

FIXED ASSETS

**A Project Report Submitted
In Partial Fulfillment of the Requirements
for the Degree of**

MASTER OF COMPUTER APPLICATIONS

**By
Aman Vashishth
(1900290149013)**

**Under the Supervision of
Dr. Sangeeta Arora**

KIET Group of Institutions, Ghaziabad



**To the
FACULTY OF MCA**

**DR. APJ ABDUL KALAM TECHNICAL UNIVERSITY
(Formerly Uttar Pradesh Technical
University) LUCKNOW**

July 2021

DECLARATION

I hereby declare that the work presented in this report entitled “Online Tourist Guide”, was carried out by me. I have not submitted the matter embodied in this report for the award of any other degree or diploma of any other University or Institute.

I have given due credit to the original authors/sources for all the words, ideas, diagrams, graphics, computer programs, experiments, results, that are not my original contribution. I have used quotation marks to identify verbatim sentences and given credit to the original authors/sources.

I affirm that no portion of my work is plagiarized, and the experiments and results reported in the report are not manipulated. In the event of a complaint of plagiarism and the manipulation of the experiments and results, I shall be fully responsible and answerable.

Name : Aman Vashishth

Roll. No. : 1900290149013

Branch : Master of Computer Application

COMPANY CERTIFICATE

JIL INFORMATION TECHNOLOGY LIMITED

An ISO 9001 : 2015 & ISO 27001 : 2013 Company
CMM-DEV V1.3

JILIT/PERS/2021
08.03.2021

Shri Aman Vashishth
50/362, Jawahar Ganj
Railway road, Hapur

Sub: Induction as "Trainee" at Sahibabad Office

Dear Madam,

With reference to your application & subsequent interview at our office. We are pleased to inform you that you are provisionally selected for the post of "Trainee" for Sahibabad Office on the terms & condition discussed and agreed by you.

You are requested to report for duty to Shri Vinay Pathak on or before 09.03.2021 at the following address failing which the offer shall stand withdrawn:

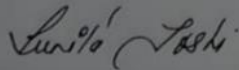
64/4, Site IV,
Sahibabad Industrial Area
Ghaziabad-201010 (U.P)
Phone : 0120-4964576 Extn.22576
Contact Person : Vinay Pathak

Please bring the following documents with you at the time of joining:

1. Three copies of your recent passport size photograph.
2. Original certificates and mark sheets of all examinations passed by you along with one set of photocopy
3. Original experience certificates.
4. Proof of your present emoluments and perquisites.
5. Relieving letter from your present employer.

Thanking You

Yours Faithfully
For JIL INFORMATION TECHNOLOGY LTD.



(AUTHORISED SIGNATORY)

JILIT

Regd. Office : 54, Basant Lok, Vasant Vihar, New Delhi - 110057 (India)
Head Office : 64/4, Site IV, Sahibabad Industrial Area, Distt. Ghaziabad-201010 (India)
Phone : +91(120) 4963100, 4964100 E-mail : sales.info@jilit.co.in
Website : www.jilit.co.in CIN : U72200 DL 2000PLC120604

CERTIFICATE

Certified that **Aman Vashishth (1900290149013)** has carried out the project work presented in this report entitled “**Fixed Assets**” for the award of **Master of Computer Application** from Dr. A.P.J. Abdul Kalam Technical University, Lucknow under my supervision. The report embodies result of original work, and studies are carried out by the student himself and the contents of the report do not form the basis for the award of any other degree to the candidate or to anybody else from this or any other University.

Dr. Sangeeta Arora

Professor

Department of Computer Applications

KIET Group of Institute

Date: 30/06/2021

ABSTRACT

The recent past showed a greater interest in managing company assets. Now there are a lot softwares existing in market existing from different websites to almost all the places over the world. A customer finds it very difficult to search for the best source for maintaining the data. As the customer has to know that it is possible to have a system like that, who can maintain the data and also store the data safely.

Fixed Assets is a Project which is useful to the organization to maintain the full record of their assets(property) of the organization. Fixed Assets contain many features like AssetTypeMaster, AssetTypeAttribute, ScrapDeclaration, ScrapAuthorization etc.

ACKNOWLEDGEMENT

It is our privilege to express our sincerest regards to our project coordinator, Prof. Sangeeta Arora for their valuable inputs, guidance, encouragement, whole-hearted cooperation and constructive criticism throughout the duration of our project. Their useful suggestions for this whole work and co- operative behavior are sincerely acknowledged.

We deeply express our sincere thanks to our Head of Department Dr. Ajay Shrivastava for encouraging and allowing us to present the project on the topic “Fixed Assets” at our department premises for the partial fulfillment of the requirements.

We take this opportunity to thank all our lecturers who have directly or indirectly helped our project. We pay our respects and love to our parents and all other family members and friends for their love and encouragement throughout our career. Last but not the least we express our thanks to our friends for their cooperation and support.

LIST OF FIGURES

1	Fig. Process Design of System Analysis	5
2	Fig. Gantt Chart	8
3	FIG. ERD DIAGRAM	20
4	FIG. PROCESS LOGIC	32
5	FIG.6.4 EXECUTION FLOW CHART	65
6	FIG.6.5 INTEGRATION TESTING	69
7	FIG. 6.6 BOTTOM UP INTEGRATION	74

LIST OF TABLES

TABLE 4.3.1 ASSETYPEMASTER	23
TABLE 4.3.2 ASSETYPEATTRIBUTE	23
TABLE 4.3.3 SCRAPDECLARATION	24
TABLE 4.3.4 SCRAPAUTHORIZATION	24

TABLE OF CONTENTS

Declaration	ii
Certificate	iii
Abstract	iv
Acknowledgement	V
List of Figures	vi
List of Tables	vii
CHAPTER 1: INTRODUCTION	1-3
1.1 Project Description	1
1.2 Project Purpose	1-2
1.3 Tools/Environment used	3
CHAPTER 2 : LITERATURE REVIEW	4-10
2.1 System analysis	4-5
2.2 Feasibility study	6
2.3 Fact finding technique	7-9
2.4 Scheduling	9-10
CHAPTER 3 : REQUIREMENT SPECIFICATIONS	11-17
3.1 Hardware Requirements	11
3.2 Software Requirements	11-16
3.3 SRS of the project	17-19
Introduction	17
Information Description	17
Functional Description	18-19

CHAPTER 4 : DESIGN	20-24
Entity Relationship Diagram	20-22
4.1 Data Dictionary	23
4.1.1 Table used	23-24
 CHAPTER 5 : USER INTERFACES / CODING	 25-60
5.1 User Interface	25-28
5.2 Coding	29-60
 CHAPTER 6 : TESTING AND VALIDATION CRITERIA	 61-76
6.1 Error Handling	61-62
6.2 Parameter Passing	63
6.3 Validation Checks	64
6.4 Testing	65-76
 CHAPTER 7 : LIMITATION AND FUTURE SCOPE	 102-103
7.1 Limitation	102
7.2 Future Scope	103

CHAPTER 1

INTRODUCTION

1.1 PROJECT DESCRIPTION

This Project 'Fixed Assets' helps the user to maintain the record of assets and property of the company. It will be used to save all the information about the assets of the Organization. The reason of making of this project is to make a well maintained record of all the fixed assets.

We can describe this project in words that it is very useful to the user to record the data into the database. Fixed assets provide a platform to the user where user enter the data to the Interface and the data is saved to the Database.

Fixed Assets is a web based portal which help the organization or any other user to save and get accurate required data in no time. This Project is helpful for the organization who wants to make a systematic data record of their fixed assets, and save new assets data and description like – AC, Computer Systems, Furnitures, etc.

1.2 PROJECT PURPOSE

This system has the following features:

- a) The Purpose of making this project is to making the record of the assets to the Database.
- b) This Project containing the modules or many different requirements according to the user.

- c) In this Project user can check the assets and add them into the list of 'Scrap' in the database.
- d) After using the assets a lot of time and when it is getting old now the authorized user can add the assets into the list of Scrap.
- e) The Manager of Organization have the authority to check and approve the Scrap Assets added by the employee.

1.3 Tools/Environment Used:

HARDWARE USED

PROCESSOR- CORE 2

RAM - 8 GB

Monitor – HCL LCD

Hard disk – 500 GB

Keyboard - Lenovo

SOFTWARE USED

Visual studio -2012

SQL server -2012

Operating System – windows 10

Ms Office -2013

CHAPTER 2

LITERATURE REVIEW

2.1 System Analysis

System Analysis by definition is a process of systematic investigation for the purpose of gathering data, interpreting the facts, diagnosing the problem and using this information to either build a completely new system or to recommend the improvements to the existing system.

A satisfactory system analysis involves the process of examining a business situation with the intent of improving it through better methods and procedures. In its core sense, the analysis phase defines the requirements of the system and the problems which user is trying to solve irrespective of how the requirements would be accomplished.

System development can generally be thought of having two major components: systems analysis and systems design. In System Analysis more emphasis is given to understanding the details of an existing system or a proposed one and then deciding whether the proposed system is desirable or not and whether the existing system needs improvements. Thus, system analysis is the process of investigating a system, identifying problems, and using the information to recommend improvements to the system.

Process Design of System Analysis

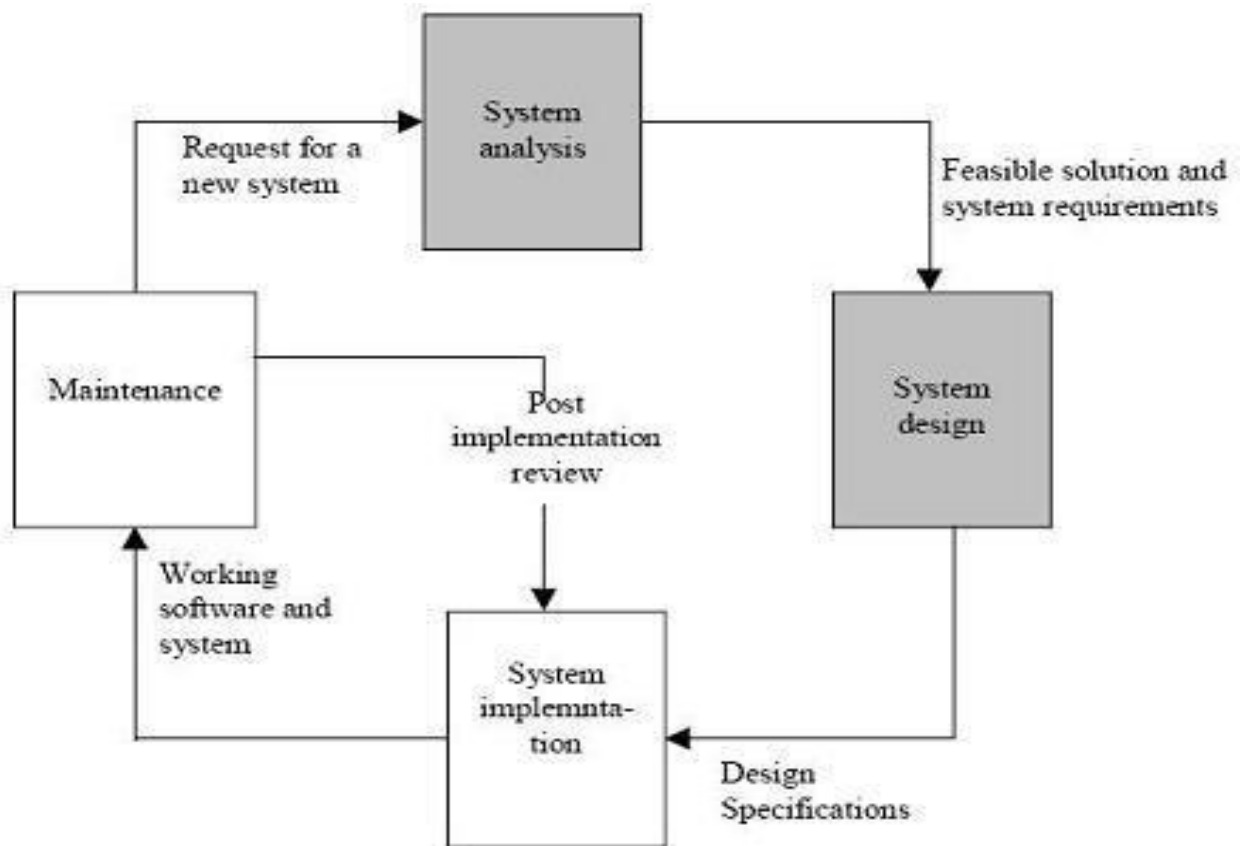


Fig:- Process Design of System Analysis

2.2 Feasibility Study

Feasibility is the determination of whether or not a project is worth doing. The process followed in making this determination is called a feasibility study.

Types of feasibility

- I. **Technical Feasibility:** -It is concerned with the availability of hardware and software required for the development of the system. The technical needs of the system may vary considerable, but might include:
 - The facility to produce outputs in a given time.
 - Response time under certain condition.
 - Ability to process a certain volume of transaction at a particular speed.
 - Facility to communication data to distinct location.
- II. **Operational Feasibility:** - Operational feasibility is all about problems that may arise during operations. There are two aspects related with this issue :-
 - What is the probability that the solution developed may not be put to use or may not work?
 - What is the inclination of the management and end users towards the solution?
- III. **Economic Feasibility:** - It is the measure of cost effectiveness of the project. The economic feasibility is nothing but judging whether the possible benefit of solving the problem is worth right or not.
- IV. **Social Feasibility:** - Social feasibility is determined a proposed project will be acceptable to the people or not.
- V. **Management Feasibility:** - This type of feasibility determines a proposed project will be acceptable to management. If Management does not support or gives a negligible support to it. The analyst will tend to view the project as a non-feasible one.
- VI. **Legal Feasibility:** - Legal feasibility studies issues arising out of the need to the development of the system. The possible consideration might include copyright law, labor law, antitrust legislation, foreign trade, regulation etc.

2.3 Fact Finding Techniques

To Study any system the analyst needs to do collect facts and all relevant information the facts when expressed in quantitative form are termed as data. The success of any project is depended upon the accuracy of available data. Accurate information can be collected with help of certain methods/ techniques. These specific methods for finding information of the system are termed as fact finding techniques. Interview, Questionnaire, Record View and Observations are the different fact finding techniques used by the analyst. The analyst may use more than one technique for investigation.

Interview

This method is used to collect the information from groups or individuals. Analyst selects the people who are related with the system for the interview. In this method the analyst sits face to face with the people and records their responses. The interviewer must plan in advance the type of questions he/ she is going to ask and should be ready to answer any type of question. He should also choose a suitable place and time which will be comfortable for the respondent.

The information collected is quite accurate and reliable as the interviewer can clear and cross check the doubts there itself. This method also helps gap the areas of misunderstandings and help to discuss about the future problems. Structured and unstructured are the two sub categories of Interview. Structured interview is more formal interview where fixed questions are asked and specific information is collected whereas unstructured interview is more or less like a casual conversation where in- depth areas topics are covered and other information apart from the topic may also be obtained.

Questionnaire

It is the technique used to extract information from number of people. This method can be adopted and used only by a skillful analyst. The Questionnaire consists of series of questions framed together in logical manner. The questions are simple, clear and to the point. This method is very useful for attaining information from people who are concerned with the usage of the system and who are living in different countries. The questionnaire can be mailed or sent to people by post. This is the cheapest source of fact finding.

Record view

The information related to the system is published in the sources like newspapers, magazines, journals, documents etc. This record review helps the analyst to get valuable information about the system and the organization.

On-Site Observation

Unlike the other fact finding techniques, in this method the analyst himself visits the organization and observes and understands the flow of documents, working of the existing system, the users of the system etc. For this method to be adopted it takes an analyst to perform this job as he knows which points should be noticed and highlighted. An analyst may observe the unwanted things as well and simply cause delay in the development of the new system.

2.4 Scheduling

2.4.1 Gantt chart:

Gantt chart is a project control technique that can be used for several purposes including scheduling and planning. Gantt chart is also known bar chart with each box representing an activity.

We estimated the no. of weeks required for each as follows :

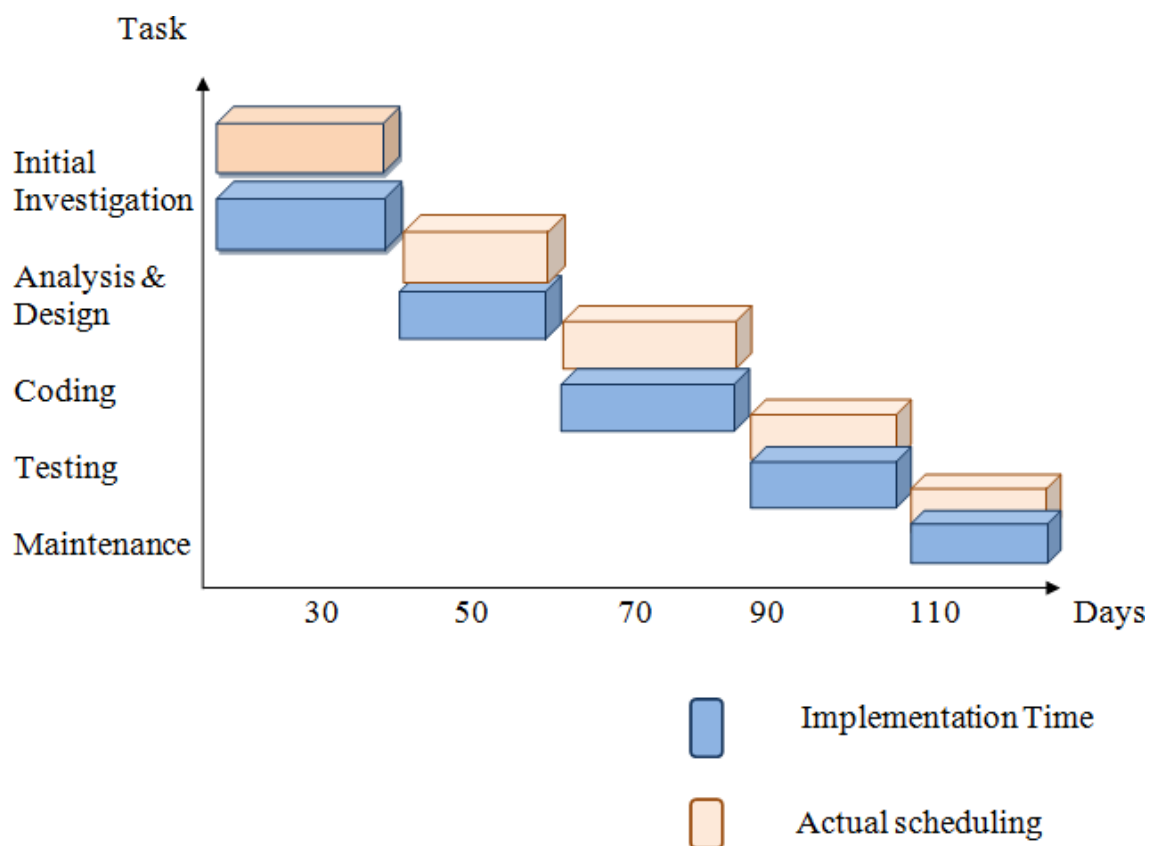


Fig 2.4.1:- Gantt Chart

2.4.2 Program Evaluation and Review Technique (PERT)

Like the Gantt chart, PERT makes use of tasks. Like milestone charts, it shows achievements. These achievements however are not task achievements. They are terminal achievement, called events. Arrows are used to represent tasks and circles represent the beginning or completion of a task. The PERT chart uses these paths and events to show the interrelationships of project activities.

The events in my project can be categorised as:

1. Meeting to the Employees of company to understand the project.
2. Table Designing
3. Form Designing
4. Writing Codes
5. Designing Reports
6. Testing the project
7. Implementation of project

Each task is limited by an identifiable event. An event has no duration; it simply tells you that the activity has ended or begun. Each task must have a beginning and an ending event. A task can start only after the tasks depends on have been completed.

PERT does not allow “looping back” because a routing that goes back to a task does not end.

CHAPTER 3

REQUIREMENT SPECIFICATIONS

3.1 HARDWARE REQUIREMENTS

- RAM: 8GB
- Operating system: Windows (32 bit or 64 bit)

3.2 SOFTWARE REQUIREMENTS

ANGULAR JS FRAMEWORK:-

AngularJS is a very powerful JavaScript Framework. It is used in Single Page Application (SPA) projects. It extends HTML DOM with additional attributes and makes it more responsive to user actions. AngularJS is open source, completely free, and used by thousands of developers around the world. It is licensed under the Apache license version 2.0.

FEATURES

AngularJS is the most widely-used and popular JavaScript Framework for the process of web application development. It's earned love from the community because it gives developers the flexibility and resources that they need to develop user-friendly, flexible and platform-independent web applications. The whole development process can be completed quickly and seamlessly, with as few hurdles as possible.

I. The MVC Framework

AngularJS provides developers with "Model View Control" architecture which is perfect for dynamic modelling. As you may already know, any application is built from a process of combining modules together. These modules work using different logics that are initialized according to individual needs. Developers need to build components separately then combine them together with code.

MVC with AngularJS makes it easier for developers to build client-side web applications. All the necessary elements are developed separately and combined automatically, which saves developers a great deal of additional time and effort.

II. HTML User Interface

Another great feature of AngularJS is the fact that it uses the HTML language to build user interfaces. The HTML language is a common and declarative language with very short tags that are simple to understand. This leads to a more organised and simplistic UI. JavaScript interfaces are often more complicated to develop and organise. If you're looking for a solution that's quick, simple, and easy to use at a moment's notice, then this could be it.

III. Access to the POJO Model

AngularJS also uses the "plain old JavaScript objects" model, which is very self-sufficient and highly functional. Earlier data models used to have to keep monitoring the data flow in an application. However, a POJO data model simply offers very well-planned objects and logics. Developers only need to create loops with the right properties and play around to get the best results. This means that developers can get the clear code they need for highly interactive and user-friendly apps.

IV. Behaviour with Directives

Angular provides extra functionality with the HTML language using directives. The additional elements in the system can be accessed with no need for the DOM to simulate additional elements. The controller doesn't need to manipulate the DOM directly, as this should be done through directives. Directives make up a separate part of the element set which can be used anywhere other than in a web application. Directives give developers the element-rich HTML they need to strengthen their online presence.

V. Filtering

As you might naturally guess, filters in the AngularJS framework simply filter out the data before it reaches the view. They perform paginations, as well as filtering data arrays with respect to available parameters. The functions can be modified according to the ideal parameters in your system, but these are the only data transformation tasks done. The system works by putting information into the right format before it's delivered to the end-user. For instance, it might put a decimal point in a number before reversing the order of numbers in a desired range.

VI. Unit Testing Facilities

Reliability and performance are an important part of making sure that a site works as it should. Before AngularJS, testing would have to be performed by creating an individual test page and using that page to test the behaviour of each component. This was a frustrating and time-consuming process. Thanks to AngularJS, the testing process can become a lot simpler. The application simply uses Dependency injection to bind the application together. This helps everything to function as it should while managing the control with a lot of simplicity.

All the controllers available within the AngularJS unit testing facilities are dependent on the dependency injection, which means that you can adjust certain aspects to find out the preferred configuration for data or an app.

SPRINGBOOT FRAMEWORK:-

The Spring Framework provides a comprehensive programming and configuration model for modern Java-based enterprise applications - on any kind of deployment platform.

A key element of Spring is infrastructural support at the application level: Spring focuses on the "plumbing" of enterprise applications so that teams can focus on application-level business logic, without unnecessary ties to specific deployment environments.

FEATURES

I. Web Development

It is well suited Spring module for web application development. We can easily create a self-contained HTTP server using embedded Tomcat, Jetty or Undertow. We can use the spring-boot- starter-web module to start and running application quickly.

II. SpringApplication

It is a class which provides the convenient way to bootstrap a spring application which can be started from main method. You can call start your application just by calling a static run() method.

III. Application Events and Listeners

Spring Boot uses events to handle variety of tasks. It allows us to create factories file that are used to add listeners. we can refer it by using ApplicationListener key.

IV. Admin Support

Spring Boot provides the facility to enable admin related features for the application. It is used to access and manage application remotely. We can enable it by simply using spring.application.admin.enabled property.

SQL:-

SQL (Structured Query Language) is a standardized programming language that's used to manage relational databases and perform various operations on the data in them. Initially created in the 1970s, SQL is regularly used not only by database administrators, but also by developers writing data integration scripts and data analysts looking to set up and run analytical queries.

SQL became the de facto standard programming language for relational databases after they emerged in the late 1970s and early 1980s. Also known as SQL databases, relational systems comprise a set of tables containing data in rows and columns. Each column in a table corresponds to a category of data -- for example, customer name or address -- while each row contains a data value for the intersecting column.

SQL contains of some important features and they are:

1. Data Definition language (DDL)

It contains of commands which defines the data. The commands are:

create: It is used to create a table.

Syntax:

create table

tablename(attribute1 datatype.....attribute2 datatype);

2. Data Manipulation Language (DML)

Data Manipulation Language contains commands used to manipulate the data.

The commands are:

insert: This command is generally used after the create command to insert a set of values into the table.

Syntax:

insert into tablename values(attribute1 datatype);

insert into tablename values (attributen datatype);

3. Client server execution and remote database access

Client server technology maintains a many to one relationship of clients(many) and server(one). We have commands in SQL that control how a client application can access the database over a network.

4. Security and authentication

SQL provides a mechanism to control the database meaning it makes sure that only the particular details of the database is to be shown the user and the original database is secured by DBMS.

5. Embedded SQL

SQL provides the feature of embedding host languages such as C, COBOL, Java for query from their language at runtime.

6. Transaction Control Language

Transactions are an important element of DBMS and to control the transactions, TCL is used which has commands like commit, rollback and savepoint.

commit: It saves the database at any point whenever database is consistent.

Syntax:

commit;

3.1 SRS of the Project

3.3.1 Introduction

Fixed Assets is a web based portal which help the organization or any other user to save and get accurate required data in no time. This Project is helpful for the organization who wants to make a systematic data record of their fixed assets, and save new assets data and description like – AC, Computer Systems, Furnitures, etc. The user can save the data by providing it a unique Id and description and also getting the information of any particular asset from database on User Interface such as Assetypecode, AssetypeId, description etc. It is designed to provide many facilities to the user like – those asset are not providing a better performance then the user also declared that asset as Scrap and add it into the Scraplist into ScrapDeclaration database.

3.3.2 Information Description:-

Fixed Assets is a Project which is useful to the organization to maintain the full record of their assets(property) of the organization. Fixed Assets contain many features like AssetTypeMaster, AssetTypeAttribute, ScrapDeclaration, ScrapAuthorization etc.

Firstly, the user have to login with their Unique Id and password. After that the user will go and use the module on which the user needs to work. Now user can see all the data on User Inteface in Datatable this will help the user to check that how many types of assets organization have in Present. The data of ScrapDeclaration the assets which are already included in Scraplist are approved by the Higher Authority only for sale of Scrap. After checking that this assets are really in Scrap Condition or not?

There is an admin panel in the system. The admin will add new data and delete the data with login details etc. Admin update the information of the asset and .Admin can also edit the details of the assets if there is any problem in the database. Admin can check SQL queries if any query creating some problem for saving data or fetvhing data to grid. Admin also see complain messages from users and handle or resolve them.

3.3.3 Functional Description :-

Fixed Assets System work with the help of following modules:

1. User Login
2. Home Page
3. Menu
4. Admin
5. Modules

1. **User Login:** If any user wants to access the data or work on this project, first of all he/she login with their Id and Password. This module is responsible for that operation. User have to provide their details unique Id and Password to use the Modules functionalities. provides
2. **Home Page:** Once the user logged in or he/she login into the system. This module provides all the rights to use all the modules of the project.
3. **Menu:** When the use logged in there is a menu containing all the modules of the Project having different specifications. So, the user can use the module on which he have to work. And also he is able to see all the data on the Grid which is saved into Database.

4. **Admin:** Admin have the control to access everything on this system. Admin add new data with their details and can also delete. Admin update the details of the assets when required. Admin can see all the data in their system, saved by the organization.
5. **Module:** Every Module has its own specifications and has totally different workings. Module division specifies that the user only use one module at a time according to the requirement.

CHAPTER 4

DESIGN

4.1 ENTITY RELATIONSHIP DIAGRAM

An ER diagram is a model that identifies the concept or entities that exist in a system and the relationships between those entities. An ERD is often used as a way to visualize a relational database: each entity represents a database table and the relationship lines represents the key in one table that point to specific records in related tables.

Advantages of ER diagram

- Professional and faster Development.
- Productivity Improvement.
- Fewer Faults in Development.
- Maintenance becomes easy.

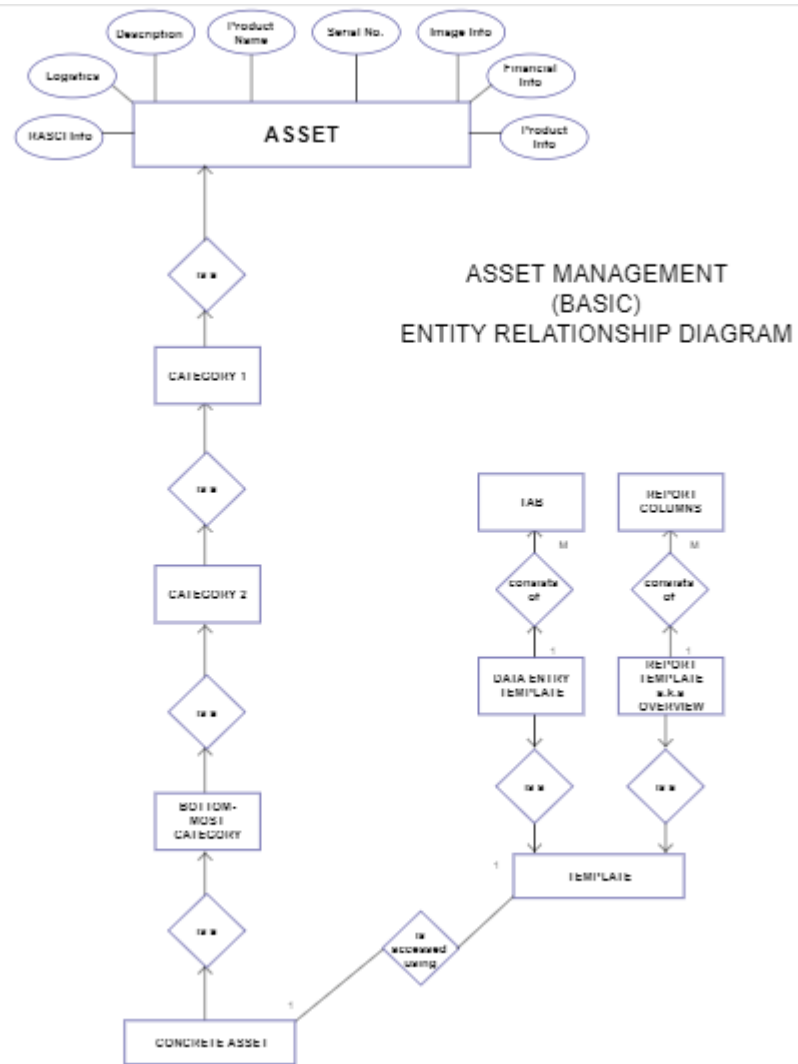


Fig 4.2:- ER Diagram

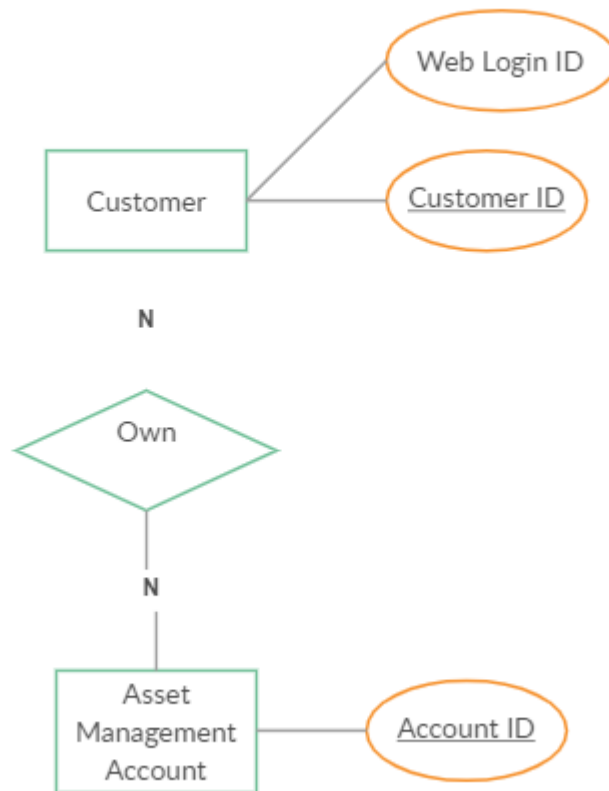


Fig 4.3:- Asset Management Diagram

4.3 DATA DICTIONARY

Tables Used

1. AssetTypeMaster

S.No.	Attribute	Data Type	Size	Description
1.	AssetTypeID	Varchar (Not Null)	20	This is the primary key which stores unique ID of the users
2.	UserID	varchar	20	This stores the UserID of the user.
3.	AssetTypeCode	Varchar	20	This stores the AssetTypeCode of the user.
4.	AssetTypeDescription	Varchar	200	This stores the AssetTypeDescription of the users.
5.	Createdon	Time	20	This stores the time and date of saved data.

Table 4.3.1:- AssetTypeMaster

2. AssetTypeAttribute

S.No.	Attribute	Data Type	Size	Description
1.	AttributeID	Varchar (Not Null)	20	This is primary key which stores the unique ID for user.
2.	AttributeName	Varchar	20	This field stores the Attributename of the user.
3.	AttributeTypeID	Varchar	20	This field stores the AttributetypeId of the user.
4.	AttributeLength	varchar	20	This field stores the AttributeLength of the user.
5.	AttributePrecision	varchar	20	This field stores the DecimalPlace.
6.	UserID	varchar	20	This stores the UserId .
7.	Createdon	Date & Time	20	This stores the Date & Time of saved data.

Table 4.3.2:- AssetTypeAttribute

3. ScrapDeclaration

S.No.	Attribute	Data Type	Size	Description
1.	ScrapID	Varchar (Not Null)	20	This is the primary key which stores unique ID of Scrap Asset.
2.	ScrapDate	varchar	20	This stores the Scrapdate of the selected asset.
3.	AssetCode	varchar	20	This stores Assetcode of the selected asset.
4.	AssetDescription	varchar	20	This stores AssetDescription of the selected asset.
5.	UserId	varchar	20	This will store UserId.
6.	AssetTypeID	varchar	20	This will store AssettypeIDof the selected asset.
7.	Createdon	varchar	Date& Time	This Stores Date & Time of saved asset.

Table 4.3.3:- ScrapDeclaration

4. ScrapAuthorization

S.No.	Attribute	Data Type	Size	Description
1.	ScrapID	Varchar (Not Null)	20	This is the primary key which stores unique ID of Scrap Asset.
2.	ScrapDate	varchar	20	This stores the Scrapdate of the asset.
3.	AssetCode	varchar	20	This stores Assetcode of the asset.
4.	AssetDescription	varchar	20	This stores AssetDescription of the asset.
5.	UserId	varchar	20	This will store UserId.
6.	Approval Date	varchar	20	This will store Approvaldate set by the Authority.
7.	Approved By	varchar	20	This stores UserId of person who Approved the asset.
8.	Createdon	varchar	Date& Time	This Stores Date & Time of saved asset.

Table 4.4.4:- ScrapAuthorization

CHAPTER – 5

USER INTERFACES / CODING

5.1 User Interface

5.1.1 Asset Type Master (Home Page)

Asset Type Master

Asset Type Attribute

Asset Type Code *

Asset type description *

☒ Active

Save Reset

Global Filter

Asset Type Code	Asset Type Description	Active
r	gfdgdg	N
TV	TABLES	N
T	TABLE	N
t	tesrefgyuh	N

5.1.2 Asset Type Attribute (Home Page)

localhost:4300/assettypemaster

Apps 2 Zimbra Web Client...

Reading li

CampusLynx

Asset Type Master

FIXED ASSETS MANAGEMENT

Masters

Asset Type Master

Asset Head Master

Asset Sub Head Master

Depreciation Rate Master

Asset Master

Fixed Assets Transactions

Fixed Asset Reports

Asset Type Master

Asset Type Attribute

Asset Type Code *

Asset Type Description *

Confirm

Attribute Type *

Alphanumeric

Attribute Name *

Mandatory

Length Max(20/200)

Decimal places(2)

Save

Reset

Global Filter

Attribute Name	Attribute Type	Length	Decimal Size	Mandatory
No records found				

Activate Windows
Go to Settings to activate Windows.

This Page consists some fields in which user entered some unique code for the new asset and write the full description of the asset.

5.1.3 Scrap Declaration (Home Page)

← → ↻ ⓘ localhost:4300/scrapdeclaration ☆ 👤 ⋮
Apps Zimbra Web Client... | Reading list

CampusLynx

FIXED ASSETS MANAGEMENT

- Masters
- Fixed Assets Transactions
 - Asset Value Addition
 - Asset Allocation
 - Asset Allocated Return
 - Asset Transfer (Dev. Inprogress)
 - AMC Contracts / Renewals
 - Insurance Details / Renewals
 - Scrap Declaration**
 - Scrap Authorization
 - Asset Sales

Scrap Declaration

Scrap Date *

Asset Code *

Asset description *

Global Filter

Attribute Name	Attribute Value
No records found	

Gross Value *

Accumulated Depreciation *

Net Value *

Remarks *

Activate Windows
Go to Settings to activate Windows.
Save Reset

5.1.4 Scrap Authorization

← → ↻ 📄 localhost:4300/scrapauthorization ☆ 👤 ⋮

📄 Apps 📄 Zimbra Web Client...

🏠 Zimbra Web Client Sign In
https://172.16.7.49

🏠 👤 Admin ▾

CampusLynx

FIXED ASSETS MANAGEMENT

- 👤 Masters >
- 👤 Fixed Assets Transactions ▾
 - 👤 Asset Value Addition
 - 👤 Asset Allocation
 - 👤 Asset Allocated Return
 - 👤 Asset Transfer (Dev. Inprogress)
 - 👤 AMC Contracts / Renewals
 - 👤 Insurance Details / Renewals
 - 👤 Scrap Declaration
 - 👤 **Scrap Authorization**
 - 👤 Asset Sales

Scrap Authorization

Approval Date * 📅

🔍 Global Filter 📄

<input type="checkbox"/>	Scrap Date	Asset Code	Asset Description
<input type="checkbox"/>	02/07/2021	F001010001	AC/1
<input type="checkbox"/>	08/07/2021	F001010001	AC/1
<input type="checkbox"/>	01/07/2021	F001010002	AC/2
<input type="checkbox"/>	02/07/2021	F001010005	AIR CONDITIONER
<input type="checkbox"/>	01/07/2021	F001010008	BATTERY1

📄 Save 🔄 Reset

Activate Windows
Go to Settings to activate Windows.

5.2 Coding

5.2.1 Asset Type Master(Component.html) Code

```
<p-growl [(value)]="msgs" life="3000" sticky="false" baseZIndex="4"></p-growl>
<div class="page-layout simple fullwidth angular-material-elements">
  <div class="header accent p-24 h-60 pt-36" fxLayout="column" fxLayoutAlign="center
center" fxLayout.gt-xs="row"
fxLayoutAlign.gt-xs="space-between center">
    <div fxLayout="column" fxLayoutAlign="center center" fxLayout.gt-xs="column"
fxLayoutAlign.gt-xs="center start">
      <div fxLayout="row" fxLayoutAlign="start center">
        <mat-icon>group_add</mat-icon>
        <span>&nbsp;&nbsp;&nbsp;</span>
        <span class="h2">Asset Type Master</span>
      </div>
    </div>
  </div>
</div>
<div class="content p-24">
  <form [formGroup]="assettypemasterform" autocomplete="off">
    <mat-tab-group>
      <mat-tab label="Asset Type Master">
        <mat-card>
          <div fxLayout="row" fxLayoutAlign="start center">
            <mat-form-field class="pl-4 pr-4" fxFlex="25">
              <mat-label>Asset Type Code</mat-label>
              <input matInput formControlName="Assettypecode" maxlength="2" required>
              <mat-error *ngIf="formErrors.Assettypecode">
                Asset Type Code is required!
              </mat-error>
            </div>
          </div>
        </mat-card>
      </mat-tab>
    </mat-tab-group>
  </form>
</div>
```



```

</mat-form-field>
<input matInput formControlName="assettypeid" hidden>
    <div fxFlex="5"></div>
<mat-form-field class="pl-4 pr-4" fxFlex="50">
    <mat-label>Asset type description</mat-label>
    <input matInput formControlName="Assettypedescription" maxlength="100"
required>
    <mat-error *ngIf="formErrors.Assettypedescription">
        Asset Type Description is required!
    </mat-error>
</mat-form-field>
</div>
<div fxLayout="row" fxLayoutAlign="space-between center">
    <mat-checkbox formControlName="isActive" fxFlex="25">Active</mat-
checkbox>
    <div fxLayout="row">
        <app-savebutton
(onClick)="saveAssetTypeMaster(assettypemasterform.value)" [showbutton]="true"
        [disabledyn]="!assettypemasterform.value.Assettypecode ||
!assettypemasterform.value.Assettypedescription">
    </app-savebutton>
    <app-resetbutton (onClick)="clearForm()" [disabledyn]="false"
[showbutton]="true">
    </app-resetbutton>
    </div>
</div>
<br>
<div class="test-container" aria-readonly="true">
    <div class="ui-widget-header" style="margin-top:20px;padding:8px;">
        <div fxLayout="row" fxLayoutAlign="space-between center">
            <div fxLayout="row" fxLayoutAlign="start center">
                <mat-icon class="s-18 secondary-text mat-icon material-icons mat-icon-no-

```

```

color">
    search
    </mat-icon>
    <input #gb1 type="text" pInputText size="50" placeholder="Global Filter"
class="ui-inputtext">
    </div>
    <button mat-button color="accent" matTooltip="export to excel">
        <mat-icon>description</mat-icon>
    </button>
</div>
</div>
<p-dataTable selectionMode="single" [globalFilter]="gb1"
[value]="assettypelist" [rows]="10"
[paginator]="true" [pageLinks]="3" [responsive]="true" [tableStyle]="{'table-
layout':'auto'}" #dt>
    <p-column [style]="{'width':'3em'}">
        <ng-template let-editrow="rowData" pTemplate type="body">
            <mat-icon color="accent"
(click)="editSelectedRow(editrow)">create</mat-icon>
        </ng-template>
    </p-column>
    <p-column field="assettype" header="Asset Type Code"
[style]="{'width':'10em'}"></p-column>
    <p-column field="assettypename" header="Asset Type Description"></p-
column>
    <p-column field="active" header="Active" [style]="{'width':'6em'}"></p-
column>
</p-dataTable>
</div>
</mat-card>
</mat-tab>

<!--tab-2-->

```

```

<mat-tab label="Asset Type Attribute">
  <mat-card>
    <div fxLayout="row" fxLayoutAlign="start center" [class]="disableupperpart">
      <mat-form-field fxFlex="25" class="pl-4 pr-4">
        <mat-label>Asset Type Code</mat-label>
        <input matInput placeholder="Enter Asset Type Code" maxlength="2"
formControlName="assettypecode"
        required readonly>
        <button mat-icon-button (click)="assettypecodeLOV()" matSuffix>
          <mat-icon>search</mat-icon>
        </button>
        <mat-error *ngIf="formErrors.assettypecode">
          Asset Type code is required!
        </mat-error>
      </mat-form-field>
    </div>
    <div fxFlex="5"></div>
    <mat-form-field class="pl-4 pr-4" fxFlex="50">
      <mat-label>Asset Type Description</mat-label>
      <input matInput formControlName="assettypedescription" maxlength="100"
required readonly>
      <mat-error *ngIf="formErrors.assettypedescription">
        Asset Type Description is required!
      </mat-error>
    </mat-form-field>
    <div fxFlex="5"></div>
    <app-confirmbutton (onClick)="confirm()" [disabledyn]="false"
[showbutton]="true">
    </app-confirmbutton>
  </div>
  <div fxLayout="row" fxLayoutAlign="start center">
    <mat-form-field fxFlex="25" class="pl-4">

```

```

    <mat-label>Attribute Type</mat-label>
    <mat-select placeholder="Select Attribute Type"
formControlName="attributetype"
      (selectionChange)="renewalslelection($event.value)" required>
      <mat-option value="A">Alphanumeric</mat-option>
      <mat-option value="N">Numeric</mat-option>
      <mat-option value="C">Currency</mat-option>
      <mat-option value="D">Date</mat-option>
    </mat-select>
  </mat-form-field>
</div fxFlex="5"></div>
<mat-form-field fxFlex="50" class="pl-4 pr-4">
  <mat-label>Attribute Name</mat-label>
  <input matInput placeholder="Enter Attribute Name"
formControlName="attributename" required
    maxlength="100">
  <mat-error *ngIf="formErrors.attributename">
    Attribute Name is required!
  </mat-error>
</mat-form-field>
<input matInput formControlName="attributeid" hidden>
<div fxFlex="10"></div>

  <mat-checkbox formControlName="mandatoryckbx"
fxFlex="25">Mandatory</mat-checkbox>

```

```

</div>
<div fxLayout="row" fxLayoutAlign="start center">
  <mat-form-field fxFlex="25" class="pl-4 pr-4">
    <mat-label>Length Max(20/200)</mat-label>
  </mat-form-field>

```

```

        <input matInput appTwoDigitDecimaNumber placeholder="Enter length
max" maxLength="3" formControlName="lengthmax" [readonly]="disablelength" >

        <mat-error *ngIf="formErrors.length max">
            length is required!
        </mat-error>
    </mat-form-field>
</div fxFlex="5"></div>

<mat-form-field fxFlex="25" class="pl-4 pr-4" >
    <mat-label>Decimal places(2)</mat-label>
    <input matInput placeholder="Enter Decimal places"
appTwoDigitDecimaNumber maxLength="1" [readonly]="disabledecimal"
formControlName="decimalplace">
</mat-form-field>

<div fxFlex="25"></div>
<div fxFlex="auto" fxLayoutAlign="space-between center">
    <!-- <mat-checkbox formControlName="mandatoryckbx"
fxFlex="25">Mandatory</mat-checkbox> -->
    <div fxLayout="row">
        <app-savebutton
(onClick)="saveAssetTypeAttribute(assettypemasterform.value)" [showbutton]="true">
        </app-savebutton>
        <app-resetbutton (onClick)="clearAttributes()" [disabledyn]="false"
[showbutton]="true">
        </app-resetbutton>
    </div>
</div>
</div>
</mat-card>
<br>
<div class="test-container" aria-readonly="true">

```

```

<div class="ui-widget-header" style="margin-top:20px;padding:8px;">
  <div fxLayout="row" fxLayoutAlign="space-between center">
    <div fxLayout="row" fxLayoutAlign="start center">
      <mat-icon class="s-18 secondary-text mat-icon material-icons mat-icon-no-
color">
        search
      </mat-icon>
      <input #gb2 type="text" pInputText size="50" placeholder="Global Filter"
class="ui-inputtext">
    </div>
    <button mat-button color="accent" matTooltip="export to excel">
      <mat-icon>description</mat-icon>
    </button>
  </div>
</div>

<p-dataTable selectionMode="single" [globalFilter]="gb2" [value]="attributes"
[rows]="10" [paginator]="true"
[pageLinks]="3" [responsive]="true" [tableStyle]="{'table-layout':'auto'}" #dt1>
  <p-column [style]="{'width':'3em'}">
    <ng-template let-editrow="rowData" pTemplate type="body">
      <mat-icon color="accent" (click)="editSelectedRow(editrow)">create</mat-
icon>
    </ng-template>
  </p-column>
  <p-column field="attributename" header="Attribute Name"></p-column>
  <p-column field="attributetypeid" header="Attribute Type"></p-column>
  <p-column field="attributelength" header="Length"
[style]="{'width':'6em'}"></p-column>
  <p-column field="attributeprecision" header="Decimal Size"
[style]="{'width':'6em'}"></p-column>
  <p-column field="attributemandatory" header="Mandatory"
[style]="{'width':'6em'}"></p-column>
</p-dataTable>

```

```

        </div>
    </mat-tab>
</mat-tab-group>
</form>
</div>
</div>

<ng-container *ngIf="popup">
    <app-lov-component [columnDefs]="columnDefs" [popuptitle]="popuptitle"
    [rowData]="rowData"
        (popup)="popupClose($event)" (popupHide)="hidelo($event)"></app-lov-
    component>
</ng-container>

```

5.3.1 Scrap Declaration(Component.html)

```

<p-growl [(value)]="msgs" life="3000" sticky="false" baseZIndex="4"></p-growl>
<div class="page-layout simple fullwidth angular-material-elements">
    <div class="header accent p-24 h-60 pt-36" fxLayout="column"
    fxLayoutAlign="center center" fxLayout.gt-xs="row"

```

```

fxLayoutAlign.gt-xs="space-between center">
  <div fxLayout="column" fxLayoutAlign="center center" fxLayout.gt-xs="column"
fxLayoutAlign.gt-xs="center start">
    <div fxLayout="row" fxLayoutAlign="start center">
      <mat-icon>group_add</mat-icon>
      <span>&nbsp;&nbsp;&nbsp;</span>
      <span class="h2">Scrap Declaration</span>
    </div>
  </div>
</div>

<div class="content p-24">
  <form [formGroup]="scrapdeclarationform" autocomplete="off">
    <mat-card style="background: #e6f0ff;">
      <div fxLayout="row" fxLayoutAlign="start center">
        <mat-form-field fxFlex="20" class="pl-4">
          <mat-label>Scrap Date</mat-label>
          <input matInput [matDatepicker]="picker1" #podate
formControlName="scrapdate" placeholder="dd/MM/yyyy"
          readonly required>
          <mat-datepicker-toggle matSuffix [for]="picker1"></mat-datepicker-toggle>
          <mat-datepicker #picker1></mat-datepicker>
        </mat-form-field>
      </div>
      <div fxLayout="row" fxLayoutAlign="start center">
        <mat-form-field fxFlex="25" class="pl-4">
          <mat-label>Asset Code</mat-label>

          <input matInput placeholder="Select Asset Code" maxlength="20"
formControlName="assettypecode" required readonly>
          <button mat-icon-button (click)="assettypecodeLOV()" matSuffix>

```



```

        <mat-icon>search</mat-icon>
    </button>
    <mat-error *ngIf="formErrors.assettypecode">
        Asset Type Code is required!
    </mat-error>
</mat-form-field>
<input matInput formControlName="scrapid" hidden>
<input matInput formControlName="assetid" hidden>

<div fxFlex="15" class="pl-4 pr-4"></div>
<mat-form-field class="pl-4 pr-4" fxFlex="35">
    <mat-label>Asset description</mat-label>
    <input matInput formControlName="assettypedescription" maxlength="100"
required>
    <!-- <mat-error *ngIf="formErrors.assettypedescription">
        Asset Type Description is required!
    </mat-error> -->
</mat-form-field>
</div>
<div class="test-container" aria-readonly="true">
    <div class="ui-widget-header" style="margin-top:20px;padding:8px;">
        <div fxLayout="row" fxLayoutAlign="space-between center">
            <div fxLayout="row" fxLayoutAlign="start center">
                <mat-icon class="s-18 secondary-text mat-icon material-icons mat-icon-no-
color">
                    search
                </mat-icon>
                <input #gb1 type="text" pInputText size="50" placeholder="Global Filter"
class="ui-inputtext">
            </div>
            <button mat-button color="accent" matTooltip="export to excel">

```

```

        <mat-icon>description</mat-icon>
    </button>
</div>
</div>

<p-dataTable selectionMode="single" [value]="attributelist" scrollable="true"
scrollHeight="500px" [rows]="10"
    [globalFilter]="gb1" [responsive]="true" #dt>
    <!-- <p-column>
<ng-template let-editrow="rowData" pTemplate type="body">
    <mat-icon color="accent" (click)="editSelectedRow(editrow)">create</mat-
icon>
</ng-template>
</p-column> -->
    <p-column field="attributename" header="Attribute Name"></p-column>
    <p-column field="attributevalue" header="Attribute Value"></p-column>
</p-dataTable>
</div>
<div fxLayout="row" fxLayoutAlign="start center">
    <mat-form-field class="pl-4 pr-4" fxFlex="20">
        <mat-label>Gross Value</mat-label>
        <input matInput formControlName="grossvalue" maxlength="20" required>
        <!-- <mat-error *ngIf="formErrors.grossvalue">
            Please Enter Gross Value!
        </mat-error> -->
    </mat-form-field>
    <div fxFlex="8" class="pl-4 pr-4"></div>
    <mat-form-field class="pl-4 pr-4" fxFlex="30">
        <mat-label>Accumulated Depreciation</mat-label>
        <input matInput formControlName="accumulateddepreciation" maxlength="20"
required>
        <!-- <mat-error *ngIf="formErrors.accumulateddepreciation">

```

```

        Please Enter Accumulated Depreciation!
    </mat-error> -->
</mat-form-field>
<div fxFlex="8" class="pl-4 pr-4"></div>
<mat-form-field class="pl-4 pr-4" fxFlex="20">
    <mat-label>Net Value</mat-label>
    <input matInput formControlName="netvalue" maxlength="20" required>
    <!-- <mat-error *ngIf="formErrors.netvalue">
        Please Enter Net Value!
    </mat-error> -->
</mat-form-field>
</div>
<div fxLayout="row" fxLayoutAlign="start center">
    <mat-form-field class="pl-4 pr-4" fxFlex="50">
        <mat-label>Remarks</mat-label>
        <input matInput formControlName="remarks" maxlength="200" required>
        <!-- <mat-error *ngIf="formErrors.remarks">
            Please Enter Remarks!
        </mat-error> -->
    </mat-form-field>
</div>
<div fxLayout="row" fxLayoutAlign="end center">
    <app-savebutton (onClick)="save(scrapdeclarationform.value)"
[showbutton]="true"></app-savebutton>
    <app-resetbutton (onClick)="Clearform()" [disabledyn]="false"
[showbutton]="true">
    </app-resetbutton>
</div>
</mat-card>
</form>
</div>

```

```

</div>
<ng-container *ngIf="popup">
  <app-lov-component [columnDefs]="columnDefs" [popuptitle]="popuptitle"
  [rowData]="rowData"
    (popup)="popupClose($event)" (popupHide)="hidelo($event)"></app-lov-
  component>
</ng-container>

```

5.3.2 Scrap Declaration(Component.ts) AngularJs Code

```

import { DatePipe } from '@angular/common';
import { Component, OnInit } from '@angular/core';
import { FormBuilder, FormControl, FormGroup } from '@angular/forms';
import { DateAdapter, MAT_DATE_FORMATS, MAT_DATE_LOCALE } from
'@angular/material';
import { MomentDateAdapter } from '@angular/material-moment-adapter';
import { BaseComponent } from 'app/commonmodules/base.component';
import { ExcelService } from 'app/commonmodules/Service/excel.service';
import { FormerrorService } from 'app/commonModules/Service/formerror.service';
import { info } from 'console';
import { ScrapDeclarationService } from './scrapdeclaration.service';
import { AssetTypeAttribute, ScrapInfo } from './viewmodel/scrapinfo_vm';

export const MY_FORMATS = {
  parse: {
    dateInput: 'DD/MM/YYYY',
  },
  display: {
    dateInput: 'DD/MM/YYYY',
    monthYearLabel: 'MMM YYYY',

```

```

    dateAllLabel: 'DD/MM/YYYY',
    monthYearAllLabel: 'MM YYYY',
  },
};

```

```

@Component({
  selector: 'app-scrapdeclaration',
  templateUrl: './scrapdeclaration.component.html',
  styleUrls: ['./scrapdeclaration.component.scss'],
  providers: [
    { provide: DateAdapter, useClass: MomentDateAdapter, deps:
[MAT_DATE_LOCALE] },
    { provide: MAT_DATE_FORMATS, useValue: MY_FORMATS },
  ]
})

```

```

export class ScrapdeclarationComponent extends BaseComponent implements OnInit {
  scrapdeclarationform: FormGroup;
  // scrapdeclaration: ScrapDeclaration[];
  formErrors: any;
  allerror: any;
  rowData: any;
  error: any;
  popuptitle: string;
  columnDefs: { headerName: string; field: string; width: number; }[];
  popup: boolean;
  fieldName: string;
  scrapdate: any;
  scraplist: any = [];
  assettypeid: any;
  scrapidtostopduplicatearraypush: any;

```

```

dummyarray: any;
deleteindex: any;
assettypecode: any;
assettypedescription: any;
scrapinfo: ScrapInfo;
profitcentreid: any;
gridlist: any;
gridData: any;
gridDataforreport: AssetTypeAttribute[];
exceldata: any[];
dataTable: any;
filteredValue: any;
assetid: any;
attributelist: any;
constructor(
    private service: ScrapDeclarationService,
    public formBuilder: FormBuilder,
    private errsvc: FormerrorService,
    private excelService: ExcelService,
    private datepipe: DatePipe
) {
    super();
    this.formErrors = {
        scrapdate: {},
        assettypecode: {},
        assettypedescription: {},
        grossvalue: {},
        accumulateddepreciation: {},
        netvalue: {},
    }
}

```

```
    remarks: {}  
  }  
}
```

```
ngOnInit(): void {  
  this.errsvc.getjson().subscribe(data => {  
    this.allerror = data;  
  });  
  this.scrapdeclarationform = new FormGroup({  
    scrapid: new FormControl(null),  
    scrapdate: new FormControl(null),  
    assettypecode: new FormControl(null),  
    assettypedescription: new FormControl(null),  
    grossvalue: new FormControl(null),  
    accumulateddepreciation: new FormControl(null),  
    netvalue: new FormControl(null),  
    remarks: new FormControl(null),  
    assettypeid: new FormControl(null),  
    assetid: new FormControl(null),  
    assettype: new FormControl(null),  
    profitcentreid: new FormControl(null)  
  });  
  this.scraplist = [];  
}
```

```
Clearform() {  
  this.scrapdeclarationform.controls.scrapdate.reset();  
  this.scrapdeclarationform.controls.assettypecode.reset();  
  this.scrapdeclarationform.controls.assettypedescription.reset();  
  this.scrapdeclarationform.controls.grossvalue.reset();  
}
```

```

this.scrapdeclarationform.controls.accumulateddepreciation.reset();
this.scrapdeclarationform.controls.netvalue.reset();
this.scrapdeclarationform.controls.remarks.reset();
this.scrapdeclarationform.reset();
this.attributelist = [];
this.attributelist = [...this.attributelist];

}

save(formdata: any): void {
  if (!formdata.scrapdate) {
    this.showError(this.allerror.scrapdate, 'Please enter Scrap date!')
  }
  if (!formdata.assettypecode) {
    this.showError(this.allerror.assettypecode, 'Please enter Asset Code!')
  }

  const JsonData =
  {
    'profitcentreid': localStorage.getItem("profitcentreid"),
    // 'profitcentreuniqueid': localStorage.profitcentreuniqueid,
    'companyid': localStorage.getItem("companyid"),
    // 'companyid': 'bbb',// localStorage.companyid,
    'scrapdate': this.datepipe.transform(formdata.scrapdate, 'dd/MM/yyyy'),
    'assettypecode': formdata.assettypecode,
    'assettypedescription': formdata.assettypedescription,
    'grossvalue': formdata.grossvalue,
    'accumulateddepreciation': formdata.accumulateddepreciation,
    'netvalue': formdata.netvalue,
    'remarks': formdata.remarks,
  }

```



```

// 'assettypeid': this.assettypeid,
'assetid': this.assetid,
'scrapid': formdata.scrapid,
'userid': localStorage.userid,
'approvaldate': formdata.approvaldate,
'approvedby': formdata.approvedby,
'status': formdata.status
// 'assettype': formdata.assettype
};
this.service.save(JsonData).subscribe(response => {
  this.Clearform();
  if (response.status.responseStatus === 'Success') {
    this.showSuccess(response.response.status, "");
  } else {
    this.showError(this.error.error.response.status, "")
  }
}, error => {
  if (error.status === 0) {
    this.showError(this.allerror.SERVER_NOT_FOUND, "");
  } else {
    this.showError(error.error.response.status, "");
  }
});
}

```

```

assettypecodeLOV(){
  this.service.assettypecodeLOV().subscribe(info => {
    if (info.status.responseStatus === 'Success') {
      this.rowData = info.response.assetcodelist;
    }
  });
}

```

```

        this.popupTitle = 'List of Asset Type Code and Description';
        this.columnDefs = [
            { headerName: 'Asset Code', field: 'assetcode', width: 100 },
            { headerName: 'Asset Description', field: 'assetdescription', width: 100 }
        ];
        this.fieldName = 'assettypecode';
        this.popup = true;
    }
}, error => {
    if (error.status === 0) {
        this.showError(this.allerror.SERVER_NOT_FOUND, "");
    } else {
        this.showError(error.error.reponse.status, "");
    }
});
}

getGridValue(Json) {

    this.service.getGridValueOnBasisOfAssetTypeCode(Json).subscribe(info => {
        if (info.status.responseStatus === 'Success') {
            this.attributelist = info.response.gridlist;
            console.log(this.scraplist);

        } else {
            this.showError(info.response.status, "");
        }
    }, error => {
        if (error.status === 0) {
            this.showError(this.allerror.SERVER_NOT_FOUND, "");
        }
    });
}

```

```

    } else {
        this.showError(error.error.response.status, "");
    }
})
}

```

```

popupClose(popup: any): void {
    if (this.fieldName === 'assettypecode') {
        this.scrapdeclarationform.controls.assettypecode.setValue(popup.assetcode);

this.scrapdeclarationform.controls.assettypedescription.setValue(popup.assetdescription
);
        // this.scrapdeclarationform.controls.assettypeid.setValue(popup.assettypeid);
        this.scrapdeclarationform.controls.assetid.setValue(popup.assetid);
        // this.scrapdeclarationform.controls.attributename.setValue(popup.attributename);
        // this.scrapdeclarationform.controls.attributevalue.setValue(popup.attributevalue);
        this.assettypeid = popup.assettypeid;
        this.assetid = popup.assetid;
        this.assettypecode = popup.assetdescription;
        const json = {
            'assetid': popup.assetid,
        }
        // this.getGridValue(json);
        this.getattributedata();
    }
    this.popup = false;
}

```

```

hidelo(event: any): void {
  this.popup = event;
}

getattributedata(): void {
const jsondata = {
  'assettypeid': this.assettypeid,
  'assetid': this.assetid
};
this.service.getattributedata(jsondata).subscribe(info => {
  if (info.status.response.Status === 'Success') {
    this.attributelist = info.response.list;
  } else {
    this.showError(info.response.status, 'Error Occured');
  }
}, error => {
  if (error.status === 0) {
    this.showError(this.allerror.SERVER_NOT_FOUND, "");
  } else {
    this.showError(error.error.response.status, "");
  }
})
}
}

```

5.4.1 Scrap Authorization(Component.html) AngularJs Code

```
<p-growl [(value)]="msgs" life="3000" sticky="false" baseZIndex="4"></p-growl>

<div class="page-layout simple fullwidth angular-material-elements">
  <div class="header accent p-24 h-60 pt-36" fxLayout="column"
    fxLayoutAlign="center center" fxLayout.gt-xs="row"
    fxLayoutAlign.gt-xs="space-between center">
    <div fxLayout="column" fxLayoutAlign="center center" fxLayout.gt-
      xs="column" fxLayoutAlign.gt-xs="center start">
      <div fxLayout="row" fxLayoutAlign="start center">
        <mat-icon>group_add</mat-icon>
        <span>&nbsp;&nbsp;&nbsp;</span>
        <span class="h2">Scrap Authorization</span>
      </div>
    </div>
  </div>
</div>

<div class="content p-24">
  <form [formGroup]="scrapauthorizationform" autocomplete="off">

    <mat-card style="background: #e6f0ff;">
      <div fxLayout="row" fxLayoutAlign="startcenter">
        <mat-form-field fxFlex="20" class="pl-4 pr-4">
          <mat-label>Approval Date</mat-label>
          <input matInput [matDatepicker]="picker1" #podate
            formControlName="approvaldate" placeholder="dd/MM/yyyy"
            readonly required>
          <mat-datepicker-toggle matSuffix [for]="picker1"></mat-datepicker-
            toggle>
          <mat-datepicker #picker1></mat-datepicker>
          <!-- <mat-error *ngIf="formErrors.approvaldate">
            Approval Date is required!
          </mat-error-->
        </mat-form-field>
      </div>
    </mat-card>
  </form>
</div>
```

```

        </mat-error> -->
    </mat-form-field>
</div>
<input matInput formControlName="assetid" hidden>
<!-- <div fxFlex="50" fxLayout="end center" class ="pl-4 pr-4"></div>
    <app-confirmbutton (click)="confirm(scrapauthenticationform.value)"
[showbutton]="true"></app-confirmbutton> -->
</mat-card>
<div class="test-container">
    <!-- header -->
    <div class="ui-widget-header" style="margin-top:20px;padding:8px">
        <div fxLayout="row" fxLayoutAlign="space-between center">
            <div fxLayout="row" fxLayoutAlign="start center">
                <mat-icon class="s-18 secondary-text mat-icon material-icons mat-
icon-no-color">
                    search
                </mat-icon>
                <input #gb3 type=text pInputText size="50" placeholder="Global
Filter" class="ui-inputtext">
            </div>
            <div fxLayout="row" fxLayoutAlign="end center">
                <button mat-button color="accent" matTooltip="export to excel">
                    <mat-icon>description</mat-icon>
                </button>
            </div>
        </div>
    </div>
</div>
<!--header close-->
<div style="overflow-x: scroll;">
    <p-dataTable [rows]="10" [pageLinks]="3" [globalFilter]="gb3"
[value]="attributelist" [responsive]="true"
[tableStyle]="{'table-layout':'auto'}" #dt>

```

```

        <p-column field="status" [style]="{ 'background-color': 'rgb(235, 237,
240)', 'text-align': 'left' }"
        frozen="true">
        <ng-template pTemplate="header">
            <input type="checkbox" [(checked)]="boolAll1"
            (click)="checkSelectedAllTagged($event.target.checked ? 'on' :
'off')"/>
        </ng-template>
        <ng-template let-row="rowData" let-index="rowIndex" let-col
pTemplate="body">
            <input type="checkbox" class="authorize-click"
            (click)="checkSelectedSingleTagged(row,$event.target.checked ?
'on' : 'off')"
            [(checked)]="row.isChecked" />
            <!-- {{row.status}} -->
        </ng-template>
    </p-column>
    <p-column field="scrapdate" header="Scrap Date"
[style]="{ 'width': '10em' }"></p-column>
    <p-column field="assetcode" header="Asset Code"></p-column>
    <p-column field="assetdescription" header="Asset Description"></p-
column>
</p-dataTable>
</div>
</div>
<div fxLayout="row" fxLayoutAlign="end center">
    <app-savebutton (onClick)="save(scrapauthorizationform.value)"
[showbutton]="true"> </app-savebutton>
    <app-resetbutton (onClick)="Clearform()" [disabledyn]="false"
[showbutton]="true">
    </app-resetbutton>
</div>
</form>

```

```

        </div>
    </div>
    <ng-container *ngIf="popup">
        <app-lov-component [columnDefs]="columnDefs" [popuptitle]="popuptitle"
        [rowData]="rowData"
            (popup)="popupClose($event)" (popupHide)="hidelo($event)"></app-
        lov-component>
    </ng-container>

```

5.4.2 Scrap Authorization(Component.ts) Angular Code

```

import { DatePipe } from '@angular/common';
import { Component, OnInit } from '@angular/core';
import { FormControl, FormGroup } from '@angular/forms';
import { DateAdapter, MAT_DATE_FORMATS, MAT_DATE_LOCALE } from
'@angular/material';
import { MomentDateAdapter } from '@angular/material-moment-adapter';
import { BaseComponent } from 'app/commonmodules/base.component';
import { FormerrorService } from 'app/commonModules/Service/formerror.service';
import { ScrapAuthorizationService } from './scrapauthorization.service';
import { ScrapAuthInfo, ScrapDeclaration } from './viewmodel/scrapauthinfo_vm';

export const MY_FORMATS = {
  parse: {
    dateInput: 'DD/MM/YYYY',
  },
  display: {
    dateInput: 'DD/MM/YYYY',
    monthYearLabel: 'MMM YYYY',

```



```

    dateAllLabel: 'DD/MM/YYYY',
    monthYearAllLabel: 'MM YYYY',
  },
};

```

```

@Component({
  selector: 'app-scrapauthorization',
  templateUrl: './scrapauthorization.component.html',
  styleUrls: ['./scrapauthorization.component.scss'],
  providers: [
    { provide: DateAdapter, useClass: MomentDateAdapter, deps:
[MAT_DATE_LOCALE] },
    { provide: MAT_DATE_FORMATS, useValue: MY_FORMATS },
  ]
})

```

```

export class ScrapauthorizationComponent extends BaseComponent implements OnInit
{
  scrapauthorizationform: FormGroup;
  scrapauthorization: ScrapDeclaration[];
  scrapauthinfo: ScrapAuthInfo;
  formErrors: any;
  approvaldate: any;
  boolAll1: boolean;
  error: any;
  allerror: any;
  gridData: any;
  assetid: any;
  attributelist: any = [];
  fieldName: string;
  assettypeid: any;

```

```

getattributedata: any;
popup: boolean;
reset: any;

constructor(
  private service: ScrapAuthorizationService,
  private errsvc: FormerrorService,
  private datepipe: DatePipe
) {
  super();
  this.formErrors = {
    approvaldate: {}
  }
}

ngOnInit(): void {
  this.errsvc.getjson().subscribe(data => {
    this.allerror = data;
  });
  this.scrapauthorizationform = new FormGroup({
    approvaldate: new FormControl(null),
    assetid: new FormControl(),
  });
  this.getGridData();
  // this.attributelist = [];
}

checkSelectedAllTagged(status: any): void {
  console.log(status);
  if (status === 'on') {
    for (const c of this.attributelist) {

```

```

        c.isChecked = true;
    }
    this.boolAll1 = true;
}
else if (status === 'off') {
    for (const c of this.attributelist) {
        c.isChecked = false;
    }
    this.boolAll1 = false;
}
}

checkSelectedSingleTagged(item: any, status: any): void {
    if (status === 'on') {
        item.isChecked = true;
    } else {
        item.isChecked = false;
    }
    if (this.attributelist.filter(x => x.isChecked).length === this.attributelist.length)
        this.boolAll1 = true;
    else
        this.boolAll1 = false;
}

getGridData() {
    this.service.getGrid().subscribe(info => {
        if (info.status.responseStatus === 'Success') {
            this.attributelist = info.response.gridlist;
            console.log(this.attributelist);
            for (const c of this.attributelist) {

```

```

        c.scrapdate = this.datepipe.transform(c.scrapdate, 'dd/MM/yyyy')
    }
    this.attributelist = [... this.attributelist];

}
}, error => {
    if (error.status === 0) {
        this.showError(this.allerror.SERVER_NOT_FOUND, "");
    } else {
        this.showError(error.error.response.status, "");
    }
});
}

save(formData: any) {
    if (!formData.approvaldate) {
        this.showError(this.allerror.approvaldate, 'Approval date is required!');
        return;
    }
    const JsonData = {
        'scrapid': localStorage.scrapid,
        // 'companyuniqueid': localStorage.companyid,
        'approvaldate': this.datepipe.transform(formData.approvaldate, 'dd/MM/yyyy'),
        'assetid': this.attributelist[0].assetid,
        'assetlist': this.attributelist.filter(x => x.isChecked),
        'userid': localStorage.userid,

    };

```

```

this.service.save(JsonData).subscribe(info => {
    if (info.status.responseStatus === 'Success') {
        this.showSuccess(info.response.status, "");
        this.Clearform();
        this.getGridData();

    } else {
        this.showError(info.response.status, "");
    }
}, error => {
    if (error.status === 0) {
        this.showError(this.allerror.SERVER_NOT_FOUND, "");
    } else {
        this.showError(error.error.response.status, "");
    }
})
}

```

```

Clearform() {
    this.scrapauthorizationform.controls.approvaldate.reset();
    // this.attributelist = [];
    this.attributelist = [...this.attributelist];

}

```

```

popupClose(popup: any): void {
    if (this.fieldName === 'assetid') {
        this.scrapauthorizationform.controls.assettypecode.setValue(popup.assetcode);

```

```

this.scrapauthorizationform.controls.assetdescription.setValue(popup.assetdescription);

```

```
this.scrapauthorizationform.controls.assettypeid.setValue(popup.assettypeid);
this.assettypeid = popup.assettypeid;
this.assetid = popup.assetid;
this.getattributedata();
}
this.popup = false;
}
hidelo(event: any): void {
    this.popup = event;
}
}
```

CHAPTER 6

TEST AND VALIDATION CRITERIA

6.1 ERROR HANDLING

An Exception occurs when a program encounter any unexpected problems. Such as running out of memory or attempting to read from a file that no longer exists. These problems are not necessarily caused by a programming error but they mainly occur because of violation of assumption that you might have made about the execution environment. When a program encounters an exception the default behavior is to throw the exception which generally translates to abruptly, terminating the program after displaying an error message. But this is not a characteristic of a robust application.

But the best way is to bindle the exception situations if possible, gracefully recover from them. This is called “exception handling”.

I used try, catch, finally and throw in my project to handle the exception.

The Try Block:

Place the code that might cause exception in a try block. A typical try block looks likethis

```
Try
{
//Code that may cause exception
}
```

A try block can have another try block inside when an exception occurs at any point rather than executing any further lines of code, the CLR (Common Language Runtime)

Searches for the nearest try block that encloses this code. This code. The control is then passed to a matching catch block if any and then to the finally block associated with this try block.

Catch Block:

There can be no of catch blocks immediately following a try block. Each catch block handles an exception of a particular type. When an exception occurs in a statement placed inside the try block the CLR looks for a mainly block that is capable of handling the type of exception.

Throw block:

A throw statement explicitly generates an exception in code. You can throw when a particular path in code results in an anomalous situation.

Finally Block:

The finally block contains the code that always executes whenever or not any exception occurs.

6.2 PARAMETER PASSING

Passing parameters from one page to another is a very common task in Web development. There are still many situations in which you need to pass data from one Web page to another. One of the simplest and most efficient ways of passing parameters among pages is to use the query string. Unfortunately, packing data into the query string via string manipulations can quickly lead to cumbersome and often difficult to maintain code, especially as the parameter list grows. To overcome this problem, I've used Session in my project.

Query String some of them are described below

- Query String is client side. But Session is server side.
- The information or data stored in Query String is visible to everyone. But in Session it is hidden and can't be viewed easily.
- Query String can store only a piece of information but in Session we can store the more and more data. The Query String speed never falls as the load increases because it stores a piece of information. But on the other hand Session increases congestion as the loads increase.

6.3 Validation Checks

- 1) **Date Validation:** The validation on date data type has been specified to be of the format DD/MM/YY. Any other format is unacceptable.
- 2) **Time validation:** The validation on time data type has been specified to be of the format hours-minutes-seconds. Any other format is unacceptable.
- 3) **Number field validation:** The field specified with number as then their data- type willnot accept character.
- 4) **User Authentication:** When a Customer/user logs on to the system to access data from tables and database, the Id & password needs to be checked.
- 5) **Password change Validation:** Only authorized users are allowed to change the password and the process requires asking the old password before changing it to the new one.

6.4 TESTING

Test Case Execution

The workflow diagram below depicts the high level steps necessary to follow in order to set up and execute test based on the Test Case Template.

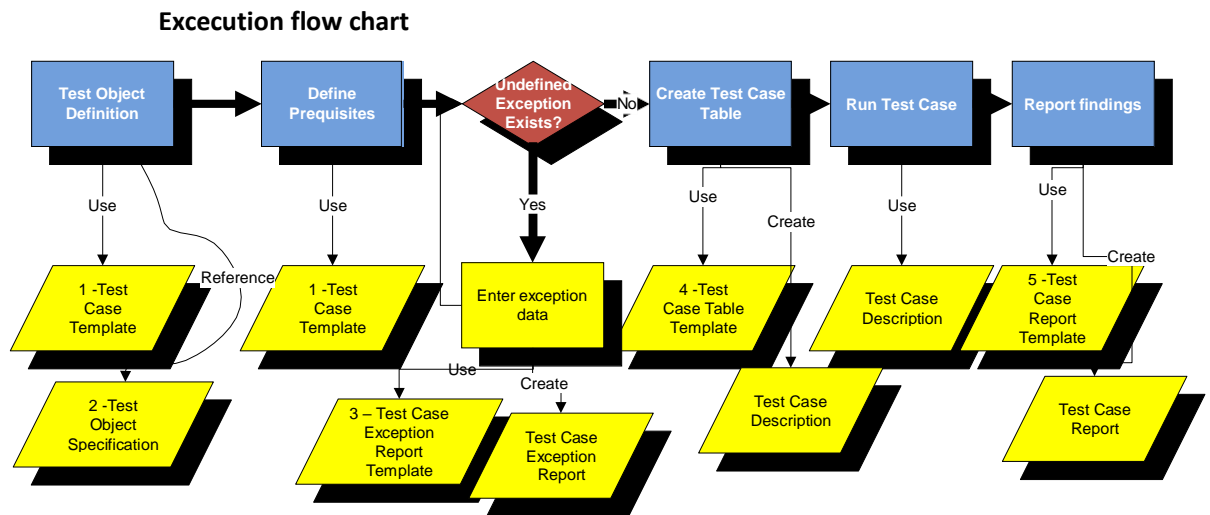


Fig 6.4: Execution Flow Chart

Legend

1. *BII WG4 Test Case Template.doc*. The **Test Case template** used to define and set up the **Test Case Description**.
2. The **test object specification** provides a reference to the object subject to test or if required, enter a copy of the object description excerpted from the object description for the test object. When referenced, the reference should include at least :

Testing is the process to uncover the errors.

Objectives of Testing: - There are following objectives:

- I. Testing is the process of executing the program to find error.
- II. A group test has a high probability of finding the errors.
- III. A successful test uncovers the all errors that have not been found.

Testing Principle: - There are following objectives:

- I. The test should be according to the customer's requirement.
- II. There should be a planning for testing before it starts.
- III. Poreto principle implies that 80 percent of all errors uncover during testing will likely be traceable to 20 percent of all program components.
- IV. Testing should begin 'in the small' and progress toward testing 'in the large'.

There are two types of testing: -

1. Black Box Testing
2. White Box Testing

Black Box Testing: - It is also called behavioral testing. The program is directly run by the computer to find the errors.

Objective of the Black Box Testing

- I. Incorrect or missing function.
- II. Interface error.
- III. Errors in data structures on database access.
- IV. Performance error.
- V. Initialization and termination error.

White Box Testing: - It is also called glass box testing. It traces all the paths of a program manually to find the errors.

Advantage of White Box Testing

- I. It guarantees that all independent paths have been checked at least once.
- II. It checks all logical decisions for true and false.
- III. Executes all loops at their boundary values.
- IV. Checks internal data structures.

Reasons for White Box Testing

- I. It can find logical errors, which cannot be found by 'black box'.
- II. We often believe that a logical path is not likely to be executed when, in fact, it may be executed on a regular basis.
- III. Typographical errors are random. The block box testing can find out typing error but typing error are in the program.

Unit Testing

It is a technique of testing individual module at a time. The important control paths are tested to find the errors within the boundary of the module.

The interface is tested to check that input and output the module are correct. The data structure is tested to check that the data flow from input to output is correct. Boundary conditions are tested to check that the module works correct at boundary. The independent paths are tested to check that each part is executed at least once then all error paths are checked.

Integration Test

It is the technique of testing after integrating the module. It finds the error related to the interface. There are two method of integration test:

- a) Top Down Test
- b) Bottom Up Test
- a) **Top down Test:** - The modules are integrated by moving down from top to bottom. The modules can be integrated by either using depth-first integrated or breadth-first integration. In DFS modules all integrated from top to down on individual path. For e.g. M1, M2, M5, & M8 INTEGRATED FIRST. In BFS integration is level by level. For e.g. Modules M1, M2, M3 and M4 all are integrated first.

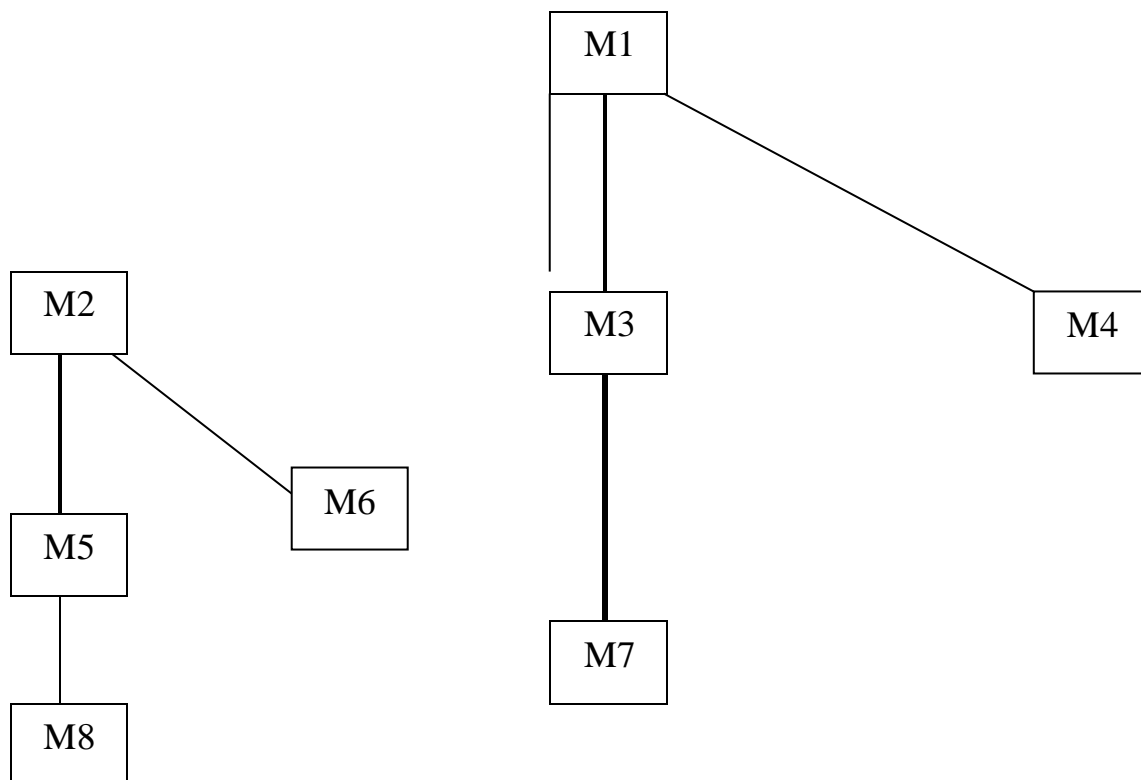


Fig 6.5:- Integration Testing

The steps of top-down test:-

- a) The main control module is used as a test driver and stubs are substituted for all components directly subordinate to the main control module.
- b) Depending on the integration approach selected subordinate stubs are replaced one at a time with actual components.
- c) Tests are conducted as each component is integrated.
- d) One completion of each set of tests, another stub is replaced with the real component.
- e) Regression testing may be conducted to ensure that new errors have not been introduced.

Bottom up Integration: - Integrates the modules from bottom to up. It has following steps:

- a) Low level components are combined into clusters.
- b) Drivers are developed for clusters.
- c) Cluster is tested.
- d) Drivers are removed and Cluster is combined moving upward.

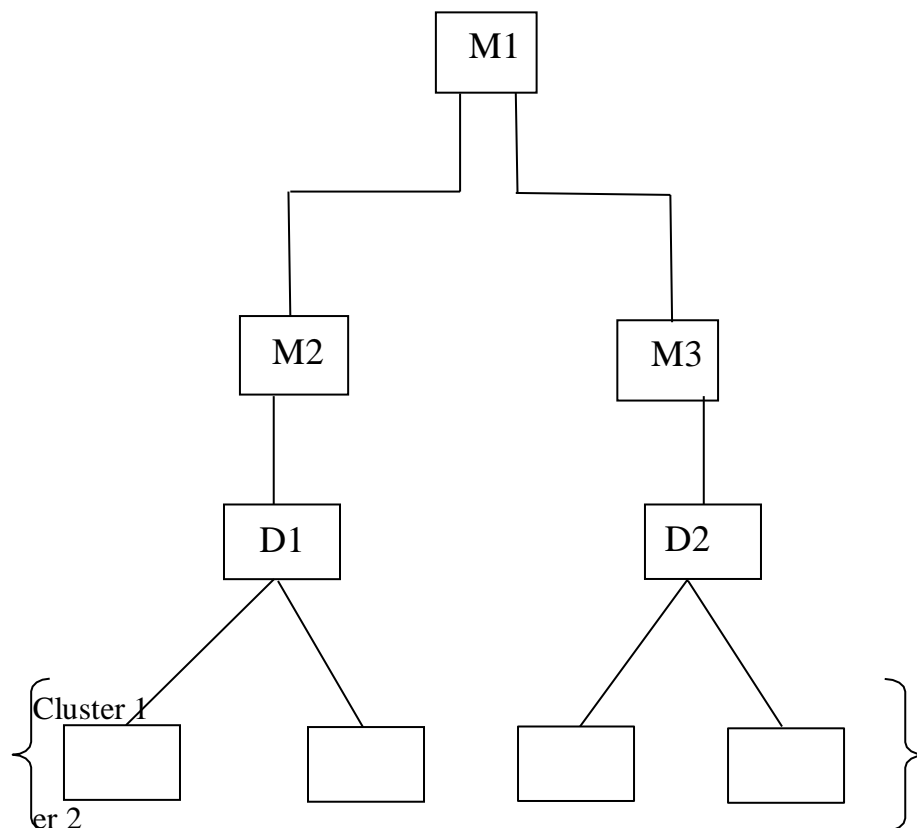


Fig 6.6:- Bottom-up Integration

System Testing

It is a series of different tests to test the overall system. Although each test has a different purpose, all work to verify those elements have been properly integrated and performed allocated functions. The type of system tests are following:

- a) **Recovery Test:** - It is performed to ensure that the data can be recovered and system can be restarted even if the system failure. It is a system test that forces the software to fail in a variety of ways and verified that recovery is performed properly. If recovery is automatic re-initialization, checkpoint mechanism data recovery and restart are evaluated for corrections. If recovery requires human intervention, the mean-time-to-repair is evaluated to determine whether it is within acceptable limits.
- b) **Security Test:** - It is performing to verify that protection mechanisms built into a system are perfect or not i.e., can it protect system from improper penetration attempt. The system should be tested for any type of for query attempt. During security testing the tester play the role of individual who desired to penetrate the system. The tester may attempt to get password, may attack the system to break down the security may purposely cause system errors, may browse through unsecure data.

The good security testing will penetrate the system and break the security. So, the role of the system designer is to make penetration cost more than the value that will be obtained by breaking the system security.

- c) **Stress Test:** - It is performed to test the abnormal situation i.e. how high we can crank the system before it fails. Stress testing executes a system in a manner that demands resources in abnormal quantity, frequency or volume. For example
 - i. Special test may be design that generates ten interrupts per second, when one or two is the average rate.
 - ii. Input data rates may be increased by an order of magnitude to determine how input function will respond.
 - iii. Test cases that require maximum memory or other resource are executed.
 - iv. Test cases that may cause thrashing in a virtual operating system are designed.
 - v. Test cases that may cause excessive hunting for disk resident data are created. Essentially, the tester attempts to break the program.

- d) **Performance Test:** - It is used to test the run time performance of the system. It occurs throughout all the steps in the testing process. The performance of an individual module may be checked using white box method. It is necessary to measure the resource utilization.

Test case

A test case has components that describe an input, action or event and an expected response, to determine if a feature of an application is working correctly.”

A test case is also defined as a sequence of steps to test the correct behavior of a functionality/feature of an application

The characteristics of a test case are that there is a *known input* and an *expected output*, which is worked out *before* the test. The known input should test a pre- condition and the expected output should test a post-condition

How to write test cases?

Here is a simple test case format

Fields in test cases:

Test case id:

Unit to test: What to be verified?

Assumptions:

Test data: Variables and their values

Steps to be executed:Expected result: Actual result: Pass/Fail: Comments:

e.g.

Test case id 101Field for Test:- Login ID PASSWORD

Assumption:-

User chooses any name for

login id.Password

Not visible

6.4.1 Debugging System

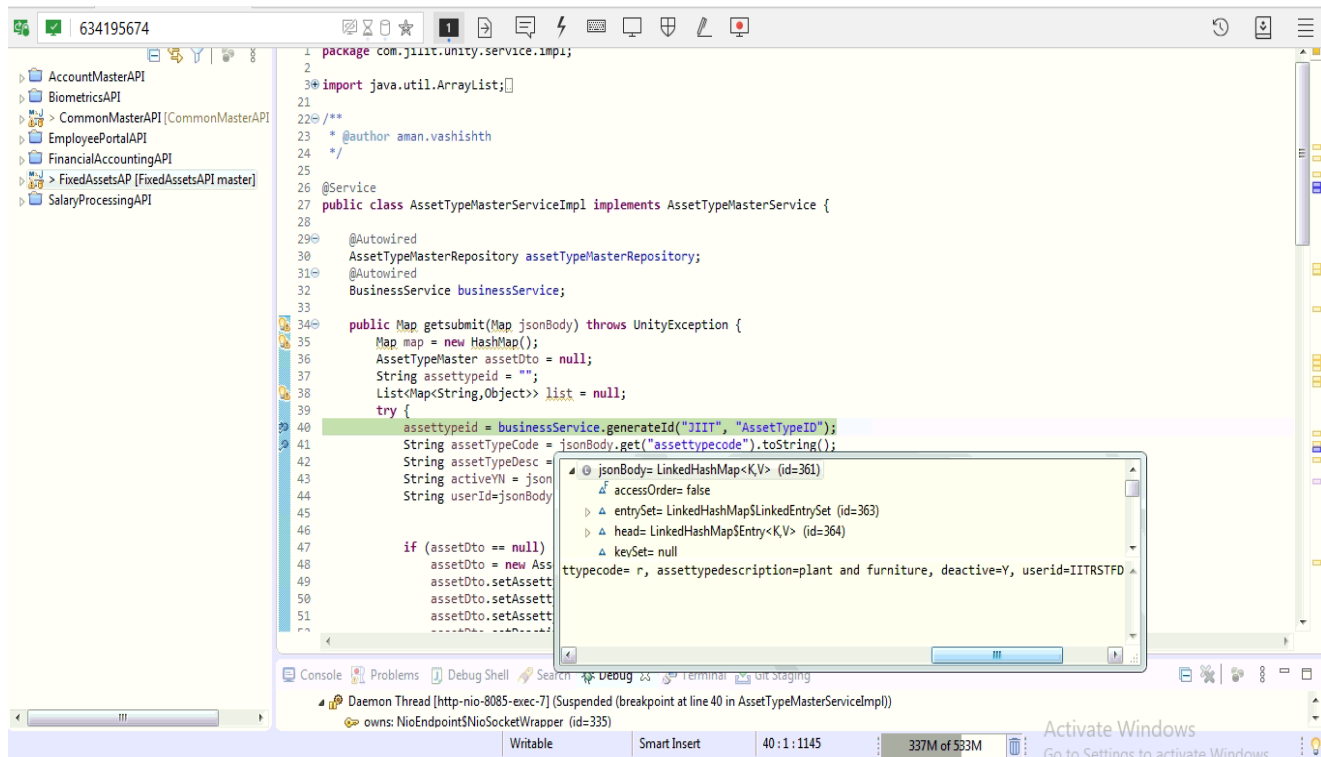


Fig 6.7:- Debugging System

This Data Shown in Popup, refers to that the data entered by the user from User Interface is been mapped to the backend code and the data is sent to the backend through `jsonBody` from which the data is saved in the database.

Debugging and Code Improvement

In ideal worlds, all programmers would be so skilled and attentive to detail that they would write bug-free code. Unfortunately, we do not live in an ideal world. As such, debugging, or tracking down the source of errors and erroneous result, is an important task that all developers need to perform before they allow end-user to use their applications. We will discuss some techniques for reducing the number of bugs in code up front.

There are three categories of bugs

Syntax error:

These errors occur when code breaks the rule of the language, such as visual Basic sub statement without a closing End sub, or a forgotten closing curly braces ({}) in c#. Theses error the easiest to locate. The language complier **or** integrated development environment (IDE) will alert you to them and will not allow you to compile your program until you correct them.

Semantic error:

These errors occur in code that is correct according to rules of the compiler, but that causes unexpected problems such as crashes or hanging on execution. A good example is code that execute in a loop but never exists the loop, either because the loop depends on the variable whose values was expected to be something different than it actually was or because the programmer forget to increment the loop counter. Another category of errors in this area includes requesting a field from a dataset, there is no way to tell if the field actually exists at compile time. These bugs are harder to detect and are one type of running error.

Logic error:

Logic errors are like semantic errors, logic errors are runtime error. That is, they occur while the program is running. But unlike semantic errors, logic errors do not cause the application to crash or hang. Logic error results in unexpected values or output. This can be a result of something as simple as a mistyped variables name that happens to match another declared variable in the program. This type of error can be extremely difficult to track down to eliminate.

Preventing Debug Write readable code:

Develop and make consistent use of naming and coding standards. It not that important which standard we use, such as Hungarian notation or Pascal, Casing (First Name) or other naming conventions, as long as we use one. We should also strive for consistency in our comments and encourage liberal commenting code.

Create effective test plan:

The only effective way to eliminate logic error is to test very path of your application with every possible data values that a user could enter.

This is difficult to manage without effective planning. We should create our test plan at the same time we are designing the application, and we should update these plans as you modify the application design.

Code Improvement:

We make the Class “Data context” in our project which is useful for reducing the code redundancy and make code consistency. “Data context” function improves the code, when by using the Class “Data context” we can create a connection with database, open

the database, close the database, dispose the database, through the “Data context” Class can access the data table. In our project we can create the object of the “Data context” then after creating the object we do not need to write the functions, such as database connectivity, open connection, close connection, dispose connection, get data table connection. “Data context” Class makes the code improvement in our project. Through the “Data context” function we can insert, delete, update the records and show the data tables and check the database so can say that the “Data context” function is more useful and make code improve.

CHAPTER 7

LIMITATION AND FUTURE SCOPE

7.1 Limitation

1. Without having UserId and Password no one can access the Project.
2. Only those who have authority access this Project.
3. It is used only in an organization (not able to access everywhere).

7.2 FUTURE SCOPE

- 1) We can add new features as and when require.
- 2) This project is reliable and can be used always.
- 3) This Project helps most of the Organization in future.
- 4) It will become most efficient and effective Project for the Organizations.
- 5) The Future Scope of this Project will never finish.

REFERENCES:-

1. International Electrotechnical Commission, "IEC 61968-1 Application integration at electric utilities – System Interfaces for distribution management Part 1: Interface Architecture and General Requirements", IEC Reference number IEC 61968-1:2003(E).
2. Shahidehpour, M., Ferrero, R. (2005), "Time Management for Assets" Chronological Strategies for Power System Asset Management" In IEEE Power & Energy Magazine May/June 2005.
3. Bertling, L., Allan, R., Eriksson, R. (2005), "A Reliability-Centered Asset Maintenance Method for Assessing the Impact of Maintenance in Power Distribution Systems" IEEE Transactions on Power Systems, Vol. 20, No. 1, February 2005.
4. Goodfellow, J.W., "Applying Reliability Centered Maintenance (RCM) to Overhead Electric Utility Distribution Systems" In proceedings of Power Engineering Society Summer Meeting, 2000, 16-20 July 2000.
5. Brown, R.E., Humphrey, B.G (2005) "Asset Management for Transmission and Distribution" IEEE Power & Energy Vol 3 No: 3 May/June 2005.
6. Gammelgård M., Närman P., Ekstedt M., Nordström L., Business Value Evaluation of IT Systems: Developing a Functional Reference Model In Proceedings at the Conference on Systems Engineering Research April 2006.
7. L. Nordström, T. Cegrell "Extended UML Modeling for Risk Management of Utility Information System Integration" In Proceedings of the IEEE Power Engineering Society's General Meeting, San Francisco, USA, June 12-17, 2005.
8. L. Nordström, T. Cegrell "Analyzing Utility Information Systems Architecture using the Common Information Model" In Proceedings of 2nd CIGRE / IEEE PES International Symposium on Congestion Management in a Market Environment, San Antonio, USA, October 5-7, 2005