**5. DETAILED DESIGN**

**5.1 Introduction**

During detailed design, the internal logic of each modules specified in system design is decided. During this phase further details of the modules are decided. Design of each of the modules usually specified in a high level description language which is independent of the language in which software eventually be implemented

**5.2 Structure of software package**

**COLLEGE ACTIVITY MANAGEMENT SYSTEM**

Login

Admin

Staff

View Activity

View Attendance

Manage Association

Add Association

Update Association

Delete Association

Manage Staff

Add Staff

Update Staff

Delete Staff

Manage Student

Add Student

Update Student

Delete Student

View Activity

Manage Activity

Edit Profile

Add Activity

Update Activity

Delete Activity

Record Attendance

**Fig(3.3.1) System Software Architecture**

**5.3 Module decomposition of software**

**Structure chart:**

Structure chart is a top-down modular design, consist of squares representing different models in a system and lines .Structure chart shows how program has been partitioned into manageable modules hierarchy and organization of those modules and communicational interface.

|  |  |  |
| --- | --- | --- |
| Symbol | Name | Process |
|  | Data flow | Show the direction flow of data. |
|  | Control flow | Show the direction of flow control. |
|  | Processing | Shows manipulation, Calculation and processing. |
|  | Module Invocation | It represents of subordinate module being invoked by superior ordinate module |
| A  B  C  Main | Condition invocation | It indicates that the invocation of subordinate’s module depends on the evaluation of a condition |
| A  B  Main | Iteration | It represents the iteration. |

**Table (5.1) Structure art**

**Flow chart:**

Flow chart is a graphical representation of solution to the given problems. A Flowchart is pictorial representation of an algorithm, workflow or process. The diagrammatic representation illustrates a solution model to given problem. It uses the following symbol

|  |  |  |
| --- | --- | --- |
| Symbol | Name | Purpose |
|  | Terminator | It indicates the start and end process. |
|  | Input/Output | Input/Output data. |
|  | Decision | It represents a comparison or question that determines an alternate path to be followed. |
|  | Flow direction | Shows the direction of data flow |
|  | Processing | It represents manipulation, calculation or information processing |
|  | Direction access storage | File storage |
|  | Preparation(Looping) | An instruction or group of instruction |

**Table (5.2) Flow chart**

**5.3.1 Login modules**

**5.3.1.1 Inputs:**

Username, Password

**5.3.1.2 Procedural details**

**Algorithm:**

Step 1: Start

Step 2: input username and password

Step 3: if input==admin then

If username and password are valid then

Display admin home page

Else

Display error message

End if

Else if input==staff then

If username and password are valid then

Display staff page

Else

Display error message

End if

End if

Step 4: End

**Structure chart:**

Login

Authentication

Input user id or password

Display Successful message

Read Username and password

Display Admin or Staff page

Display invalid user or password message

**Figure(5.2)Login**

**5.3.1.3 File I/O interface**

Admin, Staff

**5.3.1.4 Outputs**

After successful authentication respective page will be loaded

**5.3.1.5 Implementation aspects**

* Textbox
* Label
* Button

**5.3.2 Admin**

**5.3.2.1 View Activity**

**5.3.2.1.1 Input**

Button click

**5.3.2.1.2 Procedural details**

**Flow chart:**

start

Input button\_click

put OTP

activity

Retrieve activity information

Display activities

Stop

**5.3.2.1.3 File I/O interface**

Activity

**5.3.2.1.4 Outputs**

Display student Activity info to admin.

**5.3.2.1.5 Implementation aspects**

* Textbox
* Button
* Label
* Dropdown

**5.3.2.2 View Attendance**

**5.3.2.2.1 Input**

Button click

**5.3.2.2.2 Procedural details**

start

activity

Retrieve attendance

information

**Flow chart:**

Input button\_click

put OTP

Display acttendance

Stop

**5.3.2.2.3 File I/O interface**

Attendance

**5.3.2.2.4 Outputs**

Display student attendance info to admin.

**5.3.2.2.5 Implementation aspects**

* Textbox
* Button
* Label
* Dropdown

**5.3.2.3 Manage association**

**5.3.2.3.1 Add association**

**5.3.2.3.1.1 Input**

Association details

**5.3.2.3.1.2 Procedural details**

**Algorithm:**

Step 1: Start

Step 2: Input association details

Step 3: Read the input

Step 4: If the input is valid THEN

IF the association already exists

GOTO step 5

ELSE

GOTO Step 6

END IF

ELSE

Display Error message

GOTO step 7

END IF

Step 5: “Association already exists” message display.

Display add association page

**5.3.2.3.1.3 File I/O interface**

Association

**5.3.2.3.1.4 Outputs**

Store the association details to database

**5.3.2.3.1.5 Implementation aspects**

* Textbox
* Label
* Button

**5.3.2.3.2 Update Association**

**5.3.2.3.2.1 Input**

Association id

**5.3.2.3.2.2 Procedural details**

**Flow chart:**

Start

Association

Input button\_click

put OTP

Load Association detail

Read new detail

Staff

Update in database

Display successful message

put OTP

Stop

**5.3.2.3.2.3 File I/O interface**

Association

**5.3.2.3.2.4 Outputs**

Update the association details to database

**5.3.2.3.2.5 Implementation aspects**

* Textbox
* Label
* Button
* Dropdown

**5.3.2.3.3 Delete Association**

**5.3.2.3.3.1 Input**

Association id

**5.2.2.2.2.2 Procedural details**

**Structure chart:**

Display successful message

Delete

Button click

Delete from database

Read Association detail

**5.2.2.2.2.3: File I/O interface**

Association

**5.2.2.2.2.4 Outputs**

Delete the association details to database

**5.2.2.2.2.5 Implementation aspects**

* Textbox
* Label
* Button
* Dropdown

**5.3.2.3 Manage Staff**

**5.3.2.3.1 Add Staff**

**5.3.2.3.1.1 Input**

Staff details

**5.3.2.3.1.2 Procedural details**

**Algorithm:**

Step 1: Start

Step 2: Input Staf details

Step 3: Read the input

Step 4: If the input is valid THEN

IF the Staff already exists

GOTO step 5

ELSE

GOTO Step 6

END IF

ELSE

Display Error message

GOTO step 7

END IF

Step 5: “Staff already exists” message display.

Display Staff association page

**5.3.2.3.1.3 File I/O interface**

Staff

**5.3.2.3.1.4 Outputs**

Store the Staff details to database

**5.3.2.3.1.5 Implementation aspects**

* Textbox
* Label
* Button

**5.3.2.3.2 Update Staff**

**5.3.2.3.2.1 Input**

Staff id

**5.3.2.3.2.2 Procedural details**

**Flow chart:**

start Start

Input button\_click

put OTP

Staff

Load staff detail

Read new detail

Staff

Update in database

Display successful message

put OTP

Stop

**5.3.2.3.2.3 File I/O interface**

staff

**5.3.2.3.2.4 Outputs**

Update the staff details to database

**5.3.2.3.2.5 Implementation aspects**

* Textbox
* Label
* Button
* Dropdown

**5.3.2.3.3 Delete staff**

**5.3.2.3.3.1 Input**

Staff id

**5.3.2.3.3.2 Procedural details**

**Structure chart:**

Button click

Delete from database

Delete

Read staff detail

Display successful message

**5.3.2.3.3.3 File I/O interface**

Staff

**5.3.2.3.3.4 Outputs**

Delete the Staff details to database

**5.3.2.3.3.5 Implementation aspects**

* Textbox
* Label
* Button
* Dropdown

**5.3.3 Staff**

**5.3.3.1 View Activity**

**5.3.3.1.1 Input**

Button click

**5.3.3.1.2 Procedural details**

**Flow chart:**

start

Input button\_click

put OTP

Retrieve activity information

activity

Stop

Display activities

**5.3.3.1.3 File I/O interface**

Activity

**5.3.3.1.4 Outputs**

Display student Activity info to Staff.

**5.3.3.1.5 Implementation aspects**

* Textbox
* Button
* Label
* Dropdown

**5.3.3.2 Manage Student**

**5.3.3.2.1 Add Student**

**5.3.3.2.1.1 Input**

Student details

**5.3.3.2.1.2 Procedural details**

**Algorithm:**

Step 1: Start

Step 2: Input Student details

Step 3: Read the input

Step 4: If the input is valid THEN

IF the Student already exists

GOTO step 5

ELSE

GOTO Step 6

END IF

ELSE

Display Error message

GOTO step 7

END IF

Step 5: “Student already exists” message display.

Display add Student page

**5.3.3.2.1.3 File I/O interface**

Student

**5.3.3.2.1.4 Outputs**

Store the Student details to database

**5.3.3.2.1.5 Implementation aspects**

* Textbox
* Label
* Button

**5.3.3.2.2 Update Student**

**5.3.3.2.2.1 Input**

Student id

**5.3.3.2.2.2 Procedural details**

**Flow chart:**

start Start

Student

Input button\_click

put OTP

Load student detail

Read new detail

Student

Update in database

Display successful message

put OTP

Stop

**5.3.3.2.2.3 File I/O interface**

Student

**5.3.3.2.2.4 Outputs**

Update the Student details to database

**5.3.3.2.2.5 Implementation aspects**

* Label
* Textbox
* Button
* Dropdown

**5.3.3.2.3 Delete Student**

**5.3.3.2.3.1 Input**

Student id

**5.3.3.2.3.2 Procedural details**

Delete

Display successful message

Delete from database

Button click

Read student detail

**5.3.3.2.3.3 File I/O interface**

Student

**5.3.3.2.3.4 Outputs**

Delete the Student details to database

**5.3.3.2.3.5 Implementation aspects**

* Textbox
* Label
* Button
* Dropdown

**5.3.3.3 Record Attendance**

**5.3.3.3.1 Input**

Button click

**5.3.3.3.2 Procedural details**

**Algorithm:**

Step 1:start

Step 2:Input Button click

Step3: Read activity Id

Step 4:Load Student Attendance table

Step 5:Read attendance status

Step 6:Store attendance in database

Display Successful message

Step 7:Stop

**5.3.3.3.3 File I/O interface**

Student

**5.3.3.3.4 Outputs**

Record the Student Attendance to database

**5.3.3.3.5 Implementation aspects**

* Textbox
* Label
* Button
* Dropdown

**5.3.3.4 Manage Activity**

**5.3.3.4.1 Add Activity**

**5.3.3.4.1.1 Input**

Activity details

**5.3.3.4.1.2 Procedural details**

**Algorithm**:

Step 1: Start

Step 2: Input Activity details

Step 3: Read the input

Step 4: If the input is valid THEN

IF the Activity already exists

GOTO step 5

ELSE

GOTO Step 6

END IF

ELSE

Display Error message

GOTO step 7

END IF

Step 5: “Activity already exists” message display.

Display add Activity page

**5.3.3.4.1.3 File I/O interface**

Activity

**5.3.3.4.1.4 Outputs**

Store the Activity details to database

**5.3.3.4.1.5 Implementation aspects**

* Textbox
* Label
* Button

**5.3.3.4.2 Update Activity**

**5.3.3.4.2.1 Input**

Activity id

**5.3.3.4.2.2 Procedural details**

**Flow chart:**

Activity

Input button\_click

put OTP

Start

Load Activity detail

Read new detail

Update in database

Activity

Display successful message

put OTP

Stop

**5.3.3.4.2.3 Outputs**

Update the Activity details to database

**5.3.3.4.2.4 Implementation aspects**

* Textbox
* Label
* Button
* Dropdown

**5.3.3.4.3 Delete Activity**

**5.3.3.4.3.1 Input**

Activity id

**5.3.3.4.3.2 Procedural details**

**Structure chart:**

Delete

Display successful message

Button click

Delete from database

Read activity detail

**5.3.3.4.3.3 Outputs**

Delete the Activity details to database

**5.3.3.4.3.4 Implementation aspects**

* Textbox
* Label
* Button
* Dropdown

**5.3.3.5 Edit Profile**

**5.3.3.5.1 Input**

Button click

**5.3.3.5.2 Procedural details**

**Algorithm:**

Step 1: start

Step 2: fetch all the staff detail from table

Step 3: retrieve the staff detail from login

Step 4: update profile of staff from table

Step 5: display staff details

Step 6:stop

**5.3.3.5.3 File I/O interface**

Staff

**5.3.3.5.4 Outputs**

Edit the staff profile details to database

**5.3.3.5.5 Implementation aspects**

* Textbox
* Label
* Button