purchare orders issued by the user.

4.7.1 Procedures :

1. CanCancel(Index As Integer).

To check whether the user wishes to update the record before exiting the function. If it is, performs procedures **SaveInv** (see item 24) or **DeleteInv** (see item 7) based on the selected menu item.

2. cmdAdd\_Click().

To perform procedure **CheckFund** (see Section 4.37.1 item 5) and insert the vote commitment into the funds committed list box. It also performs procedure **ResetAddCmd** (see Section 4.37 item 10) to reset the **'Add'** and **'Delete'** buttons.

3. cmdCancel\_Click().

To perform the procedure **CanCancel** (see item 1).

4. cmdDelete\_Click().

To remove the selected vote commitment from the funds committed list box when the **'Delete'** button is pressed.

5. cmdList\_Click().

To load the DateFrm form for user to select a range of records to be retrieved. Performs procedure **LoadRec** (see Section 4.43.1 item 24) to load the database table tblInvRecd.

6. cmdOK\_Click().

To perform procedure **SaveInv** (see item 24), **EditInv** (see item 9) or **DeleteInv** (see item 7) based on the selected menu item.

7. DeleteInv().

To perform procedures **WrtPrevBudget** (see Section 4.36.1 item 16), **wrtDelInv** (see Section 4.34.1 item 49) and **updateVoteFunds** (see Section 4.37.1 item 13)as well as delete record in the database table tblInvRecd.

8. DisableItem().

To disable some entries in the dialog box.

9. EditInv().

To perform procedure **validEntry** (see Section 4.34.1 item 45). If the entries are valid, then updates the database table tblnvRecd. It also performs procedures **WrtPrevBudget** (see Section 4.36.1 item 16), **updateVoteFunds** (see Section 4.37.1 item 13) and **SaveVoteAC (**see Section 4.37.1 item12).

10. EnableItem().

To enable some disabled entries in the dialog box. It also performs procedures **LoadCurrency** (see Section 4.34.1 item 22), **LoadSupplier** (see Section 4.43.1 item 27) and **LoadCostC** (see Section 4.34.1 item 21) to load the records from the database tables tblCurrency, Supplier and tblCostC respectively.

11. Form\_Load().

To load the form and perform procedure **MenuItem** (see item 20).

12. GetInv\_Ref(szRef As String).

To retrieve the invoice record from the database table tblInvRecd with the entered reference number and display the detailed information in the dialog box.

13. GetInvoice(listText As String).

To retrieve a selected invoice record from the database table tblInvRecd and display the detailed information in the dialog box.

14. lstCostC\_Click().

To perform procedure **LoadSelVote** (see Section 4.34.1 item 25) and display a list of votes of a selected cost centre in the Vote list box.

15. lstCurrList\_Click().

To perform procedure **Get\_ExchRate** (see Section 4.34.1 item 12) to get the exchange rate of a selected currency.

16. lstFunds\_Click().

To set the **'Delete'** button to be visible.

17. lstInvList\_Click().

To perform procedure **GetInvoice** (see item 13) to retrieve the selected invoice record.

18. lstSupplier\_Click().

To perform procedure **GetSupplier** (see Section 4.34.1 item 18) to display the detailed information of the select supplier.

19. lstVoteList\_Click().

To perform procedure **GetPercent** (see Section 4.37.1 item 9) to compute the percentage and set the **'Add'** button to be enabled.

20. MenuItem(Index As Integer).

To perform procedures **EnableItem** (see item 10) or **DisableItem** (see item 8) based on the selected menu item.

21. mnuMItem\_Click(Index As String).

To perform procedure **MenuItem** (see item 20).

22. NewEntry() As Integer.

To check whether the entered invoice already exists in the database table tblInvRecd.

23. ResetField().

To reset each entry in the dialog box.

24. SaveInv().

To perform procedures **validEntry** (see Section 4.34.1 item 45) and **NewEntry** (see item 22). If the entries are valid, performs procedure **UpdRefNo** (see Section 4.34.1 item 43) and **SaveVoteAC** (see Section 4.37.1 item 12) as well as inserts record in the database table tblInvRecd.

25. txtAmt\_KeyPress(KeyAscii As Integer).

To set the cursor at the date sent to Bursar text box.

26. txtAmt\_LostFocus().

To perform procedure **CalAmount** (see Section 4.37.1 item 2) when this entry is valid.

27. txtCommit\_GotFocus().

To set the **'Add'** button to be visible and **'OK'**, **'Delete'** buttons to be invisible.

28. txtCommit\_LostFocus().

To perform procedure **CalPercent** (see Section 4.37.1 item 4) if this entry is valid.

29. txtExch\_LostFocus().

To perform procedure **CalAmount** (see Section 4.37.1 item 2) and **CalFunds** (see Section 4.37.1 item 3) when this entry is valid.

30. txtFinMth\_LostFocus().

To perform procedure **CheckFinMth** (see Section 4.34.1 item 3).

31. txtFinYr\_LostFocus().

To perform procedure **CheckFinMth** (see Section 4.34.1 item 3).

32. txtInvDate\_LostFocus().

To perform procedure **ValidDate** (see Section 4.34.1 item 44).

33. txtInvNo\_LostFocus().

To convert the invoice number to upper case.

34. txtPercent\_GotFocus().

To set the **'Delete'** button to be invisible and **'Add'** button to be visible.

35. txtPercent\_LostFocus().

To perform procedure **GetPercent** (see Section 4.37.1 item 9).

36. txtReason\_LostFocus().

To convert the character **"'"** to **"`"** of the reason entered in the text box by using the procedure **ConvertString** (see Section 4.34.1 item 7).

37. txtRef\_KeyPress(KeyAscii As Integer).

To perform the procedure **GetInv\_Ref** (see item 12).

38. txtRemark\_LostFocus().

To convert the character **"'"** to **"`"** of the remarks entered in the text box by using the procedure **ConvertString** (see Section 4.34.1 item 7).

39. txtSAmt\_LostFocus().

To perform procedure **CheckNumeric** (see Section 4.34.1 item 5).

40. txtSent\_LostFocus().

To perform procedure **ValidDate** (see Section 4.34.1 item 44).

41. txtSFunds\_LostFocus().

To perform procedure **ValidDate** (see Section 4.34.1 item 44).

4.8 DateFrm Form.

The form is used for user to select the necessary conditions for viewing or printing a report.

4.8.1 Procedures :

1. CheckSort(Index As Integer).

To perform procedures **LoadStaff** (see Section 4.43.1 item 26) or **LoadCostC** (see Section 4.34.1 item 21) based on the selection of the sort order.

2. chkGroup\_Click(Value As Integer).

To set the value of the global variable ***gszSelection*** based on the selection of the **'All Programs Group'** check box.

3. chkPO\_Click(Value As Integer).

To set the value of the global variable ***gszSelection*** based on the selection of the P/O check box.

4. cmdCancel\_Click().

To set the value of global variable ***gnOption*** to -2 and close the dialog box.

5. cmdOK\_Click().

To set the value for the global variables ***gnOption***, ***gnSTravel***, ***gszDateFrm*** and ***gszDateTo*** before closing the dialog box.

6. Form\_Load().

To load the form and reformat it based on the value of the global variable ***gszMenuItem***.

7. mskDateFrom\_LostFocus().

To perform procedure **CheckFinMth** (see Section 4.34.1 item 3).

8. mskDateTo\_LostFocus().

To perform procedure **CheckFinMth** (see Section 4.34.1 item 3).

9. optSort\_Click(Index As Integer, Value As Integer).

To perform procedure **CheckSort** (see item 1).

10. SelectGroup().

To transfer all the selected cost centres from the cost centre list box to the vote list box in ReportFrm (see Section 4.24) dialog box, if the **'All Programs Group'** check box is unchecked.

11. txtFinYr\_LostFocus().

To perform procedure **CheckFinYear** (see Section 4.34.1 item 4).

4.9 DebitNote Form.

The form is used for entering the credit or debit notes received from the suppliers.

4.9.1 Procedures :

1. CanCancel(Index As Integer).

To check whether the user wishes to update the record before exiting the function. If it is, performs procedures **SaveCDNote** (see item 22) or **UpdCDNote** (see item 32) based on the selected menu item.

2. cmdCancel\_Click().

To perform the procedure **CanCancel** (see item 1).

3. cmdList\_Click().

To load the DateFrm form for user to select a range of records to be retrieved. Performs procedure **LoadRec** (see Section 4.43.1 item 24) to load the database table tblInvRecd.

4. cmdOK\_Click().

To perform procedure **SaveCDNote** (see item 22), **UpdCDNote** (see item 32) or **DelCDNote** (see item 6) based on the selected menu item.

5. cmdUpdate\_Click().

To re-update the vote commitment based on the entered adjustment amount.

6. DelCDNote().

To perform procedures **UpdInvoice** (see item 33), **DelVoteFunds** (see item 7) and **WrtDelRec** (see item 36) as well as delete the record from the database table tblCDNote.

7. DelVoteFunds(formNo As String) As Integer.

To update the database tables tblVoteFunds and delete records from database table tblVoteCommit. It also performs procedures **UpdBudget** and (see item 31) and **UpdCommitInv** (see Section 4.37.1 item 14).

8. DispAdjFunds().

To perform the procedure **AdjustedFunds** (see Section 4.37 item 1) to compute the adjusted vote commitment based on the credit/debit amount and display the adjusted commitment in the funds commitment list box.

9. Form\_Load().

To load the form.

10. GetCDNote().

To retrieve the selected record from the database table tblCDNote and display detailed information in the dialog box.

11. GetInvoice().

To retrieve the selected invoice from the database table tblInvRecd and display detailed information in the dialog box.

12. LoadData(szCmd As String).

To load the records from database table tblInvRecd or tblCDNote based on the given SQL command string szCmd. The records are displayed in the Invoice list box.

13. lstCurrList\_Click().

To perform procedure **Get\_ExchRate** (see Section 4.34.1 item 12) to get the exchange rate of a selected currency.

14. lstFunds\_DblClick().

To display the detailed information of a selected vote commitment in the text boxes.

15. lstInvList\_Click().

To perform procedure **GetInvoice** (see item 11) or **GetCDNote** (see item 10) to retrieve the selected record based on the selected menu item.

16. MenuItem(Index As Integer).

To perform procedures **ResetListBox** (see item 21).

17. mnuMItem\_Click(Index As Integer).

To perform procedure **MenuItem** (see item 16).

18. NewEntry() As Integer.

To check whether the entered credit/debit note record already exists in the database table tblCDNote.

19. ResetField().

To reset each entry in the dialog box.

20. ResetItem().

To set some entries to be visible/invisible based on the selected menu item.

21. ResetListBox().

To load the records from the database table tblInvRecd or tblCDNote in the Invoice list box based on the selected menu item.

22. SaveCDNote().

To perform procedures **validEntry** (see item 35) and **NewEntry** (see item 18). If the entries are valid, inserts record in the database table tblCDNote and performs procedures **UpdInvoice** (see item 33) and **SaveVoteAC** (see Section 4.37.1 item 12).

23. txtAmt\_LostFocus().

To perform procedures **CalAmount** (see Section 4.37.1 item 2) and **DispAdjFunds** (see item 8) when this entry is valid.

24. txtCommit\_LostFocus().

To compute the committed amount in S$ equivalent if this entry is valid.

25. txtDbDate\_LostFocus().

To perform procedure **ValidDate** (see Section 4.34.1 item 44) if this entry is valid.

26. txtDebit\_LostFocus().

To convert the credit/debit note number to upper case.

27. txtExch\_LostFocus().

To perform procedures **CalAmount** (see Section 4.37.1 item 2) and **DispAdjFunds** (see item 8) when this entry is valid.

28. txtReason\_LostFocus().

To convert the character **"'"** to **"`"** of the reason entered in the text box by using the procedure **ConvertString** (see Section 4.34.1 item 7).

29. txtRemark\_LostFocus().

To convert the character **"'"** to **"`"** of the remarks entered in the text box by using the procedure **ConvertString** (see Section 4.34.1 item 7).

30. txtSAmt\_LostFocus().

To perform procedures **CheckNumeric** (see Section 4.34 item 5) and **DispAdjFunds** (see item 8).

31. UpdBudget(formNo As String, voteAC As String, voteNo As String, szStr As String) As Integer.

To update the database table tblBudget.

32. UpdCdNote().

To update the database table tblCDNote and perform procedures **UpdInvoice** (see item 33) and **UpdVoteFunds** (see item 34).

33. UpdInvoice(amt\_before As Double, amt\_after As Double, amtS\_before As Double, amtS\_after As Double) As Integer.

To update the database table tblInvRecd.

34. UpdVoteFunds(formNo As String) As Integer.

To update the database tables tblVoteFunds and tblVoteCommit. It also performs procedures **UpdBudget** (see item 31) and **UpdCommitInv** (see Section 4.37.1 item 14).

35. validEntry() As Integer.

To validate the entries in the dialog box.

36. WrtDelRec() As Integer.

To insert a deleted credit/debit note record in the database table delCDNote.

4.10 ExpPymt Form.

The form is used for entering the credit or debit notes received from the suppliers.

4.10.1 Procedures :

1. CanCancel(Index As Integer).

To check whether the user wishes to update the record before exiting the function. If it is, performs procedure **UpdatePymt** (see item 20).

2. CheckEntry().

To perform procedure **ValidPaidAmt** (see Section 4.36.1 item 14) and validate other entries in the dialog box.

3. cmdCancel\_Click().

To perform the procedure **CanCancel** (see item 1).

4. cmdList\_Click().

To load the DateFrm form for user to select a range of records to be retrieved. Perform procedure **LoadPayment** (see item 10) to load the database table tblClaim.

5. cmdOK\_Click().

To perform procedure **UpdatePymt** (see item 20) when the selected menu item is 'Update'.

6. DisableItem().

To set some enabled entries to be disabled.

7. EnableItem().

To set some disabled entries to be enabled.

8. Form\_Load().

To load the form.

9. GetPayment(listText As String).

To retrieve the selected record from the database table tblPettyCash or tblClaim based on the selected type of record and display detailed information in the dialog box.

10. LoadPayment(whereCl As String).

To load the records from database tables tblPettyCash and tblClaim. The records are displayed in the Expenses list box.

11. lstInvList\_Click().

To perform procedure **GetPayment** (see item 9).

12. MenuItem(Index As Integer).

To perform procedures **DisableItem** (see item 6) or **EnableItem** (see item 7) based on the selected menu item.

13. mnuMItem\_Click(Index As Integer).

To perform procedure **MenuItem** (see item 12).

14. ResetField().

To reset each entry in the dialog box.

15. ResetItem().

To set some entries to be visible/invisible based on the selected menu item.

16. SetPaidAmt().

To perform procedure **ValidPaidAmt** (see Section 4.36.1 item 14) to validate the paid amount. If it is valid, then performs procedure **CalVoteAmt** (see Section 4.36.1 item 2).

17. txtPaidDate\_LostFocus().

To perform procedure **ValidPaidDate** (see Section 4.34.1 item 47) if this entry is valid.

18. txtRemark\_LostFocus().

To convert the character **"'"** to **"`"** of the remarks entered in the text box by using the procedure **ConvertString** (see Section 4.34.1 item 7).

19. txtSPaidAmt\_LostFocus().

To perform procedures **SetPaidAmt** (see item 16).

20. UpdatePymt().

To perform procedure **Valid\_Pymt** (see Section 4.36.1 item 13) to validate the payment date and paid amount. If they are valid, then updates the database table tblClaim and performs procedure **UpdFundsPymt** (see Section 4.37.1 item 16).

4.11 ExchRFrm Form.

The form is used for maintaining the exchange rate of currency.

4.11.1 Procedures :

1. CanCancel(Index As Integer).

To check whether the user wishes to update the record before exiting the function. If it is, performs procedure **SaveTable** (see item 19).

2. cmdCancel\_Click().

To perform the procedure **CanCancel** (see item 1).

3. cmdOK\_Click().

To perform procedure **SaveTable** (see item 19) or **DeleteTable** (see item 4) based on the selected menu item.

4. DeleteTable().

To delete the selected record from the database table tblCurrency.

5. DisableItem().

To set some enabled entries to be disabled.

6. EnableItem().

To set some disabled entries to be enabled.

7. ExchHeader(aRemark As String).

To print the report header of the Currency Exchange Rate report.

8. Form\_Activate().

To load the records from the database table tblCurrency in the Currency list box.

9. Form\_Load().

To load the form and perform procedure **MenuItem** (see item 14).

10. Form\_QueryUnload().

To perform procedure **CanCancel** (see item 1).

11. GetAll(szCurrCode As String).

To retrieve the selected record from the database table tblCurrency.

12. GetList().

To load the records from the database table tblCurrency in the Currency list box.

13. lstCurrList\_DblClick().

To perform procedure **GetAll** (see item 11).

14. MenuItem(Index As Integer, currForm As Form).

To perform procedures **DisableItem** (see item 5), **EnableItem** (see item 6) or **PrintExch** (see item 17) based on the selected menu item.

15. mnuMItem\_Click(Index As Integer).

To perform procedure **MenuItem** (see item 14).

16. NexEntry() As Integer.

To check whether the record already exits in the database table tblCurrency.

17. PrintExch().

To print the Curreny Exchange Rate report.

18. ResetField().

To reset each entry in the dialog box.

19. SaveTable().

To perform procedures **validEntry** (see item 23) and **NewEntry** (see item 16). If the entries are valid, updates the record in database table tblCurrency.

20. txtCurrCode\_LostFocus().

To convert the currency code to upper case.

21. txtDesc\_LostFocus().

To convert the character **"'"** to **"`"** of the description entered in the text box by using the procedure **ConvertString** (see Section 4.34.1 item 7).

22. txtExch\_LostFocus().

To perform procedures **CheckNumeric** (see Section 4.34.1 item 5).

23. validEntry() As Integer.

To validate all the entries shown in the dialog box.

4.12 FundTsf Form.

The form is used for authorised user to transfer funds between votes.

4.12.1 Procedures :

1. CanCancel(Index As Integer).

To check whether the user wishes to update the record before exiting the function. If it is, performs procedures **ValidEntry** (see item 24) and **SaveTrsf** (see item 18).

2. CheckVote().

To perform procedure **GetVoteBal** (see item 9) to get the current balance of the vote.

3. cmdAdd\_Click().

To add the vote description and transferred amount into the funds transferred list box.

4. cmdCancel\_Click().

To perform the procedure **CanCancel** (see item 1).

5. cmdDelete\_Click().

To remove the selected vote from the funds transferred list box.

6. cmdOK\_Click().

To perform procedure **ValidEntry** (see item 24). If the entries are valid, performs procedure **SaveTrsf** (see item 18) to update the database table tblVoteFunds.

7. Form\_Load().

To load the form.

8. GetCC\_Budget(aVoteIndex As Integer).

To retrieve the budget amount of a cost centre from the database table tblBudget. It also checks the amount transferred whether it excees a certain percentage of the budget amount.

9. GetVoteBal(aVoteIndex As Integer).

To retrieve the current balance of a vote from the database table tblVoteFunds and compute the balance after transferred. It also performs procedure **GetCC\_Budget** (see item 8).

10. lstCostC\_Click().

To perform procedure **LoadSelVote** (see Section 4.43.1 item 25) to display all votes of the selected cost centre in the Vote list box.

11. lstSelVote\_Click(Index As Integer).

To set **Delete** button to be visible.

12. lstVoteList\_Click().

To check whether the selected vote has been selected previously. If it is not, display the vote description in the vote text box.

13. MenuItem(Index As Integer).

To perform procedures **LoadCostC** (see Section 4.34.1 item 21), **Display\_Trsf** (see Section 4.36.1 item 3) or **Print\_Trsf** (see Section 4.36.1 item 8) based on the selected menu item.

14. mnuMItem\_Click(Index As Integer).

To perform procedure **MenuItem** (see item 13).

15. mskDate\_LostFocus().

To perform procedure **ValidDate** (see Section 4.34.1 item 44). If the date is valid, performs procedure **DispFinYrMth** (see Section 4.34.1 item 9).

16. ResetField().

To reset each entry in the dialog box.

17. ResetGrid(Index As Integer).

To reformat the form based on the selected menu item.

18. SaveTrsf().

To perform procedures **GetMaxCode** (see Section 4.34.1 item 15) and **WrtFundsTrsf** (see item 25).

19. txtAmt\_LostFocus(Index As Integer).

To perform procedure **CheckNumeric** (see Section 4.34 item 5) if this entry is valid, then performs procedure **CheckVote** (see item 2).

20. txtFinMth\_LostFocus().

To perform procedure **CheckFinMth** (see Section 4.34.1 item 3).

21. txtFinYr\_LostFocus().

To perform procedure **CheckFinYear** (see Section 4.34.1 item 4).

22. txtRemark\_LostFocus(Index As Integer).

To convert the character **"'"** to **"`"** of the remarks entered in the text box by using the procedure **ConvertString** (see Section 4.34.1 item 7).

23. UpdFunds(voteAmt As Double) As Integer.

To update the records in database table tblVoteFunds and tblBudget.

24. ValidEntry() As Integer.

To validate each entry in the dialog box.

25. WrtFundsTrsf(seq As Integer, item As Integer, amt As Double) As Integer.

To insert record in the databse table tblJAdjM and perform procedure **UpdFunds** (see item 23).

4.13 InvoiceFrm Form.

The form is used for authorised user to transfer funds between votes.

4.13.1 Procedures :

1. calInv\_Funds() As Integer.

To compute the funds commitment of the invoice based on the invoice amount (this is applicable for invoice with P/O only). It returns **False** when the invoice amount or the S$ equivalent amount is equal to zero.

2. CanCancel(Index As Integer).

To check whether the user wishes to update the record before exiting the function. If it is, performs procedures **SaveInv** (see item 24) or **DeleteInv** (see item 18) based on the selected menu item.

3. CheckEntry().

To validate the entries shown in the dialog box. If they are valid, performs procedures **calInv\_Funds** (see item 1) and **DispFunds** (see item 13).

4. CheckFunds().

To perform procedure **AdjustedFunds** (see Section 4.37.1 item 1) if the funds have not been updated.

5. chkFinal\_Click().

To set the value of the variable ***szFinal*** based on the selection of the **Final Invoice** check box .

6. cmdCancel\_Click().

To perform the procedure **CanCancel** (see item 2).

7. cmdFunds\_Click().

To perform procedure **calInv\_Funds** (see item 1). If the returned value is True, performs procedure **DispFunds** (see item 13).

8. cmdList\_Click().

To load the DateFrm form for user to select a range of records to be retrieved. Performs procedure **LoadRec** (see Section 4.43.1 item 24) to load the database table tblInvRecd or tblPO based on the selected menu item.

9. cmdOK\_Click().

To perform procedures **SaveInv** (see item 24), **EditInv** (see item 14) or **DeleteInv** (see item 10) to update the database table tblInvRecd based on the selected menu item.

10. DeleteInv().

To delete the record from the database table tblInvRecd and perform procedures **WrtDelInv** (see Section 4.34.1 item 49), **updateVoteFunds** (see Section 4.37.1 item 13) and **DelVoteAC** (see Section 4.37.1 item 6).

11. DisableFunds().

To disable some list boxes.

12. DisableItem().

To disable some enabled entries.

13. DispFunds().

To display the adjusted funds commitment in the funds committed list box.

14. EditInv().

If the received invoice is for a P/O, performs procedure **CheckFunds** (see item 4). Performs procedure **CheckEntry** (see item 3) to validate the entries. If the entries are invalid, performs **RestoreFunds** (see item 29). Otherwise, updates the record in database table tblInvRecd and performs procedures **WrtPrevBudget** (see Section 4.36.1 item 16), **updateVoteFunds** (see Section 4.37.1 item 13) and **UpdVoteAC** (see Section 4.37.1 item 18).

15. EnableFunds().

To perform procedure **LoadCurrency** (see Section 4.34.1 item 22) and enable some list boxes.

16. EnableItem().

To enable some text boxes.

17. Form\_Load().

To load the form.

18. Getn\_VoteArray().

To store the funds commitment of the selected P/O in the array variable ***voteArray***.

19. GetInv\_Ref(szRef As String).

To retrieve the Invoice from the database table tblInvRecd by the entered reference number.

20. GetInvAmt(poAmt As Double).

To retrieve the total invoice amount of a selected P/O from the database table tblInvRecd. It also performs procedure **CalAmount** (see Section 4.37.1 item 2).

21. GetInvoice(listText As String).

To retrieve the selected Invoice from the database table tblInvRecd.

22. GetPO(listText As String).

To retrieve the selected P/O from the database table tblPO.

23. lstCurrList\_Click().

To perform procedure **Get\_ExchRate** (see Section 4.34.1 item 12) to get the exchange rate of the selected currency.

24. lstInvList\_Click().

To perform procedure **GetPO** (see item 22) or **GetInvoice** (see item 21) based on the selected menu item.

25. MenuItem(Index As Integer).

To perform procedures **DisableItem** (see item 12) or **EnableItem** (see item 16) based on the selected menu item.

26. mnuMItem\_Click(Index As Integer).

To perform procedure **MenuItem** (see item 25).

27. NewEntry() As Integer.

To check whether the record already exists in the database table tblInvRecd.

28. ResetField().

To reset each entry in the dialog box.

29. RestoreFunds().

To the funds commitment the array variable ***voteArray*** (see item 18) and display it in the funds commitment list box.

30. SaveInv().

To perform procedures **CheckEntry** (see item 3) and **NewEntry** (see item 27). If the record is valid, performs procedures **UpdRefNo** (see Section 4.34.1 item 43), **SaveVoteAC** (see Section 4.37.1 item 12) and inserts the record in database table tblInvRecd.

31. txtAmt\_LostFocus(Index As Integer).

To perform procedure **CheckNumeric** (see Section 4.34.1 item 5). If the amount is valid then performs procedure **CalAmount** (see Section 4.37.1 item 2).

32. txtExch\_LostFocus(Index As Integer).

To perform procedure **CheckNumeric** (see Section 4.34 item 5). If the amount is valid then performs procedure **CalAmount** (see Section 4.37.1 item 2).

33. txtInvDate\_LostFocus().

To perform procedure **ValidDate** (see Section 4.34.1 item 44) to validate the date entered.

34. txtInvNo\_LostFocus().

To convert the Invoice number to upper case.

35. txtPONo\_LostFocus().

To perform procedure **GetPO** (see item 22).

36. txtReason\_LostFocus().

To convert the character **"'"** to **"`"** of the reason entered in the text box by using the procedure **ConvertString** (see Section 4.34.1 item 7).

37. txtRef\_KeyPress(KeyAscii As Integer).

To perform procedure **GetInv\_Ref** (see item 19) when the selected menu item is **View**.

38. txtRemark\_LostFocus(Index As Integer).

To convert the character **"'"** to **"`"** of the remarks entered in the text box by using the procedure **ConvertString** (see Section 4.34.1 item 7).

39. txtSent\_LostFocus().

To perform procedure **ValidDate** (see Section 4.34.1 item 44) to validate the date entered.

4.14 InvPymt Form.

The form is used for user to update the cash payment of invoice.

4.14.1 Procedures :

1. CanCancel(Index As Integer).

To check whether the user wishes to update the record before exiting the function. If it is, performs procedures **UpdatePymt** (see item 20).

2. CheckEntry().

To perform procedure **ValidPaidAmt** (see Section 4.36.1 item 14) and validate other entries in the dialog box.

3. cmdCancel\_Click().

To perform the procedure **CanCancel** (see item 1).

4. cmdList\_Click().

To load the DateFrm form for user to select a range of records to be retrieved. Performs procedure **LoadPayment** (see item 10) to load the database table tblClaim.

5. cmdOK\_Click().

To perform procedure **UpdatePymt** (see item 20) when the selected menu item is Update.

6. DisableItem().

To set some enabled entries to be disabled.

7. EnableItem().

To set some disabled entries to be enabled.

8. Form\_Load().

To load the form.

9. GetPayment(listText As String).

To retrieve the selected record from the database table tblInvRecd or tblPO based on the selected type of record and display detailed information in the dialog box.

10. LoadPayment(whereCl As String).

To load the records from database tables tblInvRecd and tblPO. The records are displayed in the Invoices/POs list box.

11. lstInvList\_Click().

To perform procedure **GetPayment** (see item 9).

12. MenuItem(Index As Integer).

To perform procedures **DisableItem** (see item 6) or **EnableItem** (see item 7) based on the selected menu item.

13. mnuMItem\_Click(Index As Integer).

To perform procedure **MenuItem** (see item 12).

14. ResetField().

To reset each entry in the dialog box.

15. SetPaidAmt().

To perform procedure **ValidPaidAmt** (see Section 4.36.1 item 14) to validate the paid amount. If it is valid, then performs procedure **CalVoteAmt** (see Section 4.36.1 item 2).

16. txtPaidDate\_LostFocus().

To perform procedure **ValidPaidDate** (see Section 4.34.1 item 47) if this entry is valid.

17. txtRemark\_LostFocus().

To convert the character **"'"** to **"`"** of the remarks entered in the text box by using the procedure **ConvertString** (see Section 4.34.1 item 7).

18. txtSPaidAmt\_LostFocus().

To perform procedures **SetPaidAmt** (see item 15).

19. UpdatePO().

To update the record in database table tblPO.

20. UpdatePymt().

To perform procedure **Valid\_Pymt** (see Section 4.36.1 item 2) to validate the payment date and paid amount. If they are valid, then updates the database table tblInvRecd and performs procedures **UpdFundsPymt** (see Section 4.37.1 item 16) & **UpdatePO** (see item 19).

4.15 InvTravel Form.

The form is used for user to update the air fare of the staff travelling records .

4.15.1 Procedures :

1. CheckEntry().

To validate the entries shown in the dialog box.

2. cmdDelete\_Click().

To remove a selected staff travelling record from the Staff Travel list box.

3. cmdList\_Click().

To load the DateFrm form for user to select a range of records to be retrieved. Performs procedure **LoadRec** (see Section 4.43.1 item 24) to load the database table tblInvRecd.

4. cmdOK\_Click().

To perform procedures **SaveInv** (see item 22), **EditInv** (see item 9) or **DeleteInv** (see item 6) to update the database table tblInvRecd based on the selected menu item.

5. cmdOpt\_Click().

To perform procedures **ReSetTravel** (see item 21) and load the TravelList form.

6. DeleteInv().

To delete the record from the database table tblInvRecd and perform procedures **WrtDelInv** (see Section 4.34.1 item 49) and **DelTravel** (see item 7).

7. DelTravel(refNo As Integer) As Integer.

To update the record in database table tblTravel and delete record from the database table tblInvTravel.

8. DisableItem().

To disable some enabled entries.

9. EditInv().

Performs procedure **CheckEntry** (see item 3) to validate the entries. If the entries are valid, updates the record in database table tblInvRecd and performs procedures **DelTravel** (see item 7) and **SaveTravel** (see item 23).

10. EnableItem().

To perform procedure **LoadSupplier** (see Section 4.43.1 item 27) and enable some text boxes.

11. Form\_Load().

To load the form.

12. GetInvoice(listText As String).

To retrieve the selected Invoice from the database table tblInvRecd and perform procedure **GetTravel** (see item 13).

13. GetTravel().

To retrieve the staff travel records of a selected invoice from the database table tblInvTravel.

14. lstInvList\_Click().

To perform procedure **GetInvoice** (see item 12).

15. lstSupplier\_Click().

To perform procedure **GetSupplier** (see Section 4.34.1 item 18).

16. lstTravel\_Click().

To set the **Delete** button to be visible/invisible based on the selected menu item.

17. MenuItem(Index As Integer).

To perform procedures **DisableItem** (see item 8) or **EnableItem** (see item 10) based on the selected menu item.

18. mnuMItem\_Click(Index As Integer).

To perform procedure **MenuItem** (see item 17).

19. NewEntry() As Integer.

To check whether the record already exists in the database table tblInvRecd.

20. ResetField().

To reset each entry in the dialog box.

21. ReSetTravel().

To display all the staff travel records shown in the staff travel list box of this form again in the staff travel list box of TravelList form.

22. SaveInv().

To perform procedures **CheckEntry** (see item 1) and **NewEntry** (see item 18). If the record is valid, performs procedures **UpdRefNo** (see Section 4.34.1 item 43), **SaveTravel** (see item 23) and insert the record in database table tblInvRecd.

23. SaveTravel(refNo As Integer) As Integer.

To insert the records in database table tblInvTravel.

24. SetFinYr().

To set the financial year based on the invoice date.

25. txtAmt\_LostFocus().

To perform procedure **CheckNumeric** (see Section 4.34.1 item 5).

26. txtFinYr\_LostFocus().

To perform procedure **CheckFinYear** (see Section 4.34.1 item 4).

27. txtInvDate\_LostFocus().

To perform procedure **ValidDate** (see Section 4.34.1 item 44) to validate the date entered.

28. txtInvNo\_LostFocus().

To convert the Invoice number to upper case.

29. txtReason\_LostFocus().

To convert the character **"'"** to **"`"** of the remarks entered in the text box by using the procedure **ConvertString** (see item 7 Section 4.34.1).

30. txtRemark\_LostFocus(Index As Integer).

To convert the character **"'"** to **"`"** of the remarks entered in the text box by using the procedure **ConvertString** (see Section 4.34.1 item 7).

31. txtSent\_LostFocus().

To perform procedure **ValidDate** (see Section 4.34.1 item 44) to validate the date entered.

4.16 JournalAdj Form.

This form is used to input the journal adjustment entries for the vote commitment. It also allows authorised user to update the journal transactions.

4.16.1 Procedures :

1. AddVoteAmt().

To update the record in database table tblJAdj.

2. cmdList\_Click().

To load the DateFrm form for user to select a range of records to be retrieved. Performs procedure **LoadRec** (see Section 4.43.1 item 24) to load the database table tblJAdj.

3. cmdOK\_Click().

To perform procedures **SaveAdj** (see item 22), **EditAdj** (see item 8) or **DeleteAdj** (see item 4) based on the selected menu item.

4. DeleteAdj().

To delete record from database table tblJAdj and perform **DelVoteAmt** (see item 5).

5. DelVoteAmt().

To update the total adjusted amount of a vote in database table tblJAdj.

6. DisableItem().

To disable some enabled entries.

7. DiplayAdj(listText As String).

To retrieve a selected journal record from the database table tblJAdj and display the detailed information in the dialog box. Perform procedure **GetVoteBal** (see item 12).

8. EditAdj().

To perform procedure **Valid\_Entry** (see item 31). If the entries are valid, performs procedures **DelVoteAmt** (see item 5) & **AddVoteAmt** (see item 1) and updates the record in database table tblJAdj.

9. EnableItem().

To perform procedure **LoadCostC** (see Section 4.34.1 item 21) to load the cost centres and enable some disabled entries.

10. Form\_Load().

To load the form.

11. GetJAdj(whereStr As String).

To retrieve all records from the database table tblJAdj to display detailed information in the grid control of the dialog box.

12. GetVoteBal().

To retrieve the current balance of a selected vote from the database table tblJAdj.

13. lstCostC\_Click().

To perform procedure **LoadSelVote** (see Section 4.43.1 item 25) to load all vote of a selected cost centre from database table tblVote.

14. lstRefList\_Click().

To perform procedure **DisplayAdj** (see item 7).

15. lstVoteList\_Click().

To store the selected vote number in variable ***szVoteNo***.

16. MenuItem(Index As Integer).

To perform procedures **View\_JAdj** (see Section 4.36.1 item 15), **DisableItem** (see item 6), **EnableItem** (see item 9), **Print\_JAdj** (see Section 4.36.1 item 7) or **UpdateAdj** (see item 29) based on the selected menu item.

17. mnuMItem\_Click(Index As Integer).

To perform procedure **MenuItem** (see item 16).

18. mskDate\_LostFocus().

To perform procedure **validDate** (see Section 4.34.1 item 44) and **DispFinYrMth** (see Section 4.34.1 item 9).

19. NewEntry() As Integer.

To check whether the record already exists in the database tables tblJAdj and tblJAdjM.

20. ResetField().

To reset each entry in the dialog box.

21. ResetGrid(Index As Integer).

To reformat the form based on the selected menu item.

22. SaveAdj().

To perform procedures **Valid\_Entry** (see item 31) and **NewEntry** (see item 19). Performs procedures **GetMaxCode** (see Section 4.34.1 item 15) & **AddVoteAmt** (see item 1) and insert record in database table tblJAdj.

23. txtFinMth\_LostFocus().

To perform procedure **CheckFinMth** (see Section 4.34.1 item 3).

24. txtPaidAmt\_LostFocus().

To perform procedure **CheckNumeric** (see Section 4.34 item 5). If this entry is valid and the amount is greater than zero, then performs procedure **GetVoteBal** (see item 12).

25. txtRecdAmt\_LostFocus().

To perform procedure **CheckNumeric** (see Section 4.34.1 item 5). If this entry is valid and the amount is greater than zero, then performs procedure **GetVoteBal** (see item 12).

26. txtRef\_LostFocus().

To convert the reference number to upper case.

27. txtRmk\_LostFocus(Index As Integer).

To convert the character **"'"** to **"`"** of the remarks entered in the text box by using the procedure **ConvertString** (see Section 4.34.1 item 7).

28. Upd\_Budget(voteAC As String, voteNo As String, aFinYr As String, aPaid As String, aRecd As String) As Integer.

To update the record in database table tblBudget.

29. UpdateAdj().

To insert all the records from database table tblJAdj in database table tblJAdjM and perform procedure **UpdVoteFunds** (see item 30).

30. UpdVoteFunds(refNo As Integer) As Integer.

To update the records in database table tblVoteFunds and perform procedure **Upd\_Budget** (see item 28) to update database table tblBudget. It also deletes records from table tblJAdj after the completion of the update.

31. Valid\_Entry() As Integer.

To validate each entry in the dialog box.

4.17 MainForm Form.

This is the main application window for displaying all the menu commands to be accessed by the user.

4.17.1 Procedures :

1. Form\_Load().

To load the form.

2. Form\_Unload().

To unload the form and logout from the Sybase server.

3. mnuClaimItem\_Click(Index As Integer).

To load the forms **PettyFrm** (see Section 4.20), **ProfSoc** (see Section 4.22) or **ClaimFrm** (see Section 4.5) based on the selected Claims/Expense Payment menu command.

5. mnuFileItem\_Click(Index As Integer).

To load the forms **CostCFrm** (see Section 4.6), **ExchRFrm** (see Section 4.11), **SocietyFrm** (see Section 4.26) or **VoteFrm** (see Section 4.32) based on the selected File menu command. It also performs procedures **Open\_db** (see Section 4.34.1 item 32), **Close\_DB** (see Section 4.34.1 item 6) or **Supplier\_Dlg** (see Section 4.34.1 item 42) for other selected File menu command.

6. mnuFundItem\_Click(Index As Integer).

To load the forms **JournalAdj** (see Section 4.16), **FundTrsf** (see Section 4.12) or **Budget** (see Section 4.2) based on the selected Funds menu command.

7. mnuHelpItem\_Click(Index As Integer).

To load **AboutBox** form (see Section 4.1) for the **About** command of the **Help** menu.

To perform procedure **Get\_Help** (see item 12) for the on-line user's guide.

8. mnuInvItem\_Click(Index As Integer).

To load the forms **InvoiceFrm** (see Section 4.13), **CredInv** (see Section 4.7) or **InvTravel** (see Section 4.15) based on the selected Invoice menu command.

9. mnuPurchaseItem\_Click(Index As Integer).

To load the forms **POForm** (see Section 4.21), **CancelPO** (see Section 4.4) or **DebitNote** (see Section 4.9) based on the selected Purchase menu command.

10. mnuPymtItem\_Click(Index As Integer).

To load **InvPymt** (see Section 4.14) form or **ExpPymt** (see Section 4.10) form for payment made by NUS Bursar's Office.

11. mnuReimb\_Click(Index As Integer).

To load **Reimburse** (see Section 4.23) form for Petty Cash Recoupment.

12. mnuReportItem\_Click(Index As Integer).

This is the function for printing a list of reports provided by the system. Loads **ReportFrm** form (see Section 4.24) and set the value for global variable ***gszMenuItem***.

13. mnuRptSub\_Click(Index As Integer).

To load **ProfSoc** form for printing the Claims Letter to Bursar's Office.

To load **DateFrm** form and perform procedure **Print\_Subs\_List** (see Section 4.34.1 item 33) to print the Professional Societies Claims report

14. mnuTravelItem\_Click(Index As Integer).

To load **TravelFrm** form (see Section 4.30).

15. mnuUtilItem\_Click(Index As Integer).

To perform functions for accessing Calculator, Lotus, Excel and performing database backup, password amendment (see Section 4.25).

4.18 MsgFrm Form.

The form is used to display the message to inform the user that the database backup in progress.

**Note :** There is no procedure defined in this form.

4.19 OpenDB Form.

The form is used for user to enter the Sybase user id and password to login to the Sybase server.

4.19.1 Procedures :

1. cmdClose\_Click().

To close the dialog box.

2. cmdOK\_Click().

To perform procedure **GetParameter** (see Section 4.34.1 item 17) to get the necessary parameters and **Login** to login to the Sybase server.

3. cmdSetPswd\_Click().

To load the SetPswd form.

4. cmdSetPswd\_GotFocus().

To set the caption of SetPswd form to "To set/change password".

5. Form\_Activate().

To perform procedure **GetParameter** (see Section 4.34.1 item 17) to get the user name from the FCSDB.INI file.

6. Form\_Load().

To set the **OK** button to be enabled and **Close** button as the default button.

7. txtName\_GotFocus().

To set the caption of the form to "Please enter your Sybase login name".

8. txtPswd\_GotFocus().

To set the caption of the form to "Please enter your Sybase login password".

4.20 PettyFrm Form.

The form is used for user to enter the petty cash transactions.

4.20.1 Procedures :

1. CanCancel(Index As Integer).

To check whether the user wishes to update the record before exiting the function. If it is, performs procedures **SavePetty** (see item 23).

2. CheckEntry().

To validate the entries shown in the dialog box.

3. cmdAdd\_Click().

To add the vote commitment in the funds commitment list box. The procedure **CheckFund** (see Section 4.37.1 item 5) is also performed to check whether the funds of the selected vote is available.

4. cmdCancel\_Click().

To perform the procedure **CanCancel** (see item 1).

5. cmdDelete\_Click().

To remove a selected vote commitment from the funds commitment list box.

6. cmdOK\_Click().

To perform procedures **SavePetty** (see item 23), **EditPetty** (see item 8) or **DeletePetty** (see item 7) to update the database table tblPettyCash based on the selected menu item.

7. DeletePetty().

To delete the record from the database table tblPettyCash and perform procedures **WrtPrevBudget** (see Section 4.36.1 item 16), **updateVoteFunds** (see Section 4.37.1 item 13) and **UpdPettyBal** (see item 31).

8. EditPetty().

To perform procedure **CheckEntry** (see item 2) to validate the entries. If the entries are valid, perform procedures **WrtPrevBudget** (see Section 4.36.1 item 16), **updateVoteFunds** (see Section 4.37.1 item 13), **SaveVoteAC** (see Section 4.37.1 item 12) & **UpdPettyBal** (see item 31) and update record in database table tblPettyCash.

9. Form\_Activate().

To perform procedure **LoadCostC** (see Section 4.34.1 item 21) to load the cost centres from database table tblCostC.

10. Form\_Load().

To load the form.

11. GetPetty().

To retrieve a selected record from the database table tblPettyCash.

12. GetPettyBal().

To retrieve the petty cash balance from the database table tblPettyCash.

13. HideItem().

To set some entries to be visible or invisible based on the selected menu item.

14. LoadPetty().

To load all records from the database table tblPettyCash and display them in the Voucher list box.

15. lstCostC\_Click().

To perform procedure **LoadSelVote** (see Section 4.43.1 item 25) to load all votes of the selected cost centre.

16. lstFunds\_Click().

To set **Delete** button to be visible and enabled when the selected menu item is either Add or Edit.

17. lstVoteList\_Click().

To check whether the selected vote has been chosen. If it is not, sets **Add** button to be visible.

18. lstVoucher\_Click().

To perform procedure **GetPetty** (see item 11).

19. MenuItem(Index As Integer).

To perform procedures **LoadPetty** (see item 14).

20. mnuMItem\_Click(Index As Integer).

To perform procedure **MenuItem** (see item 19).

21. NewEntry() As Integer.

To check whether the record already exists in the database table tblPettyCash.

22. ResetField().

To reset each entry in the dialog box.

23. SavePetty().

To perform procedures **CheckEntry** (see item 2) and **NewEntry** (see item 21). If the record is valid, perform procedures **GetMaxCode** (see Section 4.34.1 item 15), **UpdPettyBal** (see item 31) & **SaveVoteAC** (see Section 4.37.1 item 12) and insert the record in database table tblPettyCash.

24. txtAmt\_LostFocus().

To check whether there is sufficient cash for reimbursement. If it is, compute the petty cash balance after deducting this amount and GST.

25. txtCommitt\_LostFocus().

To perform procedure **CheckNumeric** (see Section 4.34.1 item 5).

26. txtDate\_LostFocus().

To perform procedure **ValidDate** (see Section 4.34.1 item 44) to validate the date entered.

27. txtFinMth\_LostFocus().

To perform procedure **CheckFinMth** (see Section 4.34.1 item 3).

28. txtGST\_LostFocus().

To perform procedure **CheckNumeric** (see Section 4.34.1 item 5). If this entry is valid, then computes the petty cash balance.

29. txtRemark\_LostFocus(Index As Integer).

To convert the character **"'"** to **"`"** of the remarks entered in the text box by using the procedure **ConvertString** (see Section 4.34.1 item 7).

30. txtSFunds\_LostFocus().

To perform procedure **CheckNumeric** (see Section 4.34.1 item 5).

31. UpdPettyBal(amt As String) As Integer.

To update the petty cash balance in database table tblPettyCash.

4.21 POForm Form.

The form is used for user to enter the purchase orders.

4.21.1 Procedures :

1. CanCancel(Index As Integer).

To check whether the user wishes to update the record before exiting the function. If it is, performs procedure **UpdatePO** (see item 39) or **SavePO** (see item 26) based on the selected menu item.

2. CheckEntry() As Integer.

To validate the entries of P/O number and supplier. Performs procedure **CheckFunds** (see item 3) to validate other entries.

3. CheckFunds() As Integer.

To validate the entries of currency, P/O amount, exchange rate and amount committed. It also checks whether the total funds commitment is equal to the P/O amount.

4. chkPrePymt\_Click().

To set the variable ***szPrePymt*** based on the selection of the **Prepayment** check box. Set the **Cheque number** text box to be visible if the **Prepayment** check box is checked.

5. cmdAdd\_Click().

To add the vote commitment in the funds commitment list box. The procedure **CheckFund** (see Section 4.37.1 item 5) is also performed to check whether the funds of the selected vote is available.

6. cmdCancel\_Click().

To perform the procedure **CanCancel** (see item 1).

7. cmdDelete\_Click().

To remove a selected vote commitment from the funds commitment list box.

8. cmdList\_Click().

To load the DateFrm form for user to select a range of records to be retrieved. Performs procedure **LoadRec** (see Section 4.43.1 item 24) to load the database table tblPO.

9. cmdOK\_Click().

To perform procedures **SavePO** (see item 26), **UpdatePO** (see item 39) or **DeletePO** (see item 10) to update the database table tblPO based on the selected menu item.

10. DeletePO().

To delete the record from the database table tblPO and perform procedures **WrtPrevBudget** (see Section 4.36.1 item 16), **WrtDelPO** (see item 40) and **updateVoteFunds** (see Section 4.37.1 item 13).

11. DisableItem().

To disable some enabled entries.

12. EnableItem().

To enable some disabled entries.

13. Form\_Activate().

To perform procedures **LoadSupplier** (see item 27 Section 4.43.1), **LoadCurrency** (see Section 4.34.1 item 22) and **LoadCostC** (see Section 4.34.1 item 21) to load the suppliers, currency and cost centres from database tables respectively.

14. Form\_Load().

To load the form.

15. GetPO(poNo As String).

To retrieve a selected record from the database table tblPO.

16. lstCostC\_Click().

To perform procedure **LoadSelVote** (see Section 4.43.1 item 25) to load all votes of the selected cost centre.

17. lstCurrList\_Click().

To perform procedure **Get\_ExchRate** (see Section 4.34.1 item 12) to get the exchange rate.

18. lstFunds\_Click().

To set **Delete** button to be visible.

19. lstPOlist\_Click().

To perform procedure **GetPO** (see item 15).

20. lstSupplier\_Click().

To perform procedure **GetSupplier** (see Section 4.34.1 item 18) and display the supplier's name & address.

21. lstVoteList\_Click().

To check whether the selected vote has been chosen. If it is not, perform procedure **GetPercent** (see Section 4.37.1 item 9) and sets **Add** button to be visible.

22. MenuItem(Index As Integer).

To perform procedures **DisableItem** (see item 11) and **EnableItem** (see item 12).

23. mnuMItem\_Click(Index As Integer).

To perform procedure **MenuItem** (see item 22).

24. NewEntry() As Integer.

To check whether the record already exists in the database table tblPO.

25. ResetField().

To reset each entry in the dialog box.

26. SavePO().

To perform procedures **CheckEntry** (see item 2) and **NewEntry** (see item 24). If the record is valid, perform procedure **SaveVoteAC** (see Section 4.37.1 item 12) and insert the record in database table tblPO.

27. txtAmt\_LostFocus().

To perform procedure **CheckNumeric** (see Section 4.34.1 item 5) to validate the date entered. If it is valid, perform procedure **CalAmount** (see Section 4.37.1 item 2).

28. txtCommitt\_LostFocus().

To perform procedure **CheckNumeric** (see Section 4.34.1 item 5). If this entry is valid, perform procedure **CalPercent** (see Section 4.37.1 item 4).

29. txtExch\_LostFocus().

To perform procedure **CheckNumeric** (see Section 4.34.1 item 5) to validate the date entered. If it is valid, perform procedures **CalAmount** (see Section 4.37.1 item 2) and **CalFunds** (see Section 4.37.1 item 3).

30. txtFinMth\_LostFocus().

To perform procedure **CheckFinMth** (see Section 4.34.1 item 3).

31. txtFinYr\_LostFocus().

To perform procedure **CheckNumeric** (see Section 4.34.1 item 5).

32. txtPercent\_LostFocus().

To perform procedure **GetPercent** (see Section 4.37.1 item 9).

33. txtPODate\_LostFocus().

To perform procedure **ValidDate** (see Section 4.34.1 item 44) to validate the date entered. Perform procedure **DispFinYrMth** (see Section 4.34.1 item 9) if the date is valid.

34. txtPONo\_LostFocus().

To covert the P/O number to upper case.

35. txtReason\_LostFocus(Index As Integer).

To convert the character **"'"** to **"`"** of the reason entered in the text box by using the procedure **ConvertString** (see Section 4.34.1 item 7).

36. txtRemark\_LostFocus(Index As Integer).

To convert the character **"'"** to **"`"** of the remarks entered in the text box by using the procedure **ConvertString** (see Section 4.34.1 item 7).

37. txtSAmt\_LostFocus().

To perform procedure **CheckNumeric** (see Section 4.34.1 item 5).

38. txtSFunds\_LostFocus().

To perform procedure **CheckNumeric** (see Section 4.34.1 item 5).

39. UpdatePO().

To perform procedure **CheckEntry** (see item 2) to validate the entries. If they are valid, perform procedures **WrtPrevBudget** (see Section 4.36.1 item 16), **UpdVoteFunds** (see Section 4.37.1 item 13) & **SaveVoteAC** (see Section 4.37.1 item 12) and update the P/O record in database table tblPO.

40. WrtDelPO(aPO As String) As Integer.

To insert the deleted P/O record in database table delPO.

4.22 ProfSoc Form.

This form is used for user to enter the subscriptions to professional societies that are reimbursed to the staff member.

4.22.1 Procedures :

1. CalAmt().

To perform procedure **GetFeeClaim** (see item 18) to compute the actual subscription claims that should be reimbursed.

2. CalFunds().

To perform procedure **CheckFund** (see Section 4.37.1 item 5).

3. CanCancel(Index As Integer).

To check whether the user wishes to update the record before exiting the function. If it is, performs procedure **UpdateClaim** (see item 47) or **SaveClaim** (see item 33) based on the selected menu item.

4. CheckEntry() As Integer.

To validate the entries.

5. cmdAdd\_Click().

To add a claims in the Claims list box.

6. cmdCancel\_Click().

To perform the procedure **CanCancel** (see item 3).

7. cmdDelete\_Click().

To remove a selected claims record from the Claims list box.

8. cmdOK\_Click().

To perform procedures **SaveClaim** (see item 33), **UpdateClaim** (see item 47) or **DeleteClaim** (see item 10) to update the database table tblClaim based on the selected menu item.

9. cmdOption\_Click().

To load the form StaffInfor (see Section 4.27). Then perform procedures **LoadStaff** (see Section 4.43.1 item 26) and **LoadVote** (see Section 4.43.1 item 28) to load staff and vote records from the database tables.

10. DeleteClaim().

To delete the record from the database table tblClaim and perform procedures **WrtPrevBudget** (see Section 4.36.1 item 16) and **DelSubscript** (see item 11).

11. DelSubscript() As Integer.

To delete the records from the database table tblSubscription and tblSubsVote. Perform procedure **updateVoteFunds** (see Section 4.37.1 item 13).

12. DisableItem().

To disable some enabled entries.

13. DisplayClaims(lstText As String).

To display the detailed information of a claims in the dialog box.

14. DispSubs().

To retrieve the subscription records of a selected claims from the database table tblSubscription.

15. DispSubsVote(lstText As String, period As String).

To retrieve the vote commitment records of a selected claims from the database table tblSubsVote .

16. Form\_Activate().

To perform procedures **LoadCurrency** (see Section 4.34.1 item 22), **LoadProfSoc** (see Section 4.34.1 item 23) and **LoadStaff** (see Section 4.43.1 item 26) to load the currency, professional societies and staff records from database tables respectively.

17. Form\_Load().

To load the form.

18. GetClaim().

To retrieve a selected record from the database table tblClaim.

19. GetCurrClaim().

To get the total subscription claimed of a selected staff from the claims list box and add it to the variable ***totalClaim***.

20. GetFeeClaim().

To get the total subscription claimed of a selected staff from the database table tblSubscription and store it in the variable ***totalClaim***. Performs procedure **GetCurrClaim** (see item 19) if ***totalClaim*** is less than 125.

21. GetSociety(socode As String) As String.

To retrieve a selected record from the database table tblProfSoc.

22. lstClaimRef\_Click().

To perform procedure **GetClaim** (see item 18) to retrieve the detailed information of a selected claim. Performs procedure **PrintLetter** (see item 31) to print the claims letter when the selected menu type is Print.

23. lstClaims\_Click().

To perform procedure **DisplayClaims** (see item 13).

24. lstCurrList\_Click().

To perform procedure **Get\_ExchRate** (see Section 4.34.1 item 12) to get the exchange rate.

25. lstSoc\_Click().

To store the Society code and description of a selected society.

26. lstStaff\_Click().

To perform procedure **LoadVote** (see Section 4.43.1 item 28) to retrieve the cost centres of the selected staff from the database.

27. MenuItem(Index As Integer).

To perform procedures **DisableItem** (see item 12), **LoadClaim** (see Section 4.34.1 item 20) and **LoadStaff** (see Section 4.43.1 item 26).

28. mnuMItem\_Click(Index As Integer).

To perform procedure **MenuItem** (see item 27).

29. mnuPrintItem\_Click(Index As Integer).

To perform procedure **SubsLetter** (see item 37) or **Print\_Subs\_List** (see Section 4.34.1 item 33) based on the selected command of **Print** menu.

30. PrintDetail().

To print the detailed information shown in the dialog box.

31. PrintLetter().

To perform procedure **StartPrinter** (see Section 4.43.1 item 40) to use the HP Laser printer. Prints the Claims Letter header and then perform procedure **PrintDetail** (see item 30) to print the detailed information.

32. ResetField().

To reset each entry in the dialog box.

33. SaveClaim().

To perform procedures **CheckEntry** (see item 4). If the record is valid, perform procedures **GetMaxCode** (see Section 4.34.1 item 15) & **SaveSubscript** (see item 34) and insert the record in database table tblClaim.

34. SaveSubsrcipt() As Integer.

To insert records in database table tblSubsVote and perform procedures **SaveVoteAC** (see item 35).

35. SaveVoteAC() As Integer.

To update the record in database table tblVoteCommit and perform procedures **SaveFunds** (see Section 4.37.1 item 11).

36. SubsDetail(Index As Integer).

To perform procedure **CheckEntry** (see item 4). If the record is valid, perform procedure **SaveVoteAC** (see item 35) and insert the record in database table tblPO.

37. SubsLetter().

To perform procedures **DisableItem** (see item 12) and **LoadClaim** (see Section 4.34.1 item 20).

38. SubsList().

To print the heading of the Claims for Subscription to Professional Societies report and perform procedure **SubsDetail** (see item 36) to print the detailed information.

39. txtCommitt\_LostFocus().

To perform procedure **CheckNumeric** (see Section 4.34.1 item 5). if this entry is valid, perform procedure **CalFunds** (see item 2).

40. txtDate\_LostFocus().

To perform procedure **ValidDate** (see Section 4.34.1 item 44) to validate the date entered. Perform procedure **DispFinYrMth** (see Section 4.34.1 item 9) if the date is valid.

41. txtExch\_LostFocus().

To perform procedure **CheckNumeric** (see Section 4.34.1 item 5) to validate the entry. If it is valid, perform procedures **CalAmount** (see item 2 Section 4.37.1).

42. txtFee\_LostFocus().

To perform procedure **CheckNumeric** (see Section 4.34.1 item 5) to validate the entry. If it is valid, perform procedures **CalAmount** (see item 2 Section 4.37.1).

43. txtFinMth\_LostFocus().

To perform procedure **CheckFinMth** (see Section 4.34.1 item 3).

44. txtFinYr\_LostFocus().

To perform procedure **CheckFinYear** (see Section 4.34.1 item 4).

45. txtRemark\_LostFocus(Index As Integer).

To convert the character **"'"** to **"`"** of the remarks entered in the text box by using the procedure **ConvertString** (see Section 4.34.1 item 7).

46. txtRmk\_LostFocus(Index As Integer).

To convert the character **"'"** to **"`"** of the string entered in the text box by using the procedure **ConvertString** (see Section 4.34.1 item 7).

47. UpdateClaim().

To perform procedure **CheckEntry** (see item 2) to validate the entries. If they are valid, perform procedures **WrtPrevBudget** (see item 16 Section 4.36.1), **DelSubscript** (see item 11) & **SaveSubscript** (see item 34) and update the record in database table tblClaim.

4.23 Reimburse Form.

This form is used for update the petty cash amount received from NUS Bursar's Office.

4.23.1 Procedures :

1. cmdClose\_Click().

To close the dialog box.

2. cmdOK\_Click().

To perform procedure **UpdateTable** (see item 8) to update the database tables tblPettyCash and tblPettyM.

3. Form\_Load().

To load the form.

4. GetPettyBal().

To retrieve the petty cash balance from the database table tblPettyCash.

5. txtAmt\_LostFocus().

To perform procedure **CheckNumeric** (see Section 4.34.1 item 5) to validate the amount entered and make sure that it is greater than zero.

5. txtDate\_LostFocus().

To perform procedure **ValidDate** (see Section 4.34.1 item 44) to validate the date entered. Performs procedure **DispFinYrMth** (see Section 4.34.1 item 9) if the date is valid.

6. txtFinMth\_LostFocus().

To perform procedure **CheckFinMth** (see Section 4.34.1 item 3).

7. txtFinYr\_LostFocus().

To perform procedure **CheckNumeric** (see Section 4.34.1 item 5).

8. UpdateTable().

To validate the amount entered and perform procedure **GetPettyBal** (see item 4) to confirm that the petty cash is not more than 500. Transfer all the records from database table tblPettyCash to tblPettyM and update the petty cash balance in tblPettyCash.

4.24 ReportFrm Form.

The form is the command form for user to view the detailed information of all reports before printing them to the printer.

4.24.1 Procedures :

1. BudgetHeader(startCol As Integer, endCol As Integer).

To print the report header of the Budgeted Accounts report.

2. CalAmtCommit(tMTD() As Double, tVote As Double) As Double.

To compute the total month-to-date commitment of a vote.

3. CancelHeader().

To print the report header of the Cancellation of POs Transactions report.

4. CheckSTCC(prevCC As String, prevVote As String).

This procedure is used to compare the previous cost centre and the current cost centre during printing the Staff Travelling report. If they are not equal, perform procedure **Travel\_Total** (see item 69) and then print the cost centre description. Please note that the procedure **Travel\_Total** will only be printed when the report is printed by cost centre.

5. CheckSTStaff(prevStaff As String, prevCC As String, prevVote As String).

This procedure is used to compare the previous staff id and the current staff id during printing the Staff Travelling report by cost centre order. If they are not equal, perform procedure **Travel\_Total** (see item 69) and then print the cost centre description.

6. DisplayBudget().

To reformat the grid control cells and perform procedure **GetBudget** (see item 20) to display the Budget Accounts records in the cells.

7. DisplayCancel().

To reformat the grid control cells and perform procedure **GetCancel** (see item 21) to display the Cancelled P/O records in the cells.

8. DisplayExpn().

To reformat the grid control cells and perform procedure **GetExpn** (see item 25) to display the claims/expenses payments records in the cells.

9. DisplayInv().

To reformat the grid control cells and perform procedure **GetInv** (see item 26) to display the invoice transaction records in the cells.

10. DisplayPettyC().

To reformat the grid control cells and perform procedure **GetPettyCash** (see item 27) to display the petty cash transaction records in the cells.

11. DisplayPO().

To reformat the grid control cells and perform procedure **GetPO** (see item 29) to display the purchase order transaction records in the cells.

12. DisplayTAE().

To perform procedure **GetCostC** (see item 23) to store the cost centre code and description in the array variable ***voteArray***. Then reformat the grid control cells and perform procedure **GetTAE** (see item 31) to display the total accumulative expenditure records in the cells.

13. DisplayTravel().

To perform procedure **SetTravelCond** (see item 63) to set the *where* clause and *order by* clause of the SQL command. Then reformat the grid control cells and perform procedure **GetTravel** (see item 32) to display the Journal Adjustment records in the cells.

14. DisplayUPA().

To reformat the grid control cells and perform procedure **GetUPA** (see item 33) to display the unsettled payable accounts in the cells.

15. DisplayVCR().

To reformat the grid control cells and perform procedure **GetVCR** (see item 34) to display the vote control register in the cells.

16. DispVoteBal(voteDesc As String, prevAC As String).

To retrieve the current balance of a vote from the database table tblVoreFunds.

17. ExpnHeader().

To print the report header of the Claims / Expense Payment Transactions report.

18. Form\_Load().

To load the form and initialise the function.

19. Form\_Resize().

To resize the size of grid control according to the form size.

20. GetBudget(whereVote As String).

To retrieve the vote number, cost centre account code, vote description and the budgeted amount from database tables tblVoteFunds and tblVote. Display the retrieved information in the cells.

21. GetCancel(whereDate As String).

To retrieve the detailed information of cancelled P/Os from database tables tblCancelPO and tblPO. Display the records in the cells and perform procedure **GetCancelVote** (see item 22).

22. GetCancelVote(cancelNo As String) As Integer.

To retrieve the cancelled votes commitment from the database table tblCancelVote and display the records in the cells.

23. GetCostC().

To retrieve the cost centre code and description from the database table tblCostC and store them in the array variable ***voteArray***. This is used to display the information in the cells at the first row for the Total Accumulative Expenditure report.

24. GetDate(tableType As String, nextRow).

To retrieve the P/O date from the database table tblPO for printing the Invoice transactions report.

25. GetExpn(whereDate As String).

To retrieve the detailed information of cancelled P/Os from database table tblClaim. Display the records in the cells and perform procedure **GetVoteCommit** (see item 37).

26. GetInv(orderBy As String, whereDate As String).

To retrieve the detailed information of invoice records from database table tblInvRecd. Display the records in the cells and perform procedure **GetDate** (see item 24).

27. GetPettyCash(whereStr As String, tblName As String).

To retrieve the detailed information of petty cash transactions from database table tblPettyCash or tblPettyM. Display the records in the cells and perform procedures **GetPettyVote** (see item 28) & **PettySummary** (see item 44).

28. GetPettyVote(formNo As String, newRow As Integer) As Integer.

To retrieve the votes commitment of petty cash transactions from database table tblVoteCommit.

29. GetPO(orderBy As String, whereDate As String).

To retrieve the detailed information of P/O records from database table tblPO. Display the records in the cells and perform procedure **GetPO\_Inv** (see item 30).

30. GetPO\_Inv(poNo As String) As Integer.

To retrieve the detailed information of invoice of a P/O from database table tblInvRecd.

31. GetTAE().

To retrieve the detailed information of vote commitment records from database tables tblVoteFunds, tblVote and tblCostC. Perform procedure **CalAmtCommit** (see item 2) and display the records in the cells.

32. GetTravel(whereDate As String, orderBy As String).

To retrieve the detailed information of staff travelling records from database tables tblTravel and tblTravelVote. Display the records in the cells.

33. GetUPA(orderBy As String, szSel As String).

To perform procedure **SetUPA** (see item 64) to set the *order by* clause and *where* clause of the SQL command. Then retrieve the detailed information from the temporary table #t1 and display the records in the cells.

34. GetVCR(commitStr As String).

To perform procedure **SetCCType** (see item 62) to set the *where* clause of the SQL command. Then retrieve the detailed information from the temporary table #t1 and perform procedures **SetVCR\_Table** (see item 65) & **GetVCR\_Rec** (see item 35).

35. GetVCR\_Rec(whereStr As String).

To retrieve the records from temporary table #t1. Display the detailed information in the cells and perform procedure **GetVCRDetail** (see item 36).

36. GetVCRDetail(tblType$, formNo$, poInv$ voteAmt As Double) As Integer.

To perform procedures **VCRPODetail** (see item 80), .**VCRInvDetail** (see item 77), .**VCRPettyDetail** (see item 79), **VCRClaimDetail** (see item 75), **VCRTravelDetail** (see item 81) or **VCRJAdjDetail** (see item 78) based on the table type.

37. GetVoteCommit(formNo As String) As Integer.

To retrieve the votes commitment from the database table tblVoteCommit for the Claims/expense payments report.

38. InvoiceHeader().

To print the report header of the Received Invoice transactions report.

39. JAdjHeader().

To print the report header of the Journal Adjustment transactions report.

40. mnuMItem\_Click(Index As Integer).

To perform procedure **PrintReport** (see item 56) when the selected menu item is **Print**.

41. mnuOpt\_Click(Index As Integer).

This is applied to the **'Next'** and **'Previous'** menu item for the Petty Cash transactions displayed on the screen. Set the first grid control to be invisible when **'Next'** menu is selected. Otherwise, set the second grid control to be invisible.

42. NextVoteDesc(voteDesc As String, r As Integer) As Integer.

To print the next vote description.

43. PettyHeader().

To print the report header of the Petty Cash Transactions report.

44. PettySummary(tblName As String, whereDate As String).

To print the petty cash summary by vote and perform procedure **PettySummary\_costC** (see item 45).

45. PettySummary\_costC().

To print the petty cash summary by cost centre.

46. POHeader().

To print the report header of the Purchase Order transactions report.

47. PrintBudget().

To perform procedure **StartPrinter** (see Section 4.43.1 item 40) to use the Epson dot-matrix printer. Then print the details of budget accounts shown in the cells to printer and perform procedure **PrintBudget\_part2** (see item 48) when the number of columns is more than 10.

48. PrintBudget\_part2(nRow As Integer, noCols As Integer).

To print the second part of the budget accounts detailed information from the screen to printer.

49. PrintCancel().

To perform procedure **StartPrinter** (see Section 4.43.1 item 40) to use the Epson dot-matrix printer. Then print the details of cancelled P/Os shown in the cells to printer.

50. PrintExpn().

To perform procedure **StartPrinter** (see Section 4.43.1 item 40) to use the Epson dot-matrix printer. Then print the details of claims/expense payment transactions shown in the cells to printer and perform procedure **NextVoteDesc** (see item 42) to print the next vote description.

51. PrintInv().

To perform procedure **StartPrinter** (see Section 4.43.1 item 40) to use the Epson dot-matrix printer. Then print the details of received invoice transactions shown in the cells to printer.

52. PrintJAdj().

To perform procedure **StartPrinter** (see Section 4.43.1 item 40) to use the Epson dot-matrix printer. Then print the details of journal adjustment transactions shown in the cells to printer.

53. PrintPettyCash().

To perform procedure **StartPrinter** (see Section 4.43.1 item 40) to use the Epson dot-matrix printer. Then print the details of petty cash transactions shown in the cells to printer and perform **PrintPettySummary** (see item 54).

54. PrintPettySummary().

To print the petty cash summary shown in the second grid control to printer.

55. PrintPO().

To perform procedure **StartPrinter** (see Section 4.43.1 item 40) to use the Epson dot-matrix printer. Then print the details of purchase order transactions shown in the cells to printer.

56. PrintReport().

To perform procedure **PrintPO** (see item 55), **PrintInv** (see item 51), **PrintPettyCash** (see item 53), **PrintExpn** (see item 50), **PrintTravel** (see item 59), **PrintUPA** (see item 60), **PrintVCR** (see item 61), **PrintTAE** (see item 57), **PrintJAdj** (see item 52), **Print\_Trsf** (see Section 4.36.1 item 8), **PrintCancel** (see item 49) and **PrintBudget** (see item 47) based on the selected command of the **Report** menu.

57. PrintTAE().

To perform procedure **StartPrinter** (see Section 4.43.1 item 40) to use the Epson dot-matrix printer. Then print the details of total accumulative expenditure shown in the cells to printer and perform procedure **PrintTAE\_part2** (see item 58) when the number of columns is more than 10.

58. PrintTAE\_part2(nRow As Integer, noCol As Integer).

To print the second part of the total accumulative expenditure detailed information from the screen to printer.

59. PrintTravel().

To perform procedure **StartPrinter** (see Section 4.43.1 item 40) to use the Epson dot-matrix printer. Then print the details of staff travel transactions shown in the cells to printer.

60. PrintUPA().

To perform procedure **StartPrinter** (see Section 4.43.1 item 40) to use the Epson dot-matrix printer. Then print the details of unsettled payable accounts shown in the cells to printer.

61. PrintVCR().

To perform procedure **StartPrinter** (see Section 4.43.1 item 40) to use the Epson dot-matrix printer. Then print the details of vote control register shown in the cells to printer.

62. SetCCType(whereStr As String).

To set the *where* clause of the SQL command for displaying the staff travelling records based on the selected type of cost centres.

63. SetTravelCond(whereDate As String, orderBy As String).

To set the *where* clause and *order by* clause of the SQL command for displaying the staff travelling records.

64. SetUPA(orderBy As String, szSel As String) As Integer.

To insert records from tblInvRecd and tblPO into a temporary working table #t1 for printing the unsettled payable report.

65. SetVCR\_Table() As Integer.

To insert and update records in the temporary working table #t1. This table is used to display the detailed information of the vote control register.

66. SetVCRCommit() As String.

To set the SQL command string to compute the vote commitment for the vote control register.

67. SummaryHeader().

To print the petty cash summary report header.

68. TAEHeader(startCol As Integer, endCol As Integer).

To print the report header for the Total Accumulative Expenditure report.

69. Travel\_Total().

To print the total amount by vote for the staff travelling report.

70. TravelHeader().

To print the report header for the Staff Travelling report.

71. UPAHeader().

To print the report header for the Unsettled Payable Accounts report.

72. VCR\_Next\_Vote(voteBudget, voteBF, currRow).

To display the next vote description in the Vote control register when the current vote number is different from the previous one.

73. VCR\_Total().

To print the total amount of each vote in the vote control register.

74. VCRCDNote(formNo As String, aPO As String, atRow As Integer) As Integer.

To retrieve the credit/debit note records of an invoice from database tables tblCDNote and tblVoteCommit. These records are to be displayed in the vote control register.

75. VCRClaimDetail(dbProc As Integer, rowPos As Integer, formNo As String, voteAmt As Double, tblType As String).

To retrieve a claims/expense payment record from database tables tblClaim and tblVoteCommit. These records are to be displayed in the vote control register.

76. VCRHeader(aVote As String, totalFlg As String).

To print the report header for the Vote Control Register.

77. VCRInvDetail(dbProc As Integer, rowPos As Integer, formNo As String, voteAmt As Double).

To retrieve a invoice record from database tables tblInvRecd and tblVoteCommit. These records are to be displayed in the vote control register.

78. VCRJAdjDetail(dbProc As Integer, rowPos As Integer, formNo As String, voteAmt As Double, tblType As String).

To retrieve adjustment records from database table tblJAdjM. These records are to be displayed in the vote control register.

79. VCRPettyDetail(dbProc As Integer, rowPos As Integer, formNo As String, voteAmt As Double).

To retrieve petty cash records from database tables tblPettyCash, tblPettyM and tblVoteCommit. These records are to be displayed in the vote control register.

80. VCRPODetail(dbProc As Integer, rowPos As Integer, formNo As String, voteAmt As Double).

To retrieve purchase order records from database tables tblPO and tblVoteCommit. It also retrieves the invoice records of a specified P/O from database tables tblInvRecd and tblVoteCommit. These records are to be displayed in the vote control register.

81. VCRTravelDetail(dbProc As Integer, rowPos As Integer, formNo As String, voteAmt As Double, tblType As String).

To retrieve staff travel records from database table tblTravel and tblVoteCommit. These records are to be displayed in the vote control register.

4.25 SetPswd Form.

The form is used for user to change the password.

4.25.1 Procedures :

1. cmdClose\_Click().

To close the dialog box.

2. cmdOK\_Click().

To verify the new password and re-type new password whether they are equal. If it is, perform procedures **Login** and **SqlCmd** to update the new password.

3. Form\_Load().

To set the **OK** button to be disabled and **Close** button as the default button.

4. txtOldPswd\_Change().

To reset the **OK** button.

4.26 SocietyFrm Form.

The form is used for maintaining the professional societies.

4.26.1 Procedures :

1. CanCancel(Index As Integer).

To check whether the user wishes to update the record before exiting the function. If it is, perform procedure **SaveTable** (see item 14).

2. cmdOK\_Click().

To perform procedures **SaveTable** (see item 14) or **DeleteTable** (see item 3) to update the database table tblProfSoc based on the selected menu item.

3. DeleteTable().

To delete the record from the database table tblProfSoc.

4. DisableItem().

To disable some enabled entries.

5. EnableItem().

To enable some disabled entries.

6. Form\_Activate().

To perform procedures **LoadProfSoc** (see Section 4.34.1 item 23) to load the professional societies from database table tblProfSoc.

7. Form\_Load().

To load the form.

8. Form\_QueryUnload(Cancel As Integer, UnloadMode As Integer).

To perform procedure **CanCancel** (see item 1) to check whether there is any record has to be updated.

9. lstSoc\_DblClick().

To display the detailed information of the selected society code.

10. MenuItem(Index As Integer, currFrom As Form).

To perform procedures **DisableItem** (see item 4) and **EnableItem** (see item 5).

11. mnuMItem\_Click(Index As Integer).

To perform procedure **MenuItem** (see item 10).

12. NewEntry() As Integer.

To check the entered society code whether it already exists.

13. ResetField().

To reset each entry in the dialog box.

14. SaveTable().

To perform procedures **validEntry** (see item 16) and **NewEntry** (see item 12). If it is valid, then update the record in database table tblProfSoc.

15. txtDesc\_LostFocus().

To convert the character **"'"** to **"`"** of the description entered in the text box by using the procedure **ConvertString** (see Section 4.34.1 item 7).

16. validEntry() As Integer.

To check whether the entries of Society code and description are empty.

4.27 StaffInfor Form.

The form is used for user to update the the cost centres that belong to a specified staff member.

4.27.1 Procedures :

1. cboCostC\_Click().

To set the **Delete** button to be invisible and **Add** button to be visible.

2. cmdAdd\_Click().

To insert a selected cost centre from the Cost Centre combo box in the Cost centre list box.

3. cmdCancel\_Click().

To unload the form.

4. cmdDelete\_Click().

To remove a selected cost centre from the Cost centre list box.

5. cmdOK\_Click().

To perform procedure **validEntry** (see item 12). If the entries are valid, then perform procedure **SaveCostC** (see item 11) to update the record.

6. DispStaff().

To retrieve the staff details of a selected staff from the database table tblStaffInfo.

7. Form\_Load().

To load the form.

8. LoadProgram().

To load the program name and the correspondent cost centre from database tables tblProgram and tblCostC and display them in the Cost Centre combo box.

9. lstCostC\_Click().

To set the **Delete** button to be visible and the **Add** button to be invisible.

10. NewEntry() As Integer.

To check the entered staff email id whether it already exists.

11. SaveCostC().

To update the cost centres of staff in database table tblPrograms.

12. validEntry() As Integer.

To check whether the entries of staff name and email id are empty. It also check whether any cost centres are inserted in the cost centre list box.

4.28 SuppFrm Form.

The form is used for maintaining the supplier records that are used by both Inventory Control System (ICS) and this system.

4.28.1 Procedures :

1. CanCancel(Index As Integer).

To check whether the user wishes to update the record before exiting the function. If it is, perform procedure **SaveTable** (see item 16).

2. cmdCancel\_Click().

To perform the procedure **CanCancel** (see item 1).

3. cmdOK\_Click().

To perform procedures **SaveTable** (see item 16) or **DeleteTable** (see item 4) to update the database table Supplier based on the selected menu item.

4. DeleteTable().

To delete the record from the database table Supplier.

5. DisableItem().

To disable some enabled entries.

6. EnableItem().

To enable some disabled entries.

7. Form\_Activate().

To perform procedure **LoadSupplier** (see Section 4.43.1 item 27) to load the suppliers from database table Supplier.

8. Form\_Load().

To load the form.

9. Form\_QueryUnload(Cancel As Integer, UnloadMode As Integer).

To perform procedure **CanCancel** (see item 1) to check whether there is any record which needs to be updated.

10. GetAll().

To peroform procedure **GetSupplier** (see Section 4.34.1 item 18) to display the detailed information of a selected supplier..

11. lstSupplier\_Click().

To perform procedure **GetAll** (see item 10) to retrieve the detailed information of a selected supplier.

12. MenuItem(Index As Integer, currFrom As Form).

To perform procedures **DisableItem** (see item 5) and **EnableItem** (see item 6).

13. mnuMItem\_Click(Index As Integer).

To perform procedure **MenuItem** (see item 12).

14. NewEntry() As Integer.

To check the entered supplier whether it already exists.

15. ResetField().

To reset each entry in the dialog box.

16. SaveTable().

To perform procedure **NewEntry** (see item 14). Perform procedure **GetMaxCode** (see Section 4.34.1 item 15) and insert the record in database table Supplier when it does not exist. Otherwise, update the record.

17. txtAddress\_LostFocus(Index As Integer).

To convert the character **"'"** to **"`"** of the address entered in the text box by using the procedure **ConvertString** (see Section 4.34.1 item 7). It also converts the address to upper case characters.

18. txtSupplier\_LostFocus(Index As Integer).

To convert the character **"'"** to **"`"** of the supplier's name entered in the text box by using the procedure **ConvertString** (see Section 4.34.1 item 7). It convert the entered supplier's name to upper case character and checks whether it is already shown in the Supplier list box.

4.29 TAEOptFrm Form.

The form is used for user to accept the necessary data for printing the total accumulative expenditure report.

4.29.1 Procedures :

1. chkBudget\_Click(Value As Integer).

To store the value of the Budgeted amount check box in global variable ***gnIndex***.

2. chkVote\_Click(Value As Integer).

If **All Votes** check box is checked, set the value of global variable ***gszSelection*** to **'A'** and disable the Vote list box. Otherwise, set the value of global variable ***gszSelection*** to **'S'** and enable the Vote list box.

3. cmdCancel\_Click().

To close the dialog box.

4. cmdOK\_Click().

To store the range of date in the global variables ***gszDateFrm*** and ***gszDateTo***. The global variable ***gnFinYr*** stores the data entered in the Financial Year text box. It transfers the list of selected votes in the Vote list box to the Vote list box of ReportFrm form.

5. Form\_Load().

To load the form.

6. Load\_VoteAC(szCCType As String).

To load the vote records of the selected cost centre type from database tables tblVote and tblCostC and display them in the Vote list box.

7. optCostC\_Click(Index As Integer, Value As Index).

To set the value of variable szCCType based on the selected Cost centre type and perform procedure **Load\_VoteAC** (see item 6).

8. txtFinYr\_LostFocus().

To perform procedure **CheckFinYear** (see Section 4.34.1 item 4).

4.30 TravelFrm Form.

This form is used for user to enter/update the staff travelling records.

4.30.1 Procedures :

1. AddTravelVote(voteAC As String, voteNo As String, amtS As String) As Integer.

To insert record in database table tblTravelVote.

2. CalAmt().

To compute the registration amount in S$ equivalent and perform procedure **CalTotalFunds** (see item 3).

3. CalTotalFunds().

To compute the total funds and perform procedure **GenVoteFunds** (see item 16).

4. CanCancel(Index As Integer).

To check whether the user wishes to update the record before exiting the function. If it is, perform procedures **SaveTravel** (see item 27), **EditTravel** (see item 13) or **UpdTravel** (see item 38) based on the selected menu item.

5. chkStatus\_Click().

To set the variable ***szColStatus*** based on the selection of the **Update completely** check box. Perform procedure **CalTotalFunds** (see item 3) if the **'Update completely'** check box is checked and the selected menu item is **'Update'**.

6. cmdCancel\_Click().

To perform the procedure **CanCancel** (see item 4).

7. cmdList\_Click().

To load the DateFrm form for user to select a range of records to be retrieved. Perform procedure **LoadTravel** (see item 18) to load the database table tblTravel.

8. cmdOK\_Click().

To perform procedures **SaveTravel** (see item 27), **EditTravel** (see item 13), **DeleteTravel** (see item 10) or **UpdTravel** (see item 38) to update the database table tblTravel based on the selected menu item.

9. cmdOption\_Click().

To load the form StaffInfor (see Section 4.27). Then perform procedures **LoadStaff** (see Section 4.43.1 item 26) and **LoadVote** (see Section 4.43.1 item 28) to load staff and vote records from the database tables.

10. DeleteTravel().

To delete the record from the database table tblTravel and perform procedures **WrtPrevBudget** (see Section 4.36.1 item 16), **updateVoteFunds** (se Section 4.37.1e item 13) & **DelTravelVote** (see item 11).

11. DelTravelVote() As Integer.

To delete the record from the database table tblTravelVote.

12. DisplayCC().

To get the list of committed cost centres from the Funds commitment list box and perform procedure **LoadVote** (see Section 4.43.1 item 28) to retrieve the description of selected cost centres in cost centre list box.

13. EditTravel().

To perform procedures **WrtPrevBudget** (see Section 4.36.1 item 16), **updateVoteFunds** (see Section 4.37.1 item 13), **DelTravelVote** (see item 11) and **SaveVoteAC** (see item 28).The database table tblTravel is also updated.

14. Form\_Activate().

To perform procedures **LoadCurrency** (see Section 4.34.1 item 22) and **LoadStaff** (see Section 4.43.1 item 26) to load the currency and staff records from database tables tblCurrency and tblStaffInfor respectively.

15. Form\_Load().

To load the form.

16. GenVoteFunds(totalFunds As Double).

To compute the funds commitment for the votes involved based on the total funds amount and perform procedure **CheckFunds** (see Section 4.37.1 item 5).

17. GetTravel(listText As String).

To retrieve a selected staff travel record from the database table tblTravel and perform procedures **DisplayFunds** (see Section 4.34.1 item 7) and **DisplayCC** (see item 12).

18. LoadTravel(szCmd As String).

To load the staff travelling records from the database table tblTravel and display them in the Staff Travel list box.

19. lstCurrList\_Click().

To perform procedure **Get\_ExchRate** (see Section 4.34.1 item 12) to get the exchange rate.

20. lstStaff\_Click().

To perform procedure **LoadVote** (see Section 4.43.1 item 28) to load the cost centre of a selected staff from database tables tblCostC & tblVote and view vwStaffCC.

21. lstTravel\_Click().

To perform procedure **GetTravel** (see item 17).

22. MakeInvisible().

To set the currency list box to be invisible.

23. MenuItem(Index As Integer).

To set some entries to be visible/invisible based on the selected menu.

24. mnuMItem\_Click(Index As Integer).

To perform procedure **MenuItem** (see item 23).

25. mskDate\_LostFocus().

To perform procedure **validDate** (see Section 4.34.1 item 44) to validate the date entered. If it is valid, perform procedure **DispFinYrMth** (see Section 4.34.1 item 9).

26. ResetField().

To reset each entry in the dialog box.

27. SaveTravel().

To perform procedures **Valid\_Travel** (see item 40). If the record is valid, perform procedure **GetMaxCode** (see Section 4.34.1 item 15) and insert the record in database table tblTravel. It also performs procedure **SaveVoteAC** (see item 28) to update the vote commitment.

28. SaveVoteAC(frmNo As String, aPO As String, aDate As String) As Integer.

To insert the list of vote commitment in database table tblVoteCommit and perform procedure **SaveFunds** (see Section 4.37.1 item 11). If the selected menu item is **Update**, then performs procedure **UpdTravelVote** (see item 39). Otherwise, perform procedure **AddTravelVote** (see item 1).

29. txtBudget\_LostFocus().

To perform procedure **CheckNumeric** (see Section 4.34.1 item 5) to validate the entry. If it is valid, perform procedure **GenVoteFunds** (see item 16).

30. txtCurr\_GotFocus().

To set the currency list box and exchange rate text box to be visible.

31. txtEvent\_LostFocus(Index As Integer).

To convert the character **"'"** to **"`"** of the string entered in the text box by using the procedure **ConvertString** (see Section 4.34.1 item 7).

32. txtEventDate\_LostFocus(Index As Integer).

To convert the character **"'"** to **"`"** of the string entered in the text box by using the procedure **ConvertString** (see Section 4.34.1 item 7).

33. txtExch\_LostFocus().

To perform procedure **CheckNumeric** (see Section 4.34.1 item 5) to validate this entry. If it is valid, perform procedure **CalAmount** (see Section 4.37.1 item 2).

34. txtExp\_LostFocus(Index As Integer).

To perform procedure **CheckNumeric** (see Section 4.34.1 item 5) to validate this entry. If it is valid, perform procedures **CalTotalFunds** (see item 3) and **CalFunds** (see Section 4.37.1 item 3).

35. txtFinMth\_LostFocus().

To perform procedure **CheckFinMth** (see Section 4.34.1 item 3).

36. txtFinYr\_LostFocus().

To perform procedure **CheckFinYear** (see Section 4.34.1 item 4).

37. txtRegn\_LostFocus().

To perform procedure **CheckNumeric** (see Section 4.34.1 item 5) to validate this entry if the selected menu item is **'Update'**. If it is valid, perform procedure **CalAmt** (see item 2).

38. UpdTravel().

To perform procedure **WrtPrevBudget** (see Section 4.36.1 item 16) and update record in database table tblTravel. Performs procedures **updateVoteFunds** (see Section 4.37.1 item 13) and **SaveVoteAC** (see item 28) if the **'Update completely'** check box is checked. Otherwise, perform procedure **UpdateTravelVote** (see item 39).

39. UpdTravelVote(voteAC As String, voteNo As String, amtS As String, i As Integer) As Integer.

To update the record in database table tblTravelVote.

40. Valid\_Travel() As Integer.

To validate each entry in the dialog box.

4.31 TravelList Form.

The form is used to display list of staff travelling records for user to update the air fare amount after entering the invoice received from the travel agent.

4.31.1 Procedures :

1. cmdAdd\_Click().

To insert the selected staff travel record in the selected staff travel list box and set the **Add** button to be invisible.

2. cmdCancel\_Click().

To unload the form.

3. cmdDelete\_Click().

To remove a selected record from the selected staff travel list box and set the **Delete** button to be invisible.

4. cmdOK\_Click().

To check whether the total air fare is equal to the invoice amount. If it is, then perform procedure **ReSetTravel** (see item 11) to update the record.

5. Form\_Load().

To load the form and perform procedures **ResetField** (see item 10) and **LoadTravel** (see item 7).

6. GetTravel(listText As String).

To retrieve a selected staff travel record from the database table tblTravel.

7. LoadTravel(szCmd As String).

To load the staff travelling records from the database table tblTravel and display them in the Staff Travel list box.

8. lstSel\_Click().

To set the **Delete** button to be visible.

9. lstTravel\_Click().

To perform procedure **GetTravel** (see item 6).

10. ResetField().

To reset each entry in the dialog box.

11. ReSetTravel().

To clear the selected staff travel list box in InvTravel form (see Section 4.15) and transfer all the items shown in the selected staff travel list box to the one in form InvTravel.

12. txtAmt\_LostFocus(Index As Integer).

To perform procedure **CheckNumeric** (see Section 4.34.1 item 5) to validate the entry. If it is valid, computes the total expenses.

4.32 VoteFrm Form.

The form is used for maintaining the vote records of the accounts.

4.32.1 Procedures :

1. CanCancel(Index As Integer).

To check whether the user wishes to update the record before exiting the function. If it is, performs procedure **SaveTable** (see item 21).

2. CheckFunds\_exist() As Integer.

To check the selected vote whether there is any funds commitment (exists in database table tblVoteFunds) before deleting it. If it is, return **False**.

3. chkAll\_Click(Value As Integer).

To disable the Cost centre list box if the **'All cost centres'** check box is checked. Otherwise, enable it.

4. chkFund\_Click(Value As Integer).

To set the **Period** and **Financial year** text box to be visible if the **'Development Fund'** check box is checked. Otherwise, they should be invisible.

5. cmdCancel\_Click().

To perform the procedures **CanCancel** (see item 1) and **ResetField** (see item 20).

6. cmdOK\_Click().

To perform procedures **SaveTable** (see item 21), **UpdTable** (see item 25) or **DeleteTable** (see item 7) to update the database table tblVote based on the selected menu item.

7. DeleteTable().

To perform procedure **CheckFunds\_exist** (see item 2). If the returned value is **True** (no commitment), then deletes the record from the database table tblVote.

8. DisableItem().

To disable some enabled entries.

9. EnableItem().

To enable some disabled entries.

10. Form\_Activate().

To perform procedures **LoadCostC** (see Section 4.34.1 item 21) to load the cost centres from database tables tblCostC.

11. Form\_Load().

To load the form.

12. Form\_QueryUnload(Cancel As Integer, UnloadMode As Integer).

To perform procedure **CanCancel** (see item 1).

13. GetAll().

To retrieve a selected vote record from the database table tblVote and display the detailed information on the screen.

14. lstCostC\_Click().

To perform procedure **LoadSelVote** (see Section 4.43.1 item 25) to load all votes of the selected cost centre.

15. lstVoteList\_Click().

To perform procedure **SetVoteNo** (see item 23).

16. MenuItem(Index As Integer, currForm As Form).

To perform procedure **DisableItem** (see item 8) or **EnableItem** (see item 9) based on the selected menu item.

17. mnuMItem\_Click(Index As Integer).

To perform procedure **MenuItem** (see item 16).

18. NewEntry() As Integer.

To check the entered vote record whether it already exists.

19. ResetCostC().

To set a *selected* item in the Cost Centre list box to be *not selected*.

20. ResetField().

To reset each entry in the dialog box.

21. SaveTable().

To perform procedures **validEntry** (see item 26). If the record is valid, perform procedure **SaveVote** (see item 22) to insert the record in database table tblVote.

22. SaveVote(voteAC As String, voteNo As String, voteDesc As String).

To check whether the record already exists in the database table tblVote. If it is not, insert the record in the table.

23. SetVoteNo().

To display the vote number, cost centre account and the vote description of a selected vote record. Then perform procedure **GetAll** (see item 13) to retrieve the detailed information from the database.

24. txtVoteDesc\_LostFocus().

To convert the character **"'"** to **"`"** of the vote description entered in the text box by using the procedure **ConvertString** (see Section 4.34.1 item 7).

25. UpdTable().

To perform procedures **vaildEntry** (see item 26) and **NewEntry** (see item 18). If the record is valid, then update record in database table tblVote.

26. validEntry() As Integer.

To validate each entry in the dialog box. It also ensures that at least one cost centre should be selected if the **'All Cost Centre'** check box is unchecked.

4.33 VoteFund Form.

The form is used for user to update the vote commitment of an invoice when it is different from the commitment of purchase order.

4.33.1 Procedures :

1. cmdCancel\_Click().

To restore the funds commitment from the global variable array ***voteArray*** (the data was stored in InvoiceFrm form (see Section 4.13)) to the funds commitment list box.

2. cmdOK\_Click().

To compare the total amount committed with the total invoice amount entered in form InvoiceFrm (see Section 4.13) before closing this form and switching back to InvoiceFrm form.

3. cmdUpdate\_Click().

To re-update the amount committed of the selected vote in funds commitment list box.

4. Form\_Load().

To load the form and transfer the funds commitment from the global variable array ***voteArray*** (the data was stored in InvoiceFrm form (see Section 4.13)) to the funds commitment list box.

5. lstFunds\_DblClick().

To display the detailed information of a selected funds commitment.

6. txtCommit\_LostFocus().

To perform procedure **CheckNumeric** (see Section 4.34.1 item 5).

4.34 FCSAP.BAS Module.

The is a module to store the common procedures to be shared among the forms.

4.34.1 Procedures :

1. CheckDate\_From(aForm As Form).

To perform procedure **ValidDate** (see item 44) to validate the first date (date from) in a dialog box. If it is valid, then perform procedure **DispFinYrMth** (see item 9).

2. CheckDate\_To(aForm As Form) As Integer.

To perform procedure **ValidDate** (see item 44) to validate the first date (date from) in a dialog box. If it is valid, then perform procedure **ValidDate** to validate the second date (date to) in a dialog box and performs Visual Basic function **DateDiff** to ensure that the second date should be later than first date.

3. CheckFinMth(txtFinMth As Control) As Integer.

To perform procedure **CheckNumeric** (see item 5) to validate the financial month in a dialog box. It also ensures that the month should be between 1 and 12.

4. CheckFinYear(txtFinYr As Control) As Integer.

To perform procedure **CheckNumeric** (see item 5) to validate the financial year in a dialog box and ensure that the year should be greater than zero.

5. CheckNumeric(textBox As Control).

To perform Visual Basic **IsNumeric** to check whether the data entered in the text box is a numeric data.

6. Close\_DB(dbPorc As Integer).

To close all the loaded dialog boxes of this application. It then performs procedure **Logout** (see item 31) to logout from the SyBase server.

7. ConvertString(txtBox As Control).

To convert a character (**'**) of a string to (**`**).

8. DaysAfter(aDate As String, daysNo As Integer) As String.

To generate a new date that is a number of days (given) after a date (given). If the new date falls on Sunday, the system will add one more day after the new date.

9. DispFinYrMth(aForm As Form, textBox As Control).

To generate the default financial year and month from a date entered in the dialog box.

10. Drop\_table(tableName As String).

To drop a temporary table from the database.

11. ExecApp(execFile As String).

To perform the Windows function **WinExec** to execute a given application and check the error code if the application is unable to run.

12. Get\_ExchRate(currCode As String).

To retrieve the exchange rate of a currency from the database table tblCurrency.

13. Get\_Help(HelpTopic As Integer).

To perform Windows function **WinHelp** to run the Windows Help or the On-line user's guide of this system.

14. GetFunction(szCmd As String, szEnable As String, szDisable As String) As Integer.

To retrieve records from database table tblFunction for setting the accessible functions to the user.

15. GetMaxCode(sqlStr As String) As Integer.

To get the last generated statement number from the related database table (ie. last generated reference number of invoice in table tblInvRecd).

Note : Parameter **sqlStr** is the given SQL command.

16. GetMaxNo() As String.

To retrieve the last generated sequence number from database table tblPettyCash.

17. GetParameter(inHeader As String, inString As String, retString As String).

To perform Windows function **GetPrivateProfileString** to retrieve the parameter in file **'FCSDB.INI'**.

18. GetSupplier(szSup As String).

To retrieve the detailed information of a selected supplier from database table Supplier.

19. Hide\_InvLst(A\_form As Form).

To set the label of the cash payment text box in a dialog box to be invisible.

20. LoadClaim(lstRef As Control, szType As String).

To retrieve claims records from database table tblClaim and display them in Reference list box.

21. LoadCostC(listBox As Control, whereStr As String).

To retrieve cost centres from database table tblCostC and display them in the cost centre list box.

22. LoadCurrency(listBox As Control).

To retrieve all currency records from database table tblCurrency and display them in the currency list box.

23. LoadProfSoc(listBox As Control).

To retrieve all professional society records from database table tblProfSoc and display them in the Society list box.

24. LoadRec(szCmd As String, listBox As Control).

To retrieve records from database table according to the given SQL command and display them in a list box.

25. LoadSelVote(listBox As Control, aCostC As String).

To retrieve vote records of a selected cost centre from database table tblVote and display them in the Vote list box.

26. LoadStaff(listBox As Control).

To retrieve all staff records from database table tblStaffInfor and display them in the Staff information list box.

27. LoadSupplier(szCmd As String, listBox As Control).

To retrieve all supplier records from database table Supplier and display them in the Supplier list box.

28. LoadVote(listBox As Control, whereStr As String).

To retrieve vote records of a selected cost centre from database table tblVote and display them in Vote list box.

29. Login(dbproc As Integer) As Integer.

To perform the DLL function **SqlLogin** (see item ) to login to the Sybase server.

30. Logout(dbproc As Integer) As Integer.

To perform the DLL function **sybclose** (see item ) to logout (disconnect) from the Sybase server.

31. Main().

This is the main function to start the application. It loads the MainForm (see Section 4.17) and performs procedure **Open\_db** (see item 32) for user to login to SyBase server.

32. Open\_db(dbproc As Integer).

To load the OpenDB form for user to enter his/her user id and password. Then the user will be able to access the server. It also performs procedure **SetApplnMenu** (see item 37) to setup the application window.

33. Print\_Subs\_List().

To perform procedure **StartPrinter** (see item 40) to use the HP Laser printer. Retrieve claims records from database tblSubscription and prints the claims letter to Bursar's Office. The procedure **Sub\_List\_Header** (see item 41) is performed to print the heading when the letter is more than a page.

34. recordCount(szCmd As String) As Integer.

To retrieve the number of records in a given database table.

35. Remove\_Items\_From\_Sysmenu(A\_form As Form).

To perform Windows function **GetSystemMenu** to get a window's system menu for modifying. Then perform another function **RemoveMenu** to remove all the menu items except **MOVE** and **CLOSE** command.

36. roundUp(amt As Double, amtFormat As String) As Double.

To round up a given amount based on the given format.

37. SetApplnMenu().

To perform procedure **GetFunction** (see item 14) to get the disabled and enabled items. Then the menu command **Funds** **Transfer** and **Budget** of **Funds** menu will be disabled or enabled. This is also applied to the **Update** menu in the **JournalAdj** form (refer to Section 4.16).

38. SetDate(aYear As Integer, aYear As Integer) As String.

To set a date to the format of ***'dd/mm/yy'*** based on the given year and month.

39. SqlCmd(dbproc As Integer, szCmdStr As String) As Integer.

This is the procedure for calling the DLL function **SqlCommand** to access the database tables.

40. StartPrinter(printerName As String).

To perform procedure **GetParameter** (see item 17) to get the printer device and perform Windows function **WriteProfileString** to replace the printer device in file WIN.INI.

41. Sub\_List\_Header().

To print the report header of the Claims for Subscription to Professional Societies report.

42. Supplier\_Dlg(suppStat As Integer).

To load the SuppFrm form (refer to Section 4.28).

43. UpdRefNo(aForm As Form, aFinYr As String, aRef As Integer) As Integer.

To update the last generated reference number of invoice in database table tblCtrl.

44. ValidDate(txtDate As Control) As Integer.

To check whether the date is entered as ***'dd/mm/yy'*** format and the month is between 1 and 12. It also ensures that the day is between 1 and the number of days of the month.

45. ValidEntry(aForm As Form, txtBox As Control, txtMsg As String) As Integer.

To validate entries in some given text boxes and ensure that valid data is entered.

46. ValidFunds(aForm As Form) As Integer.

To validate the exchange rate, total commitment (ie. Invoice amount, P/O amount & etc) and vote committed amount by using procedure **CheckNumeric** (see item 5). It also checks whether the total committed amount of votes is equal to the total commitment.

47. ValidPaidDate(aForm As Form, pdMonth As String, pdYear As String, pymtDate As String) As Integer.

To check whether the payment date is entered as ***'mm/yy'*** format and the month is between 1 and 12. It also reformat the payment data the format as ***'dd/mm/yy'***.

48. Visible\_InvLst(A\_form As Form).

To set the label of the cash payment text box in a dialog box to be visible.

49. WrtDelInv(aForm As Form, aInv As String) As Integer.

To insert a deleted invoice record in database table delInvRecd before deleting the invoice from table tblInvRecd.

4.35 FCSAP.GBL Module.

The is a module to define all the global variables. It also declares Windows and DLL functions to be called by this system.

**Note:** No procedure is defined in this module.

4.36 FCSBUDGET.BAS Module.

The is the second module to store the common procedures to be shared among the forms.

4.36.1 Procedures :

1. AdjBudget(aFinYr As String) As Integer.

To update the adjusted vote commitment and the balance of the selected records in database table tblBudget.

2. CalVoteAmt(aForm As Form, exchRate As Double, paidAmt As Double).

To allocate the payment amount (in S$) of committed votes (shown in the funds commitment list box) based on the total cash payment.

3. Display\_Trsf(gridDate As Control).

To perform procedure **Get\_Trsf** (see item 5) to display the funds transfer records on the screen.

4. Get\_JAdj(gridDate As Control, tblName As String, whereStr As String).

To retrieve journal adjustment records from database table tblJAdj or tblJAdjM (colType = 'J') and display them on the screen.

5. Get\_Trsf(gridDate As Control, whereStr As String).

To retrieve funds transfer records from database table tblJAdjM (colType = 'T') and display them on the screen.

6. JAdj\_Header().

To print the report header of Journal Adjustment Transactions report.

7. Print\_JAdj(gridDate As Control).

To perform procedure **StartPrinter** (see Section 4.34.1 item 40) to use the Epson dot-matrix printer. Then print the details of journal adjustment transactions shown in the cells to printer.

8. Print\_Trsf(gridDate As Control).

To perform procedure **StartPrinter** (see Section 4.34.1 item 40) to use the Epson dot-matrix printer. Then print the details of journal adjustment transactions shown in the cells to printer.

9. SetJAdj\_header(gridDate As Control).

To set the heading at the first row for displaying journal adjustment transactions on the screen.

10. SetTrsf\_header(gridDate As Control).

To set the heading at the first row for displaying funds transfer transactions on the screen.

11. Trsf\_Header().

To print the report header of Fund Transfer Transactions report.

12. UpdBudget(voteAC As String, voteNo As String, aFinYr As String, szStr As String) As Integer.

To update/insert records in database table tblBudget.

13. Valid\_Pymt(aForm As Form, pdYear As String, pdMonth As String, pymtDate As String, oldPymt As String) As Integer.

To perform procedure **ValidPaidAmt** (see item 14). If the payment amount is valid, generate the payment date in ***'dd/mm/yy'*** format according to the date entered in ***'mm/yy'*** format.

14. ValidPaidAmt(aForm As Form, exchRate As Double) As Integer.

To perform procedure **CheckNumeric** (see Section 4.34.1 item 5). Compare the payment and the invoice amount, the system does not accept the payment if it is more than invoice amount. Otherwise, a warning message is displayed and the payment will be accepted.

15. View\_JAdj(gridDate As Control, tblName As String, JType As String).

To reformat the grid control cells and perform procedure **Get\_JAdj** (see item 4) to display the journal adjustment records in the cells.

16. WrtPrevBudget(formNo As String, aFinYr As String) As Integer.

To store the total committed amount (in S$) of a selected vote from the database table tblVoteCommit into a temporary table #t1. Please note that this table #t1 should be generated before updating table tblVoteCommit and it will be used to update the database table tblBudget later.

4.37 FCSFUND.BAS Module.

The is the third module to store the common procedures to be shared among the forms.

4.37.1 Procedures :

1. AdjustedFunds(aForm As Form, szMsg As String) As Integer.

To compute list of vote commitment (stored in global variable ***voteArray***) that should be adjusted based on the amount of credit/debit note and percentage.

2. CalAmount(aForm As Form).

To compute the *total commitment* (i.e. amount of P/O, invoice, claims & etc.) in S$ equivalent based on the given exchange rate.

3. CalFunds(aForm As Form).

To compute the vote committed amount in S$ equivalent based on the given exchange rate.

4. CalPercent(aForm As Form).

To compute the committed amount of the selected vote based on the *total commitment* (i.e. amount of P/O, invoice, claims & etc.) and the entered percentage. The percentage will be computed by the system if the computed vote commitment is different from the commitment entered by the user.

5. CheckFund(funds As Double, voteAC As String, aForm As Form).

To retrieve the current balance of a vote from database table tblVoteFunds and check whether there is sufficient funds to be committed.

6. DelVoteAC(formNo As String) As Integer.

To insert the deleted vote commitment records from database table tblVoteCommit to table delVoteCommit before deleting the records from table tblVoteCommit.

7. DisplayFunds(formNo As String, colType As String, ListBox As Control, aForm As Form).

To retrieve votes committed of a selected *form number* (i.e. invoice, P/O, Claims & etc.) from the database table tblVoteCommit. Display them in the funds commitment list box.

8. genVoteArray(aForm As Form).

To store a list of committed votes from the funds commitment list box to the global variable ***voteArray***.

9. GetPercent(aForm As Form).

To perform procedure **CheckNumeric** (see Section 4.34.1 item 5) to validate the entered percent. If it is valid, compute the committed amount of the selected vote based on the given percentage. It also performs procedure **CalFunds** (see item 3).

10. ResetAddCmd(aForm As Form, textBox As Control).

To set the **Add** button to be visible and enable **OK** & **Cancel** button in a dialog box.

11. SaveFunds(voteAC As String, voteNo As String, amtS As String, aForm As Form) As Integer.

To insert/update the committed amount of record in database table tblVoteFunds. Perform procedure **UpdBudget** (see Section 4.36.1 item 12) to update the committed amount in database table tblBudget.

12. SaveVoteAC(aForm As Form, frmNo As String, aPO As String, aDate As String) As Integer.

To insert records in database table tblVoteCommit where the records shown in the funds commitment list box.. It also performs procedure **UpdPrePymt** (see item 17) if this vote commitment is for an invoice of a pre-paid P/O. Procedures **UpdCommitInv** (see item 14) and **UpdFundsAdj** (see item 15) will be performed when this vote commitment is for Credit/Debit note of an Invoice. Otherwise, perform procedure **SaveFunds** (see item 11).

13. updateVoteFunds(oldFinMth As String, oldFinYr As String, frmNo As String) As Integer.

To update the cash payment and amount committed of record in database table tblVoteFunds. It also performs procedures **AdjBudget** (see Section 4.36.1 item 1) and **DelVoteAC** (see item 6).

14. UpdCommitInv(voteAC As String, voteNo As String, amt As String, amtS As String, frmNo As String, aInv As String, aFunction As String) As Integer.

To update the cancelled amount of record in database tables tblVoteCommit where the record is the vote commitment of an invoice and the cancelled amount is the amount of a credit/debit note.

15. UpdFundsAdj(voteAC As String, voteNo As String, amtS As String, szForm As Form) As Integer.

To update the funds adjustment of a record in database tables tblVoteCommit and tblVoteFunds. It also performs procedure **UpdBudget** (see Section 4.36.1 item 12) to update the records in database table tblBudget.

16. UpdFundsPymt(szForm As Form, formNo As String, finMth As String, finYr As String, pymtAmt As Double, exchDiff As Double, oldPymt As String) As Integer.

To update the cash payment of records in database tables tblVoteCommit and tblVoteFunds where the records are shown in the funds commitment list box.

17. UpdPrePymt(voteAC As String, voteNo As String, frmNo As String, aPO As String) As Integer.

To update the cash payment of a record in database table tblVoteCommit. This is applied to the vote commitment of an invoice received for a pre-paid purchase order.

18. UpdVoteAC(szForm As Form, frmNo As String, aPO As String) As Integer.

To update the records in database table tblVoteCommit where the records are shown in the funds commitment list box. It also performs procedure **SaveFunds** (see item 11).

# 5. DATA BASE INTERFACE SOFTWARE DESIGN.

To access to the SyBase database, the FCS A/P VISUAL BASIC programs must call the C routines of a DLL program *DbSever.DLL*. This program provides the functions such as: logging into the server, sending SQL commands to the server, processing the results returned from the server, handling the DB-Library errors and server messages, and closing the connection with the server. Each of the C routines and the SyBase DB-Library functions are defined Section 5.1. Listing of DbServer.DLL is given in Programmer's Manual Supplement.

5.1 C Routines.

5.1.1 SqlStart(void).

This is the user created functions that uses the Sybase:

5.1.2 SqlCommand(dbproc, LPSTR cmdstring).

It adds text to DBPROCESS command buffer, sends it to the server and waits for a response from the server and verify the correctness of the instructions of the server is responding to. The DB-Library functions used in this routine are defined as below :

DB-Library

function Explanation

dbcmd() To add text (SQL statement) to the DBPROCESS command buffer.

dbsqlsend() To send a command batch to SQL server for execution without waiting for a response.

dbsqlok() To wait for the results from the server and verify the correctness of the commands the server is responding to.

5.1.3 SqlExist(dbproc, LPSTR cmdstring).

It checks whether a particular record exists in the database table.

5.1.4 SqlLogin(dbproc, login, option).

It allocates a login record structure **LOGINREC**, then it creates and initialises a DBPROCESS structure. It also establishes the database required by the user.

5.1.5 sybabsolute(dbproc).

It returns the absolute address of the dbproc The DB-Library function **dbabsolute()** is used in this routine.

5.1.6 sybadata(dbproc, int computeid, int column).

It returns a pointer to the data for a compute column The DB-Library function **dbadata()** is used in this routine.

5.1.7 sybadlen(dbproc, int computeid, int column).

It returns the actual length of the data for a compute column. The DB-Library function **dbadlen()** is used in this routine.

5.1.8 sybaltbind(dbproc, int computeid, int column, int vartype, DBINT varlen, BYTE far \* varaddr).

It binds a compute column to a program variable. The DB-Library function **dbaltbind()** is used in this routine.

5.1.9 sybaltcolid(dbproc, int computeid, int column).

It returns the column id of the data for a compute column. The DB-Library function **dbaltcolid()** is used in this routine.

5.1.10 sybaltlen(dbproc, int computeid, int column).

It returns the maximum length of the data for a particular compute column. The DB-Library function **dbaltlen()** is used in this routine.

5.1.11 sybaltop(dbproc, int computeid, int column).

It returns the type of aggregate operator for a particular compute column. The DB-Library function **dbaltop()** is used in this routine.

5.1.12 sybalttype(dbproc, int computeid, int column).

It returns the data type for a compute column. The DB-Library function **dbalttype()** is used in this routine.

5.1.13 sybbind(dbproc, int computeid, int column, int vartype, DBINT varlen, BYTE far \* varaddr).

It binds a regular result column to a program variable. The DB-Library function **dbbind()** is used in this routine.

5.1.14 sybbylist(dbproc).

It returns the bylist for a compute row. The DB-Library function **dbbylist()** is used in this routine.

5.1.15 sybcancel(dbproc).

It cancels the current command batch. The DB-Library function **dbcancel()** is used in this routine.

5.1.16 sybcanquery(dbproc).

It cancels any rows pending from the most recently executed query. The DB-Library function **dbcanquery()** is used in this routine.

5.1.17 sybchange(dbproc).

It determines whether a command batch has changed the current database. The DB-Library function **dbchange()** is used in this routine.

5.1.18 sybclose(dbproc).

It closes the server connection and de-allocates DBPROCESS structures. The DB-Library function **dbclose()** is used in this routine.

5.1.19 sybclrbuff(dbproc, DBINT n).

It drops rows from the row buffer DBPROCESS structures. The DB-Library function **dbclrbuff()** is used in this routine.

5.1.20 sybclropt(dbproc, int option, LPSTR param).

It cleans an option set by **dbsetopt()** or **sybsetopt()**. The DB-Library function **dbclropt()** is used in this routine.

5.1.21 sybcmd(dbproc, LPSTR cmdstring).

It adds text to the **DBPROCESS** command buffer. The DB-Library function **dbcmd()** is used in this routine.

5.1.22 SYBDBCMDROW(dbproc).

It determines whether the current command can return rows. The DB-Library function **DBCMDROW()** is used in this routine.

5.1.23 sybcolbrowse(dbproc, int colnum).

It determines whether the source of a regular result column is updatable via the DB-Library browse mode facilities. The DB-Library function **dbcolbrowse()** is used in this routine.

5.1.24 sybcollen(dbproc, int computeid, int column).

It returns the maximum length of the data in a regular result column. The DB-Library function **dbcollen()** is used in this routine.

5.1.25 sybcolname(dbproc, int computeid, int column).

It returns the name of a regular result column. The DB-Library function **dbcolname()** is used in this routine.

5.1.26 sybcolsource(dbproc, int computeid, int column).

It returns a pointer to the name of the database column from which the specified regular result column was derived. The DB-Library function **dbcolsource()** is used in this routine.

5.1.27 sybcoltype(dbproc, int computeid, int column).

It returns data type for a regular result column. The DB-Library function **dbcoltype()** is used in this routine.

5.1.28 sybcolutype(dbproc, int computeid, int column).

It returns user defined data type for a regular result column. The DB-Library function **dbcolutype()** is used in this routine.

5.1.29 sybconvert(dbproc, int srctype, BYTE dar \*srcm DBINT srclen, int desttype, BYTE far \*dest, DBINT destlen).

It converts data from one data type to another. The DB-Library function **dbconvert()** is used in this routine.

5.1.30 SYBCOUNT(dbproc).

It returns the number of rows affected by a Transact-SQL command. The DB-Library function **DBCOUNT()** is used in this routine.

5.1.31 SYBCURCMD(dbproc).

It returns the number of the current command. The DB-Library function **DBCURCMD()** is used in this routine.

5.1.32 SYBCURROW(dbproc).

It returns the number of the row currently being read. The DB-Library function **DBCURROW()** is used in this routine.

5.1.33 sybdata(dbproc, int computeid, int column).

It returns a pointer to the data in a regular result column The DB-Library function **dbdata()** is used in this routine.

5.1.34 sybdatlen(dbproc, int computeid, int column).

It returns the actual length of the data in a regular result column. The DB-Library function **dbdatlen()** is used in this routine.

5.1.35 SYBDEAD(dbproc).

It determines whether a particular DBPROCESS is dead. The DB-Library function **DBDEAD()** is used in this routine.

5.1.36 sybexit(void).

It closes and de-allocates all DBPROCESS structures and cleans up any structures initialized by dbinit() or sybinit(). The DB-Library function **dbexit()** is used in this routine.

5.1.37 SYBFIRSTROW(dbproc).

It returns the number if the first row in the row buffer. The DB-Library function **DBFIRSTROW()** is used in this routine.

5.1.38 sybfreebuf(dbproc).

It clears the command row buffer. The DB-Library function **dbfreebuf()** is used in this routine.

5.1.39 sybfreelogin(LOGINREC NEAR \*loginptr).

It is to free a login record. The DB-Library function **dbfreelogin()** is used in this routine.

5.1.40 sybfreequal(LPSTR qualptr).

It is to free a login record. The DB-Library function **dbfreequal()** is used in this routine.

5.1.41 sybgetchar(dbproc, int n).

It returns a pointer to a character in the command buffer. The DB-Library function **dbgetchar()** is used in this routine.

5.1.42 sybgetmaxprocs(void).

It determines the current maximum number of simultaneously open DBPROCESSes. The DB-Library function **dbgetmaxprocs()** is used in this routine.

5.1.43 sybgetoff(dbproc, DBSMALLINT offtype, int startfrom).

It checks for the existence of Transact-SQL constructs in the command buffer. The DB-Library function **dbgetoff()** is used in this routine.

5.1.44 sybgetrow(dbproc, DBINT row).

It returns the specified row in the row buffer. The DB-Library function **dbgetrow()** is used in this routine.

5.1.45 SYBGETTIME(void).

It returns the number of seconds that DB-Library will wait for a server response to a SQL command. The DB-Library function **DBGETTIME()** is used in this routine.

5.1.46 sybinit(void).

It initialises certain private DB-Library structures by using the DB-Library function **dbinit()**.

5.1.47 SYBISAVAIL(void).

It determines whether a DBPROCESS is available for general use. The DB-Library function **DBISAVAIL()** is used in this routine.

5.1.48 SYBISOPT(dbproc, int option, LPSTR param).

It checks the status of a server or DB-Library option. The DB-Library function **dbisopt()** is used in this routine.

5.1.49 SYBLASTROW(dbproc).

It returns the number of the last row in the row buffer. The DB-Library function **DBLASTROW()** is used in this routine.

5.1.50 syblocklib(void).

It locks the DBPROCESS. The DB-Library function **dblocklib()** is used in this routine.

5.1.51 syblogin(dbproc, login, option).

It allocates a login record for use in dbopen() or sybopen() and returns a pointer to a **LOGINREC** structure. The DB-Library function **dblogin()** is used in this routine.

5.1.52 SYBMORECMDS(dbproc).

It indicates whether there are more commands to be processed. The DB-Library function **DBMORECMDS()** is used in this routine.

5.1.53 sybmoretext(dbproc, DBINT size, BYTE far \* text).

It sends part of a text or image value to the server. The DB-Library function **dbmoretext()** is used in this routine.

5.1.54 sybname(dbproc).

It returns the name of the current database. The DB-Library function **dbname()** is used in this routine.

5.1.55 sybnextrow(dbproc).

It reads the next result row. The DB-Library function **dbnextrow()** is used in this routine.

5.1.56 sybnumalts(dbproc, int computeid).

It returns the number of columns in a compute row. The DB-Library function **dbnumalts()** is used in this routine.

5.1.57 sybnumcols(dbproc).

It determines the number of regular columns for the current set of results. The DB-Library function **dbnumcols()** is used in this routine.

5.1.58 sybnumcompute(dbproc).

It returns the number of COMPUTE clauses in the current set of results. The DB-Library function **dbnumcompute()** is used in this routine.

5.1.59 SYBNUMORDERS(dbproc).

It returns the number of columns specified in a Transact-SQL SELECT statement's ORDER BY clause. The DB-Library function **DBNUMORDERS()** is used in this routine.

5.1.60 sybnumcompute(dbproc).

It returns the number of COMPUTE clauses in the current set of results. The DB-Library function **dbnumcompute()** is used in this routine.

5.1.61 sybopen(LOGINREC NEAR \* login, LPSTR lpServer).

By using the DB-Library function **dbopen()**, this routine creates and initialises a DBPROCESS structure and returns a DBPROCESS pointer.

5.1.62 sybordercol(dbproc, int order).

It returns the id of a column appearing in the most recently executed query's ORDER BY clause. The DB-Library function **dbordercol()** is used in this routine.

5.1.63 sybprtype(int token).

It converts a token value to a readable string. The DB-Library function **dbprtype()** is used in this routine.

5.1.64 sybqual(dbproc, int tabnum, LPSTR tabname).

It returns a pointer to a WHERE clause suitable for use in updating the current row in a browsable table. The DB-Library function **dbqual()** is used in this routine.

5.1.65 sybreadpage(dbproc, LPSTR dbnameint, DBINT pageno, BYTE far \* buf).

It reads a page of binary data from the server and returns the number of bytes read. The DB-Library function **dbreadpage()** is used in this routine.

5.1.66 sybresults(dbproc).

It sets up the results of the next query. The DB-Library function **dbresults()** is used in this routine.

5.1.67 SYBROWS(dbproc).

It indicates whether the current command actually returned rows. The DB-Library function **DBROWS()** is used in this routine.

5.1.68 SYBROWTYPE(dbproc).

It returns the type of the current row. The DB-Library function **DBROWTYPE()** is used in this routine.

5.1.69 sybsetavail(dbproc).

It makes a DBPROCESS as being available for general use. The DB-Library function **dbsetavail()** is used in this routine.

5.1.70 SYBSETLAPP(LOGINREC NEAR \* loginrec, LPSTR application).

It sets the application name in the LOGINREC structure by using the DB-Library function **DBSETLAPP()** is used in this routine.

5.1.71 SYBSETHOST(LOGINREC NEAR \* loginrec, LPSTR hostname).

It sets the host name in the **LOGINREC** structure by using the DB-Library function **DBSETLHOST()**.

5.1.72 sybsetlogintime(INT seconds).

It sets the length of time in seconds that DB-Library will wait for a server response for a DBPROCESS connection. The DB-Library function **dbsetlogintime()** is used in this routine. The return value nResult is either SUCCEED or FAIL.

5.1.73 SYBSETLPWD(LOGINREC NEAR \* loginrec, LPSTR password).

It sets the user server password in the **LOGINREC** structure by using the DB-Library function **DBSETLPWD()**. The password will be sent to the server.

5.1.74 SYBSETLUSER(LOGINREC NEAR \* loginrec, LPSTR username).

It sets the user's name in the **LOGINREC** structure by using the DB-Library function **DBSETLUSER()**.

5.1.75 sybsetmaxprocs(int maxprocs).

It sets the maximum number of simultaneously open DBPROCESSes. The DB-Library function is **dbsetmaxprocs()**.

5.1.76 sybsetnull(dbproc, int bindtype, int bindlen, BYTE far \* bindval).

It defines substitution value to be used when binding null values. The DB-Library function is **dbsetnull()**.

5.1.77 sybsetopt(dbproc, int option, LPSTR char\_param).

It defines substitution value to be used when binding null values. The DB-Library function is **dbsetopt()**.

5.1.78 sybsettime(INT sec).

By using the DB-Library function **dbsettime()**, it sets the length of time in seconds that DB-Library will wait for a SQL server response to a SQL command. The return value nResult is either SUCCEED or FAIL.

5.1.79 sybsqlexec(dbproc).

It sends a command batch to the server. The DB-Library function is **dbsqlexec()**.

5.1.80 sybsqlok(dbproc).

It waits for results from the server and verifies the correctness of the instructions the server is responding to. The DB-Library function is **dbsqlok()**.

5.1.81 sybsqlsend(dbproc).

It sends a command batch to the server and does not wait for a response. The DB-Library function is **dbsqlsend()**.

5.1.82 sybstrcpy(dbproc).

It copies a portion of the command buffer. The DB-Library function is **dbstrcpy()**.

5.1.83 sybstrlen(dbproc).

It returns the length in characters of the command buffer. The DB-Library function is **dbstrlen()**.

5.1.84 sybunlocklib(void).

It unlocks DB-Library which is locked by dblocklib(). The DB-Library function is **dbunlocklib()**.

5.1.85 sybuse(dbporc, LPSTR dbname).

By using the DB-Library function **dbuse()**, it sets the current database.

5.1.86 sybwillconvert(int srctype, int desttype).

It determines whether a specific data type conversion is available within DB-Library. The DB-Library function is **dbwillconvert()**.

5.1.87 sybwillexit(int srctype, int desttype).

It closes and de-allocates all DBPROCESS structures and cleans up any structures initialized by dbint() or sybinit(). The DB-Library function is **dbwillexit()**.

## 5.2 Generating the DLL program DbServer.DLL.

For information on how to generate the DLL program, please refer the FCS Invoices/Accounts Receivable Programmer's Manual.

# 6. CONFIGURATION INFORMATION.

There are two SyBase software programs : Net-Library and DB-Library. These should be installed before accessing the SyBase database through Windows. The procedures for SyBase Net-Library and DB-Library installation, special requirements for the AUTOEXEC.BAT and SYSTEM.INI files have been described in the Programmer's manual of FCS Invoices/Accounts Receivable model.

## 6.1 FCS A/P Software Installation.

The section defines the necessary files to be installed on both programmer's PC and user's PC.

### 6.1.1 Files Installed On User's PC.

1. Files should be kept in the directory **C:\FCSAP**.

FCSAPP.EXE (FCS A/P program executed file)

DBSERVER.DLL (DLL program)

DBEMSG.DLL (DLL program)

FCS2USR.HLP (On-line user's guide)

2. Files should be kept in the directory **C:\WINDOWS|SYSTEM**.

CMDIALOG.VBX

GRID.VBX

MSMASKED.VBX

THREED.VBX

MSAFINX.DLL

### 6.1.2 Files Installed On Programmer's PC.

Besides having all the files that are installed on the user's PC, the following softwares should be included on the programmer's PC.

VISUAL BASIC software in the directory **C:\VB**.

FCS A/P sourse codes in the directory **C:\FCSAP**.

|  |  |  |
| --- | --- | --- |
| Approval Record ISS/Forms/Approval | | |
| Project Name Financial Control System (FCS) | | |
| **Document Ref** ISS**/**FCS2/TS.3/1 |  | |
| **Approved by** | **Date** | |
| **Authorised by** | **Date** | |
| The document authorisation appears on the title page. | |  |
| The structure of the publication is logical. | |  |
| The distribution list is correct. | |  |
| The title page is signed. | |  |
| Calculations appear reasonable, are neatly presented and have been checked. | |  |
| Theory and formulae are correct and properly applied. | |  |
| Illustrations are relevant, readable and logically placed. | |  |
| There are no typographical errors. | |  |
| Units are consistent throughout. | |  |
| The security classification is correct. | |  |
| There are no obvious omissions. | |  |
| The document complies with the Client's requirements, however specified. | |  |
| Responsibility is accepted for all opinions, conclusions and recommendations. | |  |
| The document does not run counter to company policy. | |  |
|  | |  |
|  | |  |
|  | |  |
|  | |  |