**Chapter 7**

**Project Planning & Scheduling**

|  |  |
| --- | --- |
|  |  |
|  |

Project planning and scheduling is used to show the project scheduling time. This project is specific, measurable, achievable, realistic and timed. So, we need planning and scheduling for this project.

**7.1 Function Specification:**

|  |  |
| --- | --- |
| **Name** | **Short Name** |
|  |  |
| User Management System | [F1] |
|  |  |
| Product Management System | [F2] |
|  |  |
| Brand Management System | [F3] |
|  |  |
| Category management System | [F4] |
|  |  |
| Order management System | [F5] |
|  |  |
| Purchase Management system | [F6] |
|  |  |
| View Report | [F7] |
|  |  |

**7.2 Function Point Estimation:**

In this project, Function point estimation is measured the amount of functionality. It will be also measured the project size. Function point estimation has five standard functions which are:

**Data Functions:**

* Internal Logical Files [ILF]
* External interface files [EIF]

**Transactional Functions:**

* External Inputs [EI]
* External Outputs [EO]
* External Queries [EQ]

Also DET, RET and FTR have been applied for the analysis of data functions and transactional functions.

|  |  |
| --- | --- |
|  |  |
|  |

The following table shows the complexity matrix for function components.

**Table IV:** Complexity Matrix for FP Function Components

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **ILF/EIF** |  | DET |  | **EI** |  | DET |  | **EO/EQ** |  | DET | |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
| RET | 1-19 | 20-50 | 51+ | FTR | 1-4 | 5-15 | 16+ | FTR | 1-5 | 6-19 |  | 20+ |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
| 1 | Low | Low | Avg | 0-1 | Low | Low | Avg | 0-1 | Low | Low |  | Avg |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
| 2-5 | Low | Avg | High | 2 | Low | Avg | High | 2-3 | Low | Avg |  | High |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
| 6+ | Avg | High | High | 3+ | Avg | High | High | 4+ | Avg | High |  | High |
|  |  |  |  |  |  |  |  |  |  |  |  |  |

The following table shows the function component complexity weight assignment.

**Table V: Function Component Complexity Weight Assignment**

|  |  |  |  |
| --- | --- | --- | --- |
| **Component** | **Low** | **Average** | **High** |
|  |  |  |  |
| External Inputs | 3 | 4 | 6 |
|  |  |  |  |
| External Outputs | 4 | 5 | 7 |
|  |  |  |  |
| External Inquiries | 3 | 4 | 6 |
|  |  |  |  |
| Internal Logical Files | 7 | 10 | 15 |
|  |  |  |  |
| External Interface Files | 5 | 7 | 10 |
|  |  |  |  |

**7.2.1 Function Point Estimation: FTR, DET and RET**

The following table shows the functionality of the system with input and output.

**Table VI:** Function Point Estimation

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Functionality** | **Input** |  | **Output** |  |
|  |  |  |  |  |
| Security system | Click on SignIn by giving | Show | confirmation | message |
|  | input of UserId, Password | for successful login or an | | |
|  |  | error | message for | wrong |
|  |  | input |  |  |
|  |  |  |  |  |
| Role Management | Assign role to user after |  |  |  |
|  | login |  |  |  |
|  |  |  |  |  |

|  |  |
| --- | --- |
|  |  |
|  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Functionality** | **Functionality** | | | **Functionality** | |  |
|  |  |  |  |  | | |
| Sign up system | Click | on | SignUp by | Fields will be added to | | |
|  | filling | up | registration | database |  |  |
|  | form |  |  |  |  |  |
|  |  | | |  | | |
| Manage user by admin | Click user info | | | Showing all user information | | |
|  |  |  |  | and edit or delete or update | | |
|  |  |  |  | user |  |  |
|  |  | | |  | |  |
| Brand Management on | Click on add brand and fill | | | Showing Brand | |  |
| System |  | |  |  |  |  |
|  |  | | |  |  |  |
| Product Management | Click on product to add product | | | Showing | product |  |
| System |  | | |  | |  |
|  |  | | |  | | |
| Approve purchase | Click on pending | | | Approved purchase will be | | |
|  | purchase and approve | | | shown in the posts | |  |
|  |  | | |  | |  |
| Requisition management | Click on requisition | | | Make requisition | |  |
|  |  | |  |  |  |  |
|  |  | | |  | |  |
| Order management | Click on order | | | Show order | |  |
|  |  | | |  | |  |
|  |  | | |  |  |  |
| View report | Click date wise | | | Showing | date | wise |
|  | report |  |  | report |  |  |
|  |  |  |  |  |  |  |

|  |  |
| --- | --- |
|  |  |

**7.2.2 Identify complexity:**

The following table shows the identifying complexity of transaction functions.

**Table VII:** Identify complexity of transaction function

|  |  |  |  |
| --- | --- | --- | --- |
| **Transactions Functions** | **Fields/Files involvement** | **FTRs** | **DETs** |
| Registration (EI) | Fields – Username , name , password , role , contact  File - User | 1 | 5 |
| Login to system(EI) | Fields – username , password  File - user | 1 | 2 |
| Add / Update / Delete  (3EI) | Fields – brandname , status , categoryname , ststus , product name , product image , category , brand , quantity , rate , status , brand id , category id  File – brands , category , product | 3 | 13 |
| Search (EO) | Fields – brand name , category name , product name  File – brand , category , product | 3 | 3 |
| Requisition (EI) | Fields – product id , product name , product quantity , product rate  File - Supplier | 1 | 4 |
| View Product details | Fields –user name , password, brandname , status , categoryname , ststus , product name , product image , category , brand , quantity , rate , status , brand id , category id  File – brand , category , product | 3 | 15 |
| Approve and reject requisition (EI) | Fields – Productid , product name , product rate , quantity , Date , status  File - supplier | 1 | 6 |

|  |  |  |  |
| --- | --- | --- | --- |
| **Transactions Functions** | **Fields/Files involvement** | **FTRs** | **DETs** |
| Add order (EI) | Fields – date , customer name , contact , total , vat , total amount , discount ,status  File – Order , order item | 2 | 8 |
| View Product by customer (EO) | Fields –user name , password, brandname , status , categoryname , ststus , product name , product image , category , brand , quantity , rate , status , brand id , category id  File – brand , category , product | 3 | 15 |
| View Requisition (EO) | Fields – Productid , product name , product rate , quantity , Date , status  File - supplier | 1 | 6 |

The following table shows the identifying complexity of data functions.

**Table VIII:** Identify complexity of data functions

|  |  |  |  |
| --- | --- | --- | --- |
| **Data Function** | **Fields/Files involvement** | **RETs** | **DETs** |
| User (ILF) | Fields – Username, name, password, role, contact | 1 | 5 |
| Manage all System (ILF) | Fields – brandname , status , categoryname , ststus , product name , product image , category , brand , quantity , rate , status , brand id , category id | 3 | 13 |
| Order (ILF) | Fields – date , customer name , contact , total , vat , total amount , discount ,status | 2 | 8 |
| Requisition request (EIF) | Fields – product id , product name , product quantity , product rate | 1 | 4 |
| Generate report (EIF) | Fields – Start date , end date , status | 2 | 3 |

**7.2.3 Unadjusted function Point Contribution for Transaction Function:**

The following table show the unadjusted function point contribution for transaction function.

**Table IX:** Unadjusted function Point Contribution for Transaction Function

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Transactions Functions** | **FTRs** | **DETs** | **Complexity** | **UFP** |
| Registration (EI) | 1 | 5 | Low | 3 |
| Login to system(EI) | 1 | 2 | Low | 3 |
| Add / Update / Delete  (3EI) | 3 | 13 | High | 18 |
| Search (EO) | 3 | 3 | Low | 4 |
| Requisition (EI) | 1 | 4 | Low | 3 |
| View Product details (EO) | 3 | 15 | Average | 5 |
| Approve and reject requisition (EI) | 1 | 6 | Low | 3 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Transactions Functions** | **FTRs** | **FTRs** | **Complexity** | **UFP** |
| Add order (EI) | 2 | 8 | Average | 4 |
| View Product by customer (EO) | 3 | 15 | Average | 5 |
| View Requisition (EO) | 1 | 6 | Low | 4 |
| **Total** |  |  |  | 52 |

The following table shows the unadjusted function point contribution for data functions.

**Table X:** Unadjusted function Point Contribution for Data Function

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Data Function** | **RETs** | **DETs** | **Complexity** | **UFP** |
| User (ILF) | 1 | 5 | Low | 7 |
| Manage all System (ILF) | 3 | 13 | Low | 7 |
| Order (ILF) | 2 | 8 | Low | 7 |
| Requisition request (EIF) | 1 | 4 | Low | 5 |
| Generate report (EIF) | 2 | 3 | Low | 5 |
| **Total** |  |  |  | 31 |

**7.2.4 Performance and environmental impact:**

|  |  |
| --- | --- |
| **GSC** | **DI** |
|  |  |
| **Data Communications** | **5** |
|  |  |
| **Distributed Data Processing** | **0** |
|  |  |
| **Performance** | **5** |
|  |  |
| **Heavily Used Configuration** | **0** |
|  |  |
| **Transaction Rate** | **0** |
|  |  |
| **Online Data Entry** | **4** |
|  |  |
| **End-user Efficiency** | **5** |
|  |  |
| **Online Update** | **4** |
|  |  |
| **Complex Processing** | **1** |
|  |  |
| **Reusability** | **4** |
|  |  |
| **Installation Ease** | **4** |
|  |  |
| **Operational Ease** | **5** |
|  |  |
| **Multiple Site** | **0** |
|  |  |
| **Facilitate Change** | **4** |
|  |  |
| **Total Degree of Influence (TDI)** | **41** |
|  |  |

|  |  |
| --- | --- |
|  |  |
|  |

**7.2.5 Counting Adjusted Function Points:**

UFP for TF = 52

UFP for DF = 31

Total UFP = 83

Value Adjustment Factor (VAF) = (0.65+ (0.01×41))

* + 1.06

Adjusted Function Point (AFP) = UFP × VAF

* 83× 1.06
* 87.98

Efforts for PHP = AFP × Productivity

* + 87.98× 1.06
* 536.678 hours / person

=536.678 /6(a day) =89.44 days/person

=89.44/1 person=89.44+32 holidays = 121.44 days= 3.5 Months

**7.3 Process Based Estimation:**

In process-based estimation, process is decomposed into a relatively small set of tasks and the effort required to accomplish each task is estimated. Process based estimation begins with a delineation of software functions obtained from the project scope. A series of software process activities must be performed for each function.

|  |  |
| --- | --- |
|  |  |
|  |

The following table show process based estimation.

**Table XI:** Process based estimation

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Function** | **CC** | **PLN** | **Risk** | **Engineering** | | **Construction** | | **CE** | **Total** |
| **Name** |  |  | **Analysis** |  |  | **and release** | |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  | **Analysi** | **Design** | **Code** | **Test** |  |  |
|  |  |  |  | **s** |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
| Login System |  |  |  | **2.4** | **4.2** | **4.2** | **1.3** | **N/A** | **12.1** |
|  |  |  |  |  |  |  |  |  |  |
| User |  |  |  | **3.2** | **4.2** | **4.2** | **3.1** | **N/A** | **14.7** |
| Management |  |  |  |  |  |  |  |  |  |
| System |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
| Product |  |  |  | **6.7** | **7.6** | **8.3** | **4.3** | **N/A** | **26.9** |
| Management |  |  |  |  |  |  |  |  |  |
| System |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
| Order |  |  |  | **5.4** | **8.6** | **7.6** | **6.4** | **N/A** | **28** |
| Management |  |  |  |  |  |  |  |  |  |
| System |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
| Requisition System |  |  |  | **3.8** | **4.1** | **4.8** | **2.4** | **N/A** | **15.1** |
|  |  |  |  |  |  |  |  |  |  |
| Purchase |  |  |  | **3.5** | **3.7** | **5.7** | **4.5** | **N/A** | **17.4** |
| System |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
| View report |  |  |  | **2.5** | **4.9** | **4.9** | **3.5** | **N/A** | **15.8** |
|  |  |  |  |  |  |  |  |  |  |
| **Total** | **9** | **12** | **6** | **23.5** | **34.3** | **36.7** | **25.5** |  | **157** |
|  |  |  |  |  |  |  |  |  |  |
| **%Effort** | **6.1** | **8.2** | **4.1%** | **16%** | **23.3%** | **25%** | **17.3** |  | **100** |
|  | **%** | **%** |  |  |  |  | **%** |  | **%** |
|  |  |  |  |  |  |  |  |  |  |

**Effort = 157/2**

**=78.5 days**

Total Days = 78.5 Workdays + 32 Holyday = 110.5 Days = 3.5 Months.

|  |  |
| --- | --- |
|  |  |
|  |

**7.4 Effort Distribution:**

The following chart show the effort distribution of this project.



**Effort Distribution**

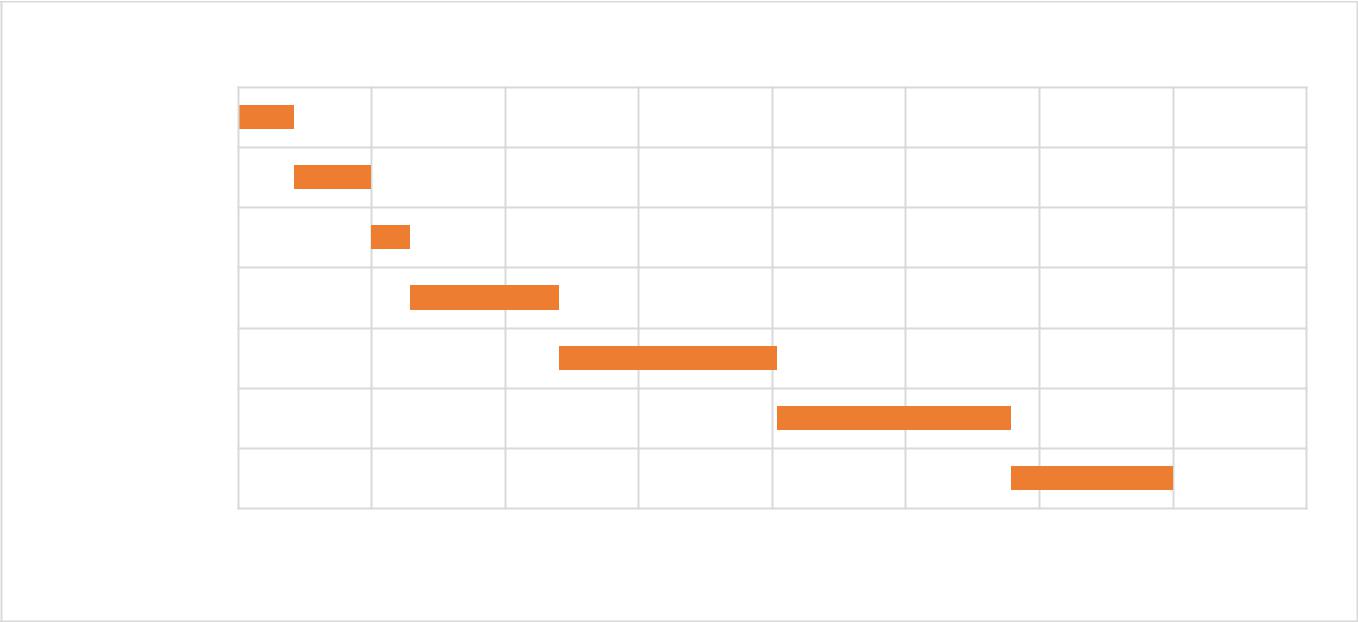
6%

|  |  |  |  |
| --- | --- | --- | --- |
|  | 17% | 8% |  |
|  |  | Customer Communication |
|  |  |  |
|  |  | 4% | Planning |
|  |  |  |
|  |  |  | Risk Analysis |
|  |  | 16% | Analysis |
|  |  |  |
| 25% |  |  | Design |
|  |  |  |
|  |  |  | Coding |
|  |  |  | Testing |
|  |  | 24% |  |

**Figure 7.1:** Effort Distribution Chart.

**7.5 Project Schedule Chart:**

Total system development is a combination of set of tasks. These set of tasks should one sequentially and timely. Project schedule works as the guideline of the system developer. The following is the schedule chart of this project:



0 2 4 6 8 10 12 14 16

CC

Planning

Risk Analysis

Analysis

Design

Coding

Testing

Time  Week

**Figure 7.2:** Project Schedule Chart

|  |  |
| --- | --- |
|  |  |
|  |

**Chapter 8**

**Project Estimation**

|  |  |
| --- | --- |
|  |  |
|  |

**8.1 Personnel cost:**

Number of days in a year = 365

Number of government holidays in a year =22 (estimated)

Number of weekly holidays in a year =104

Total number of working days to develop projects =365-(104+22) =239 days

Total number of working days per months to develop the project =239/12 =19.92 days Organization working hours per day = 6 hours

Organization working hours per month=19.92\*6= 119.52 hours The following table show the personnel cost of this project.

**Table XII:** Personnel cost

|  |  |  |
| --- | --- | --- |
| **Position** | **Salary/Month** | **Salary/Hour** |
|  |  |  |
| System Analyst | 15,000 | 125.5 |
|  |  |  |
| Programmer | 10,000 | 83.67 |
|  |  |  |

**8.1.1 Salary for the Technical Staffs Engaged:**

The following table shows the salary of technical staffs of this organization.

**Table XIII:** Salary for the Technical Staffs Engaged

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **Person** |  |  | **Total** |  |
| **Job Title** | **Duration** | **Weeks** | **Working** | **Total Salary** |
| **Required** |
|  |  |  |  | **Hour** |  |
|  |  |  |  |  |  |
| System | 1 |  |  | 14\*5\*6 = 420 | TK 420\*125.5 = 52,710.00 |
| Analyst | 3.5 | 14 | H |
|  |  |
|  |  |  |  |
|  |  | Months | Weeks |  |  |
| Programmer | 1 | 14\*5\*6 = 420 | TK 420\*83.67= 35,141.40 |
|  |  |
|  |  | H |
|  |  |  |  |  |
|  |  |  |  |  |  |

|  |  |
| --- | --- |
|  |  |
|  |

**8.1.2 Depreciation Calculation:**

The first step is to sum the digits or numbers starting with the life and going back to one. For example, an asset with a life of 5 would have a sum of digits as follows: 5+ 4+ 3 +2 + 1 = 15. To find the percentage for each year divide the year's digit by the sum. In the example above the percentage would be calculated as follows:

**Table XIV:** Depreciation Calculation

|  |  |
| --- | --- |
| **Year 1** | **5/ 15 = 33.34%** |
|  |  |
| **Year 2** | **4/ 15= 26.67%** |
|  |  |
| **Year 3** | **3/ 15 = 20 %** |
|  |  |
| **Year 4** | **2/ 15= 13.33 %** |
|  |  |
| **Year 5** | **1/ 15 = 6.67%** |
|  |  |

**8.2 Hardware cost:**

The following table show the hardware cost for this project. It is given bellow.

**Table XV:** Hardware Cost Estimation

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Description | Quantity | Price | Lifetime | Using Year | Depreciation Cost (TK) |
|  |  |  |  |  |  |
| Computer | 2 | 30,000 | 5 Years | 3rd | 2\*(30,000 \*20%) = 12,000 |
|  |  |  |  |  |  |
| Printer | 1 | 5,000 | 5 Years | 2nd | (5000\* 26.67%) = 1,333.5 |
|  |  |  |  |  |  |
|  | **Total** | **Hardware** | **Cost:** |  | **13,333.5** |
|  |  |  |  |  |  |

**8.3 Software cost:**

The following table show the software cost for this project.

**Table XVI:** Software Cost Estimation

|  |  |  |  |
| --- | --- | --- | --- |
| Description | Price | Lifetime | Depreciation Cost (TK) |
|  |  |  |  |
| Windows 10 | 10,000 | 5 Years | (10,000 \*13.33%) = 1,333 |
|  |  |  |  |
| Microsoft Office 2007 | 8,000 | 5 Years | (8,000\* 13.33%) = 1,066.4 |
|  |  |  |  |
| **Total Software Cost:** | |  | **2399.4** |
|  |  |  |  |

|  |  |
| --- | --- |
|  |  |
|  |

**8.4 Accounts Table:**

The accounts table of this project is given below. This table is shown accounts status.

**Table XVII**: Accounts Table

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Particulars** | **Taka** |  |
|  |  |  |  |
|  | **Salary** |  |  |
|  |  |  |  |
|  | System Analyst | 52,710.00 |  |
|  |  |  |  |
|  | Programmer | 35,141.40 |  |

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | **87,851.4** |  |
|  |  |  |  |
|  | **Hardware Cost** |  |  |
|  |  |  |  |
|  | Computer | 12,000.00 |  |
|  |  |  |  |
|  | Printer | 1,333.50 |  |

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | **13,333.50** |  |
|  |  |  |  |
|  | **Software Cost** |  |  |
|  |  |  |  |
|  | Windows 10 | 1,333.00 |  |
|  |  |  |  |
|  | Microsoft Office 2007 | 1,066.40 |  |
|  |  |  |  |
|  | Sublime Text | 0.00 |  |
|  |  |  |  |
|  | My Sql Server | 0.00 |  |

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | **2399.40** |  |
|  |  |  |  |
|  | **Other Costs** |  |  |
|  |  |  |  |
|  | Furniture | 2,000.00 |  |
|  |  |  |  |
|  | House Rent | 7,000.00 |  |
|  |  |  |  |
|  | Electricity bill | 1,000.00 |  |
|  |  |  |  |
|  | Utilities | 1,000.00 |  |
|  |  |  |  |
|  |  | **11,000.00** |  |
|  |  |  |  |
|  | Total: | **1,14,584.3** |  |
|  |  |  |  |

|  |  |
| --- | --- |
|  |  |
|  |

**Chapter 9**

**Quality Management**

|  |  |
| --- | --- |
|  |  |
|  |

Quality management provides an independent check on the software development process. The quality management process checks the project deliverables to ensure that they are consistent with organizational standards and goals.

**9.1 System Quality Management:**

Quality Management concerned with ensuring that the required level of quality is achieved in a software product. It involves the application of specific quality processes and checking that these planned processes have been followed. Also establish a better quality plan for a project. The quality plan should set out the quality goals for the project and define what processes and standards are to be used. Agreement on quality requirements as well as clear communication to the software engineer on what constitutes quality, require that the many aspects of quality formally defined and discussed. A software engineer should understand the underlying meanings of quality concepts and characteristics and their value to the software under development or to maintenance. The important concept is that the software requirements define the required quality characteristics of the software and influence the measurement methods and acceptance criteria for assessing these characteristics. Some quality criteria are objective, and can be measured accordingly. Some quality criteria are subjective, and are therefore captured with more arbitrary measurement.

**9.1.1 Quality Assurance Matrix:**

|  |  |  |  |
| --- | --- | --- | --- |
|  | Quality Assurance Scenario No: 1 |  |  |
|  |  |  |  |
| Requirement | Admin can log in the system |  |  |
|  |  |  |  |
| Provided outputs | enters the system using UserId, Password. |  |  |
|  |  |  |  |
| Decision | This system working correctly. It should be working in future. |  |  |
|  |  |  |  |
|  |  |  |  |
|  | Quality Assurance Scenario No: 2 |  |  |
|  |  |  |  |
| Requirement | Admin can view and delete user |  |  |
|  |  |  |  |
| Provided outputs | Admin can manage user. |  |  |
|  |  |  |  |
| Decision | This system working correctly. It should be working in future. |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |  |

|  |  |
| --- | --- |
|  | Quality Assurance Scenario No: 3 |
|  |  |
| Requirement | Admin can add, update, view and delete his Post |
|  |  |
| Provided outputs | Admin manage the post system |
|  |  |
| Decision | This system working correctly. It should be working in future. |
|  |  |

|  |  |
| --- | --- |
|  | Quality Assurance Scenario No: 4 |
|  |  |
| Requirement | User can order product |
|  |  |
| Provided outputs | Order products |
|  |  |
| Decision | This system working correctly. It should be working in future.. |
|  |  |

|  |  |
| --- | --- |
|  | Quality Assurance Scenario No: 5 |
|  |  |
| Requirement | User can approve or reject requisition |
|  |  |
| Provided outputs | Easily create |
|  |  |
| Decision | This system working correctly. It should be working in future. |
|  |  |
|  |  |
|  | Quality Assurance Scenario No: 6 |
|  |  |
| Requirement | Automatically generate report |
|  |  |
|  | Admin can generate it |
| Provided outputs |  |
|  |  |
| Decision | This system working correctly. It should be working in future. |
|  |  |

|  |  |
| --- | --- |
|  |  |
|  |

**Chapter 10**

**Coding**

|  |  |
| --- | --- |
|  |  |
|  |

Coding is analytical part of a project. It is one of the process of the system development method. In this project is used two type coding.

**10.1 Coding:**

Coding is needed for developing a software. In this system coding part is come after designing. To develop this software is used waterfall process model. In this process model coding is one of the important approach. Here is used two type of coding.

* Backend coding
* Frontend coding

**10.1.1 Backend coding:**

Sublime text is used to develop this software as a backend code. Backend code is used to handle back side of a software. It will also handle functionality of the project. The coding part will show the project demonstration part.

**10.1.2 Frontend coding:**

In this system to develop for frontend code is used HTML, CSS, Bootstrap, JavaScript, Ajax etc. These codes are used to make user friendly interface. This coding part also show on the project demonstration part.

|  |  |
| --- | --- |
|  |  |
|  |

**Chapter 11**

**Testing**

|  |  |
| --- | --- |
|  |  |
|  |

According to the common process framework, the software testing is the final activity that has to initiate after developing. Software testing is a critical element of software quality assurance and represents the ultimate review of specification, design and code generation.

**11.1 System Testing Strategy:**

A strategy for software testing integrates software test case design methods into a well-planned series of steps that result in the successful construction of a software. The strategy provides a road map that describes the steps to be conducted as part of testing.

Testing strategy that will be followed in this software project:

* Unit testing
* Integration testing
* Validation testing

The first step in software testing is unit testing. Unit testing concentrates on each unit of the software as implemented in the source code. Unit testing focuses on each component individually. The unit test is white-box oriented. Thus, unit testing of this software will be done after completion of every module or component.

The next step is integration testing. Integration testing is a systematic technique for constructing the program structure while at the same time conducting tests to uncover errors associated with interfacing. The objective of integration testing is to take unit tested components and build a program structure that has been dictated by design.

The integration testing strategy that has been chosen for this project is top down testing. Black-box testing method is the most prevalent for integration testing. Top down integration strategy will be used to perform integration testing. Top down integration will be done by breadth-first manner. Breadth-first integration incorporates all components directly subordinate at each level, moving across the structure horizontally.

After the software has been integrated, a set of high order tests are conducted. Hence, the validation criteria that have been mentioned in requirements engineering should be tested. Validation testing provides final assurance that software meets all functional, behavioral and performance requirements. The black-box testing method is exclusively used in validation.

|  |  |
| --- | --- |
|  |  |
|  |

**11.2 System Testing Methodology:**

**Black-box Testing:**

In this project, Black-box testing method has been used to examine the functionality of this application without peering into its internal structures or workings. This method has been applied virtually to every level of software testing: unit, integration, system and acceptance.

**White-box Testing:**

White-box testing has been used to in this project to design test cases. Inputs were chosen to exercise paths through the code and determine the appropriate outputs.

**11.3 Testing Scenario**

|  |  |
| --- | --- |
|  | Testing scenario No:1 |
|  |  |
| Scenario | Login testing scenario of the system |
|  |  |
| Input(s) | Correct UserId, Password |
|  |  |
|  | When enter a valid UserId, password then get access to level |
| Desired Output(s) | defined. |
|  |  |
| Actual Output(s) | For login system worked correctly |
|  |  |
|  | Getting result from Desired Output’s and Actual Output’s decided |
| Verdict | this system is successful for login. |
|  |  |

|  |  |
| --- | --- |
|  | Testing scenario No:2 |
|  |  |
| Scenario | Admin can delete user |
|  |  |
| Input(s) | User’s basic info for registration with access level status byUserId |
|  |  |
| Desired Output(s) | When enter all basic info correctly, system shows a success message |
|  |  |
| Actual Output(s) | For delete user, system worked correctly |
|  |  |
|  | Getting result from Desired Output and Actual Output decided this |
| Verdict | system is successful for Adding User. |
|  |  |

|  |  |
| --- | --- |
|  |  |
|  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  | Testing scenario No:3 |  |  |
|  |  |  |  |  |
| Scenario |  | Admin can add new Product. |  |  |
|  |  |  |  |  |
| Input(s) |  | Product info is provide correctly |  |  |
|  |  |  |  |  |
| Desired Output(s) |  | When adding new product system shows success message |  |  |
|  |  |  |  |  |
| Actual Output(s) |  | Adding product |  |  |
|  |  |  |  |  |
| Verdict |  | The process worked correctly and successfully. |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  | Testing scenario No:4 |  |  |
|  |  |  |  |  |
| Scenario |  | Automatically report generation |  |  |
|  |  |  |  |  |
| Input(s) |  | User will give date |  |  |
|  |  |  |  |  |
| Desired Output(s) |  | It will generate report |  |  |
|  |  |  |  |  |
| Actual Output(s) |  | Store report |  |  |
|  |  |  |  |  |
| Verdict |  | The process worked correctly and successfully. |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  | Testing scenario No:5 |  |  |
|  |  |  |  |  |
| Scenario |  | User can order |  |  |
|  |  |  |  |  |
| Input(s) |  | Provide order as a text form |  |  |
|  |  |  |  |  |
| Desired Output(s) |  | If successfully order it show success message |  |  |
|  |  |  |  |  |
| Actual Output(s) |  | Store order info |  |  |
|  |  |  |  |  |
| Verdict |  | The process worked correctly and successfully. |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  | Testing scenario No:6 |  |  |
|  |  |  |  |  |
| Scenario |  | User get new requisition |  |  |
|  |  |  |  |  |
| Input(s) |  | Provide requisition info |  |  |
|  |  |  |  |  |
| Desired Output(s) |  | Successfully get new message |  |  |
|  |  |  |  |  |
| Actual Output(s) |  | Get new view |  |  |
|  |  |  |  |  |
| Verdict |  | The process worked correctly and successfully. |  |  |
|  |  |  |  |  |
|  |  | |  |  |
|  |  | |
|  |  |  |
|  |  |  |  |  |

**Chapter 12**

**Conclusion**

|  |  |
| --- | --- |
|  |  |
|  |

**12.1 The Experience:**

The biggest experience of working at BLCF Limited is indeed being a part of designing and implementing software. The widest area of experience was gathered round the designing issue. Definitely, there are a lot of new issue to learn from this system.

Working in a project is a big opportunity for anyone. One can learn a lot about the technical and political Environment of IT &Software Development Organization. The followings will indicate anyone’s learning achievements from this project.

* The designing and development strategy of a web application.
* Database Management.
* The analyzing strategy of software.
* Hands-on experience about system security management.
* Version controlling using git

**12.2 Limitation:**

As it has been mentioned early in this report that practicum is the bridge between theoretical and practical life, practicum program at IUBAT has given the students this great opportunity to see how theories are put into action. From this point of view, a 12-16 weeks practicum program is not good enough for a fresh graduate to work in an organizational and at the same time develop software. Following by, there were lots of terms, conditions and systems that were not understandable by the students at the beginning of their organizational attachment. So far, it may be learned seeing senior classmates and friends, they experienced the same during their time as well. After applying the software engineering procedures, it is very difficult to develop complete software within short time and it might be required to include more modules for the project. But at the later stage, one should be familiar with the overall scenario.

|  |  |
| --- | --- |
|  |  |
|  |

**12.3 Future Work:**

This software is web application; by this software, the organization will be able to manage content management system for multiple purposes. This system can be introduced as a very initial version of the big software. By this limited time of internship, one can develop the core features of this system but in future, it can be possible to add more features e.g.:

* Create an API

**12.4 Conclusion:**

This report helps a person to understand this whole application. How the software works and what is the use of this application. Where reader can get help from this application. We tried to make this application easy and faster for the user. Now we can say that the system will fulfill the user requirements and it will cover all their basic requirements. After the testing and user feedback, We are happy to work with this system.

|  |  |
| --- | --- |
|  |  |
|  |

**Bibliography:**

W3schools.com

Tutorialspoint.com

|  |  |
| --- | --- |
|  |  |
|  |