

Process Flow Diagram, Use Case Diagram, ER Diagram, Application Architecture, Development Process Document of Budget Management System Software For Dhaka WASA



ing is intelligence



Submitted By,
Coder De Dhaka Ltd.
House No.45/A, Dilu Road, New Eskaton, Dhaka 1217.
Web: <https://coderdedhaka.com>,
Email: coderdedhaka@gmail.com
Contact: +(88) 01516197588

Budget Management System Software for DWASA

TABLE OF CONTENTS

Content	Page
1. Introduction.....	03
2. Process Flow Diagram.....	03
3. Use Case Diagram.....	08
4. Development Process.....	09
5. Application Architecture.....	15
6. Database Table Description.....	26
7. ERD.....	45
8. Conclusion.....	46



Introduction:

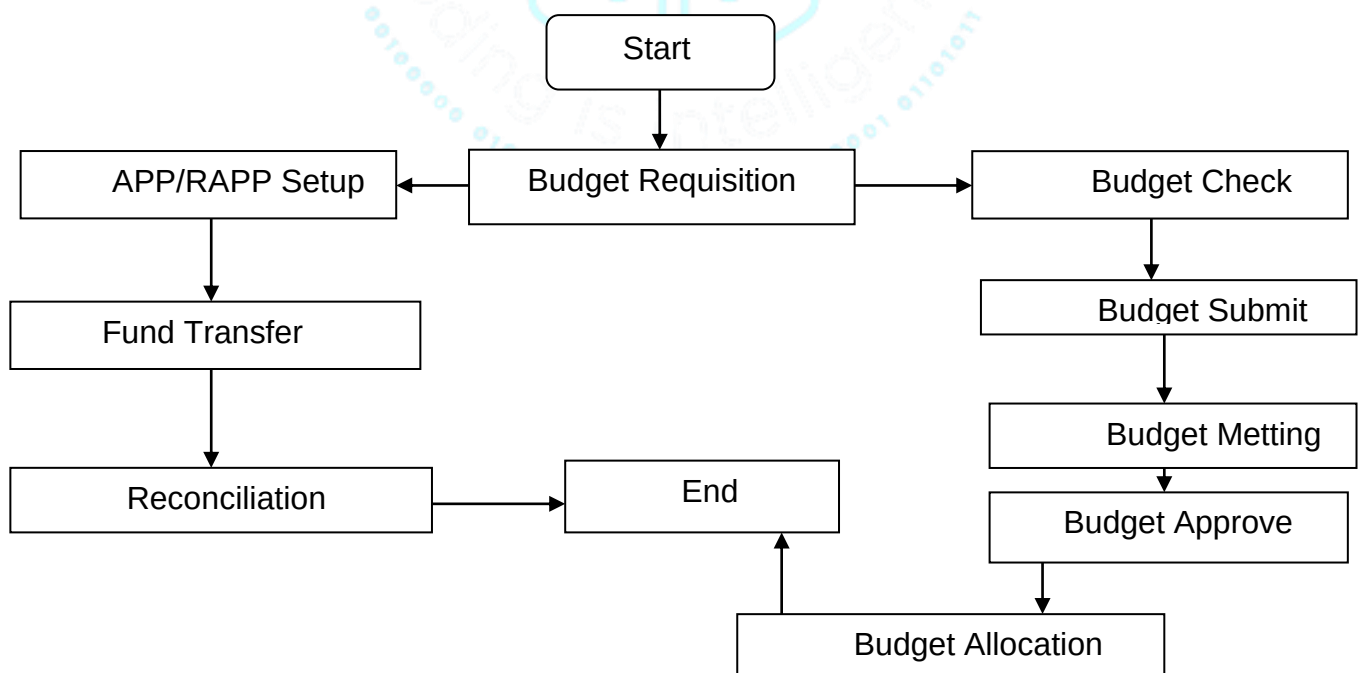
Dhaka WASA Budget Management System Software, an innovative solution designed to streamline and enhance the budget management processes of Dhaka Water Supply and Sewerage Authority (WASA). With the ever-increasing demand for efficient water supply and sewerage services in Dhaka, it is crucial for Dhaka WASA to effectively manage its budget to ensure sustainable development and meet the needs of its residents. The Dhaka WASA Budget Management System Software provides a comprehensive and user-friendly platform that enables Dhaka WASA to efficiently plan, allocate, monitor, and control its financial resources. This software leverages cutting-edge technology and best practices in budget management to empower Dhaka WASA with accurate and real-time financial information, enabling informed decision-making and effective resource utilization.

The Dhaka WASA Budget Management System Software is a powerful tool that revolutionizes the budget management processes of Dhaka WASA. By leveraging technology and automation, it enables Dhaka WASA to effectively allocate resources, monitor expenditures, and make informed decisions to meet the water supply and sewerage needs of the city. With this software, Dhaka WASA is well-equipped to ensure financial sustainability and deliver reliable services to the residents of Dhaka.

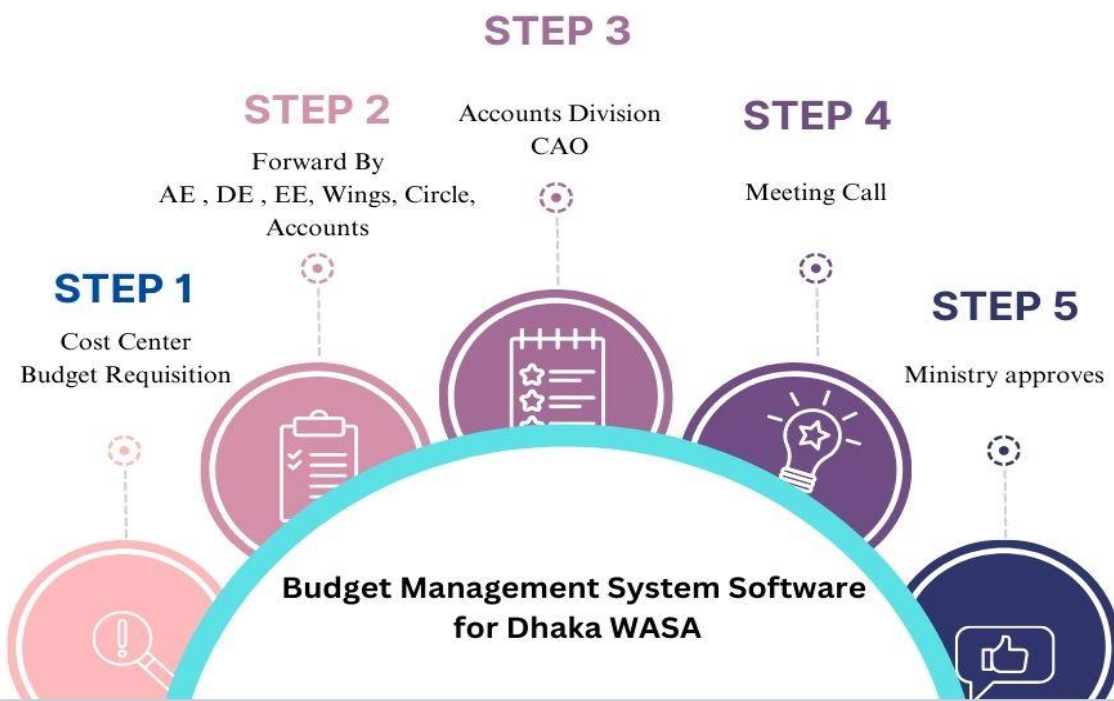
Process Flow Diagram:

A **Process Flow Diagram (PFD)** is a type of flowchart that illustrates the general flow of processes within a system. It typically shows the major steps in a process, how they are connected, and the inputs/outputs at each stage.

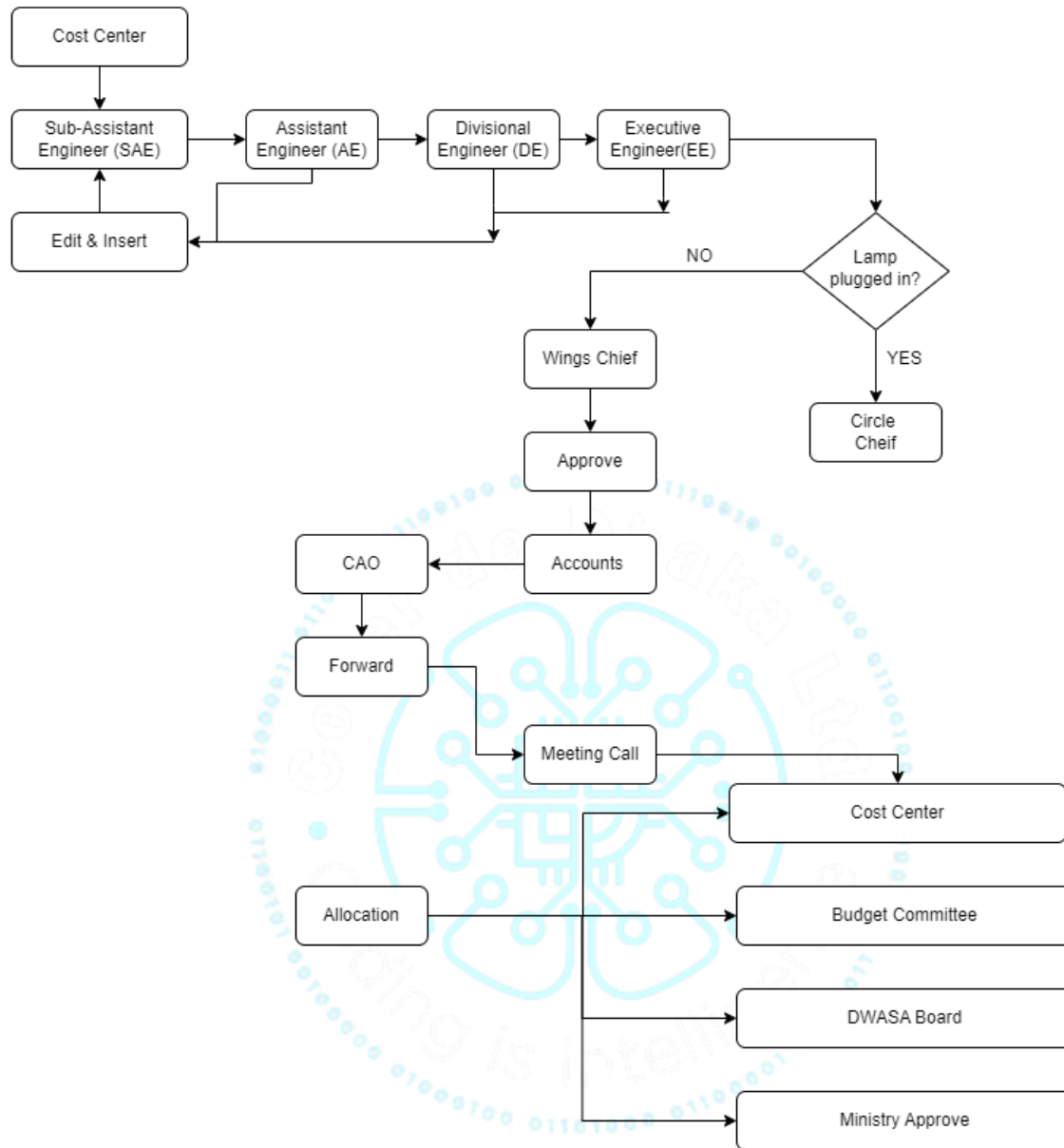
Division Part:



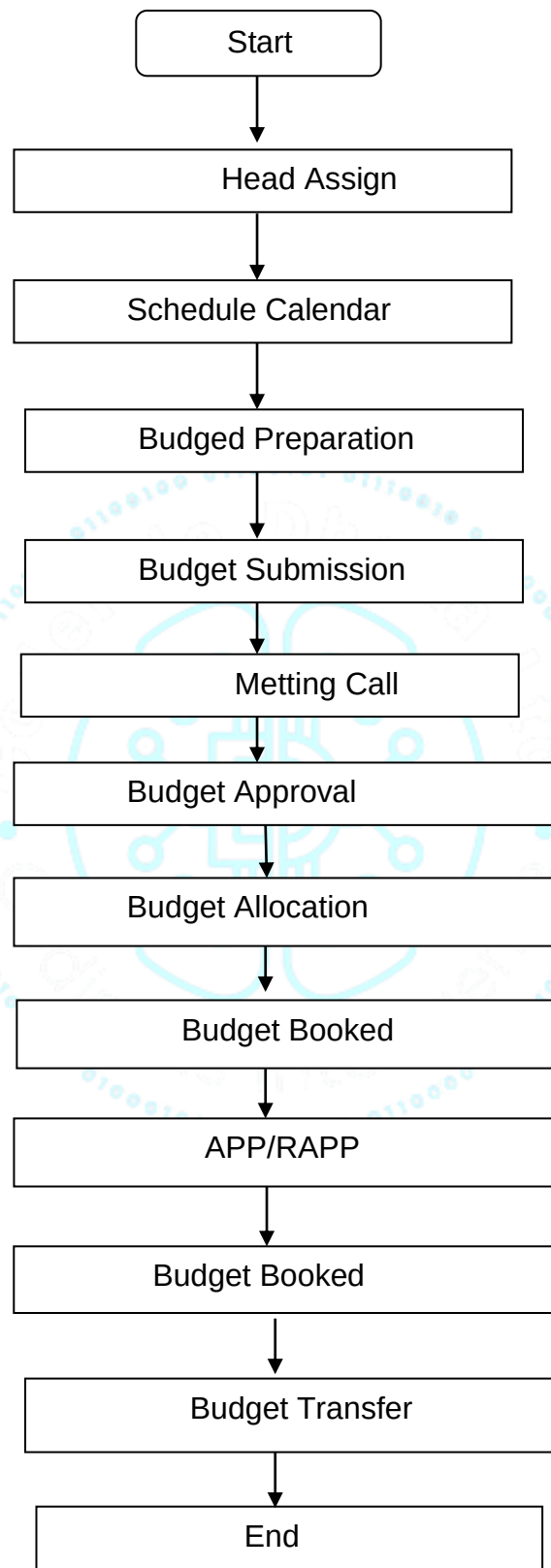
Process Diagram



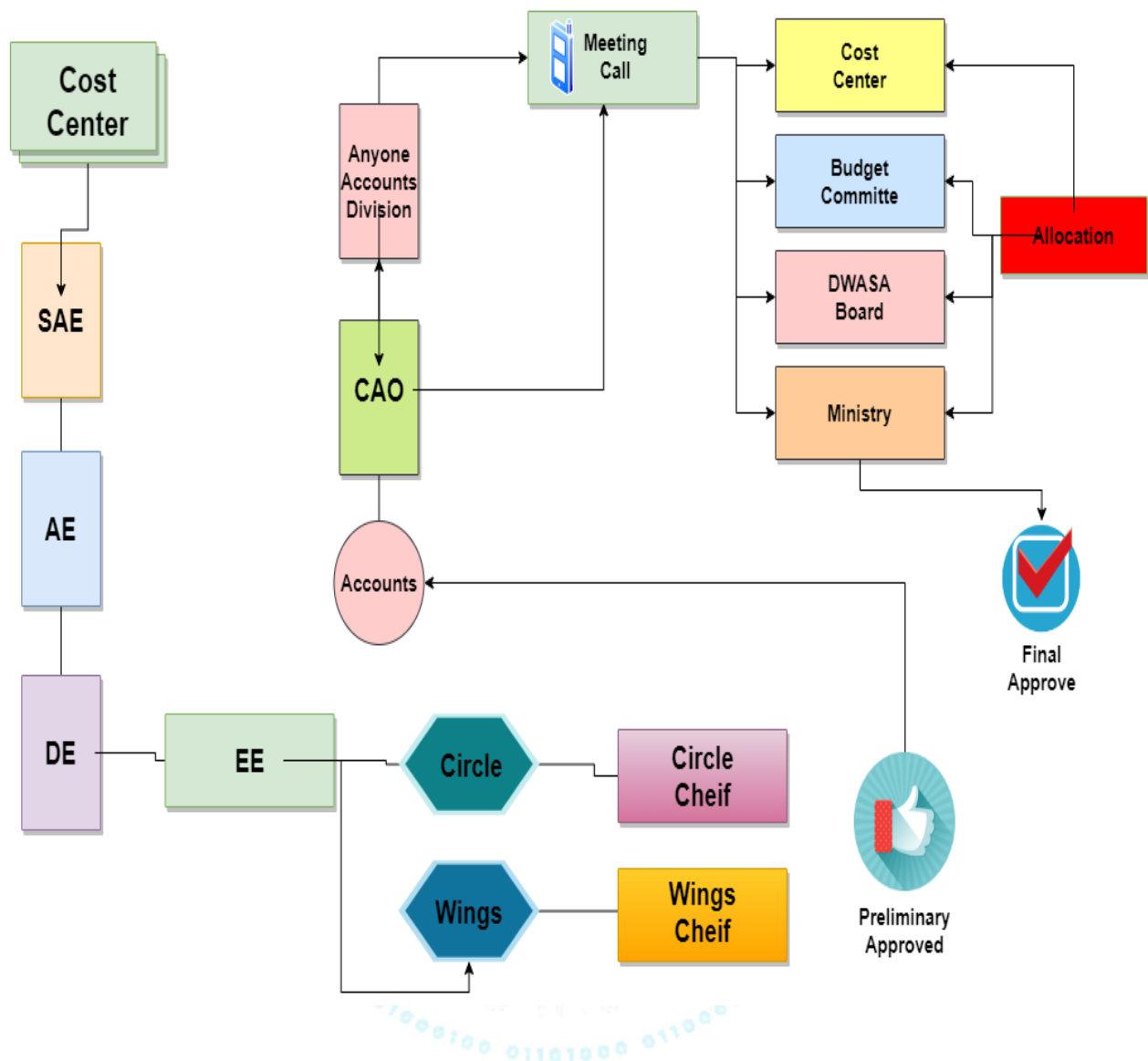
Workflow and Data Flow Design



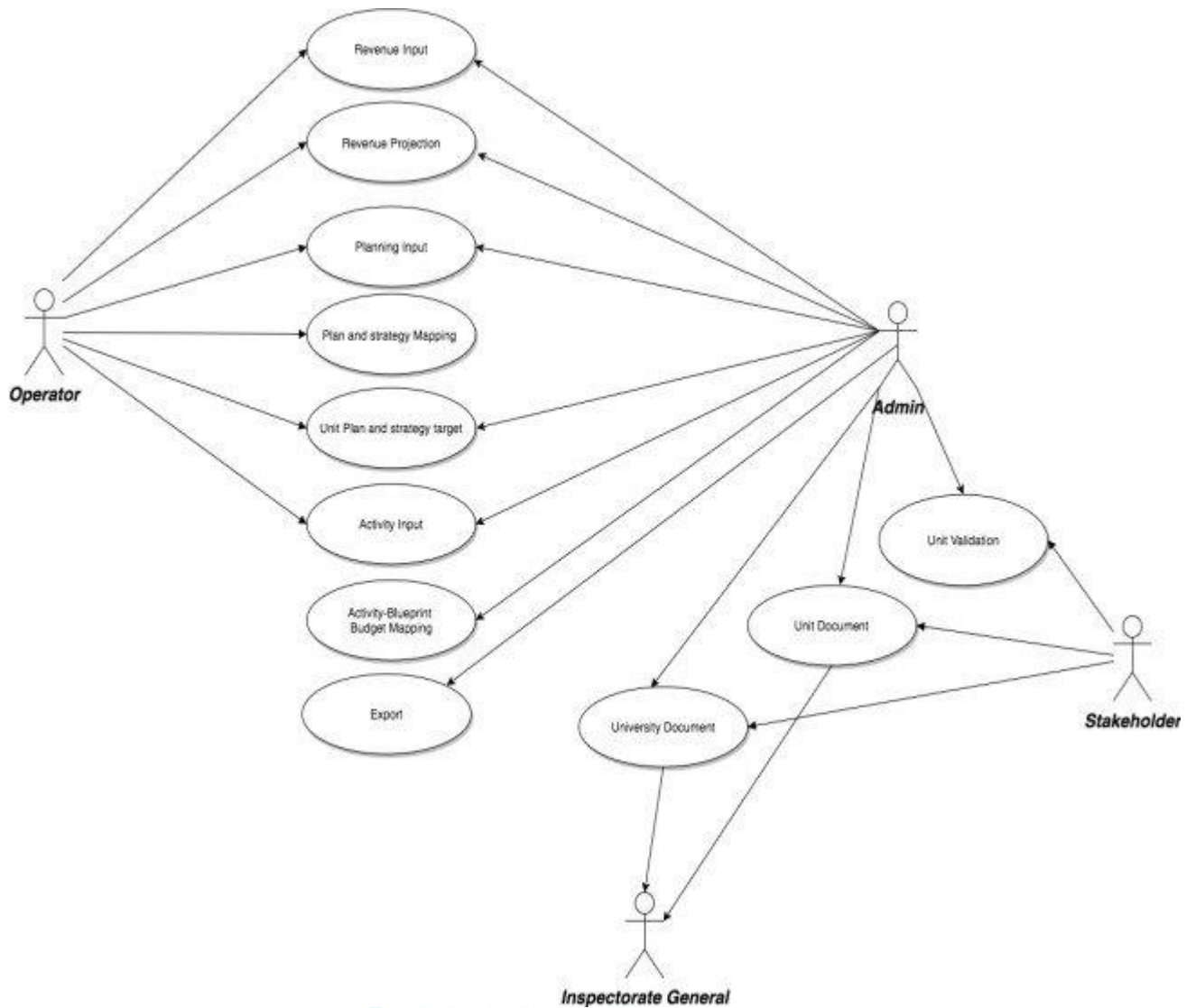
Division Part:

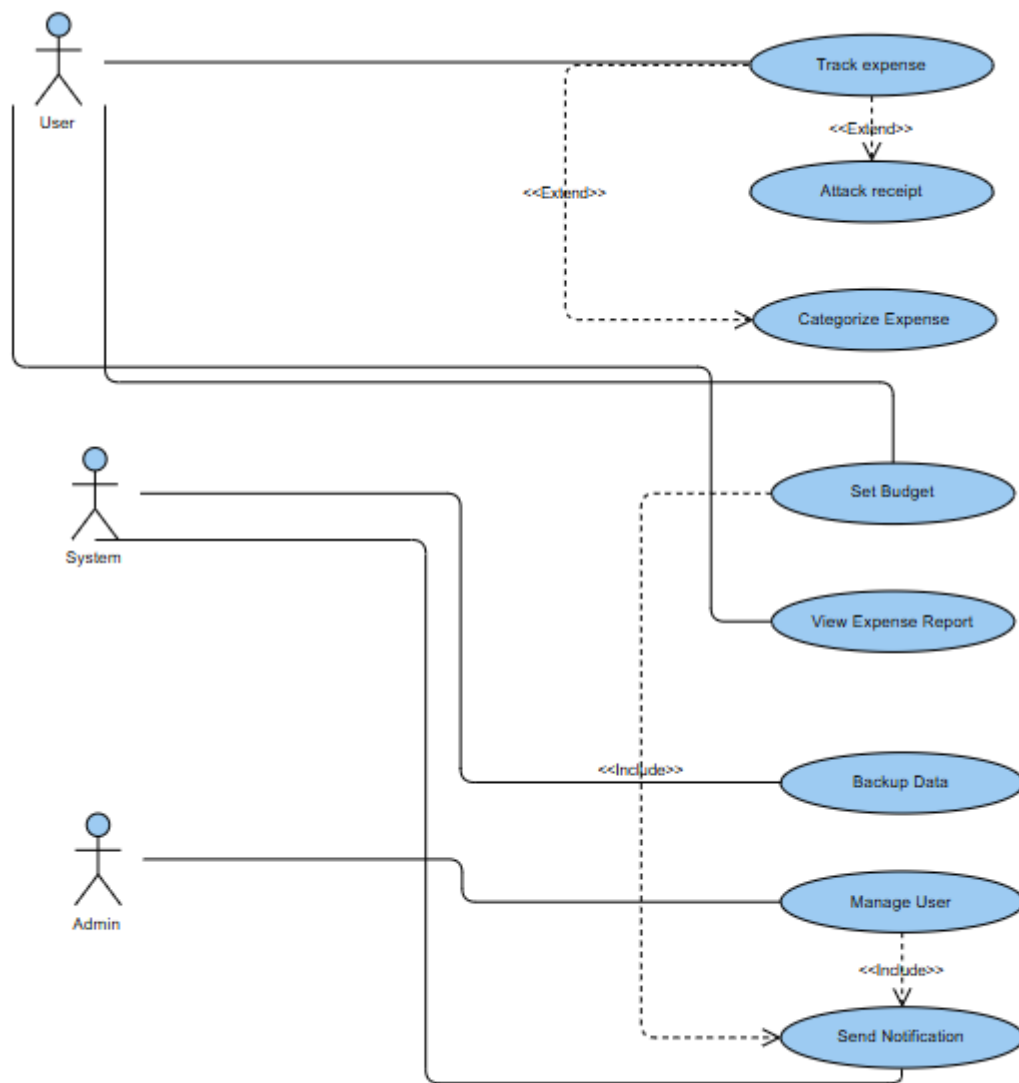


User Interface Design



Use Case Diagram





Development Process

The Software Development Flowchart outlines the steps involved in developing software. These steps include identifying the problem or need, defining requirements, designing the solution, implementing the code, fixing bugs, and testing the code. By following a standardized **software development process**, businesses can ensure that the software meets the needs of the users, is reliable and efficient, and is aligned with the overall goals of the project.

A well-managed software development project can help businesses reduce costs, improve productivity, and increase customer satisfaction. By having a clear and documented process, businesses can minimize errors and risks, improve the accuracy of project estimates, timelines, and budgets, and ensure that everyone involved in the project understands their **roles and responsibilities**. This can result in a smoother and more efficient development process.

Furthermore, a well-managed software development project can help businesses stay competitive by delivering software that meets the needs of the users and is up-to-date with the **latest technologies and**

trends. By keeping pace with the industry, businesses can ensure that their software remains relevant, engaging, and useful to their customers, ultimately contributing to the long-term success of the business. Overall, the Software Development Flowchart provides a framework for businesses to develop software that meets the needs of the users, is reliable and efficient, and contributes to the **long-term success** of the business. By following a standardized software development process, businesses can reduce costs, improve productivity, increase customer satisfaction, and stay competitive in the industry.

Benefits of creating this flowchart

Creating a Software Development Flowchart can bring several benefits to businesses. Firstly, it provides a clear and concise representation of the steps involved in developing software. By visualizing the **software development process**, businesses can better understand how it works and identify areas for improvement. This can help businesses optimize their software development process for efficiency, reduce errors, and improve consistency.

Secondly, a Software Development Flowchart can help businesses improve their software development process by providing a standardized process for identifying problems or needs, defining requirements, designing the solution, implementing the code, fixing bugs, and testing the code. By following a well-defined process, businesses can ensure that the software meets the needs of the users, is reliable and efficient, and is aligned with the overall goals of the project. This can result in improved **customer satisfaction**, reduced costs, and increased business success.

Overall, creating a Software Development Flowchart can help businesses improve their **software development process**, reduce errors, save time and money, and improve their bottom line. By using flowcharts to visualize the software development process, businesses can optimize their software development process and improve their overall business success. Additionally, a well-managed software development project can result in increased customer satisfaction, improved retention rates, and increased profitability, ultimately contributing to the long-term success of the business.

Initial request & feasibility study

The Initial request & feasibility study establishes a high-level view of the intended project and determines its goals.

Deliverables:

- Feasibility Report i.e. Cost benefits analysis and total time frame.

Requirement Analysis & Specification

The Requirement Analysis & Specification refines project goals into defined functions and operation of the intended application after analyzing end-user's information needs.

Deliverables:

- Requirement report.

Systems Design

This phase describes desired features and operations in detail, including screen layouts, business rules, process diagrams, pseudo code and other documentation. It consists of two phases. Those are:

➤ *Logical System Design*

At this stage the logical view of the system is specified by system architecture in terms of components and their interaction.

➤ *Physical System Design:*

In this step logical system design will be transformed into Physical design.

Deliverables:

- Data Dictionary,
- Forms Layout,
- Report Format and
- Menu Structure.

System Development (code & unit test)

In this stage, the development of the user procedures and the interfaces is carried out. Ongoing and throughout development, and also as a separate stage after development, a thorough Testing is carried out. The real code is written in this stage.

Deliverables:

- Complete system specification and documentation (system documentation),
- Forms, Reports and Menu executables.

System Integration and testing (Implementations)

Once the programs have been written and unit tested, Integration, Testing of all programs running as a complete system is carried out. This stage brings all the pieces together into a special testing environment and then checks for errors, bugs and interoperability.

Deliverables:

- Integration Report.

System Acceptance Test:

This is the final stage of initial development, where the software is put into production and runs actual business.

Deliverables:

- User's sign off document.

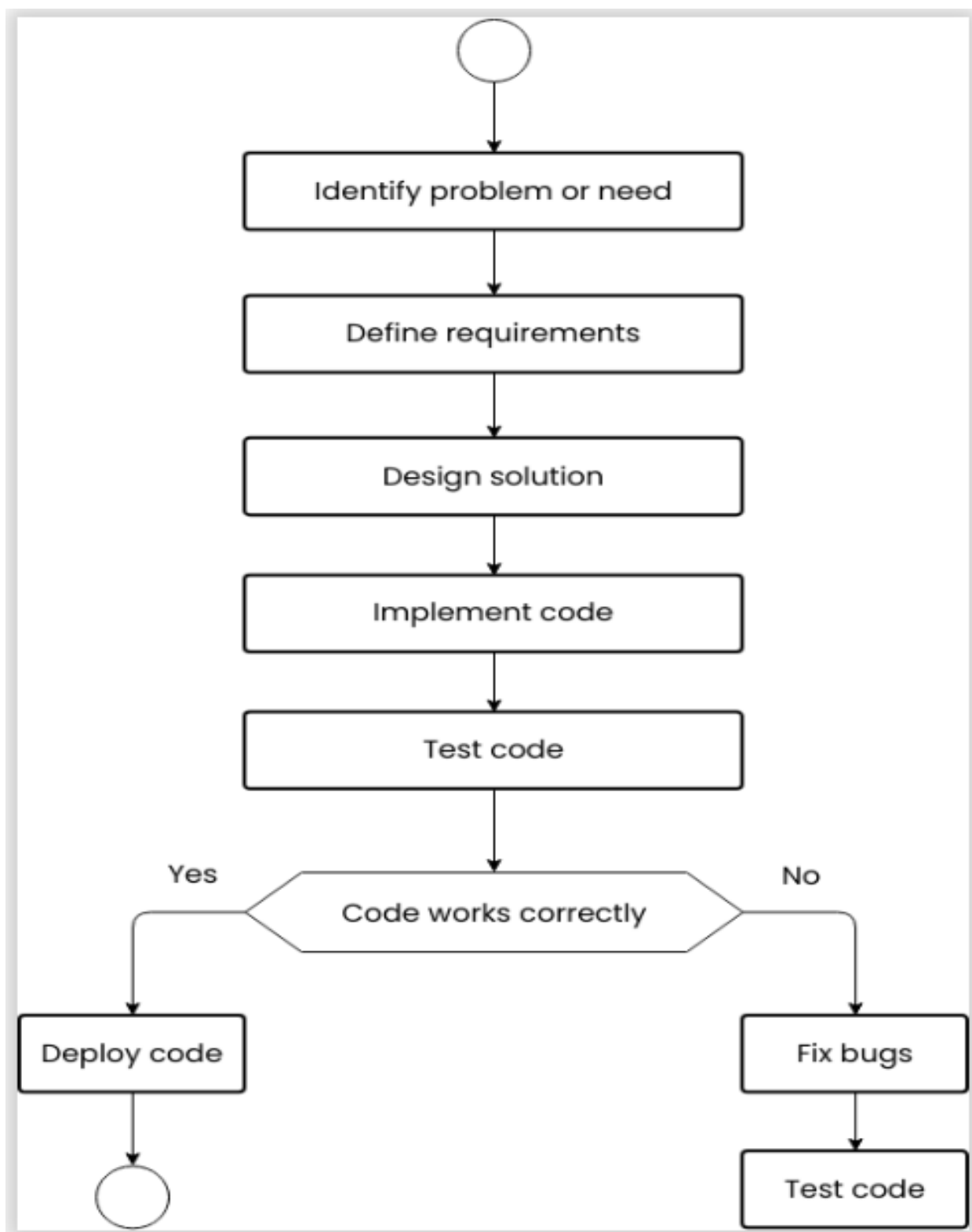
Maintenance

The maintenance activities are those activities that are required after an application has been successfully delivered into production. This includes changes, correction, additions, move to a different computing platform and more. This, the least glamorous and perhaps most important step of all, goes on seemingly forever.

- Reports: Contains the local reports that would be used to export to PDF and CSV format.

There are some uses report tools are:

1. Bi Publisher
 - a. Report Queries
 - b. Report Layouts (.rtf)
2. Dynamic Content Reports
3. Classic Report
4. Interactive Reports
5. Chart
6. Interactive Grid



Development Flow chart

SL	Tools/Technology	Features
1.	Software Development Mythology	Agile Scrum
2.	Operating System	Windows/Linux
3.	Architecture	Three Tier Base
4.	Backend Technology	Oracle
5.	Frontend Technology	Oracle Apex 21.2
6.	Database	Oracle 19c
7.	Web Server	Apache Tomcat or Jetty
8.	API	REST
9.	Load Balancer	Nginx

System Environment

Design Patterns:

This application is designed as an object-oriented system for an Internet-based architecture using four-layer architecture by factoring application classes into the following layers:

The Presentation layers. This is the layer where the physical window and widget objects live. It will also contain Database driven. Any new user interface widgets developed for this application are put in this layer. In most cases this layer is completely developed within Database.

The Domain Mode. Most objects identified in the OO analysis and design will reside. To a great extent, the objects in this layer can be application-independent. Generic objects may be used in this application to reap the benefits of Object-Oriented programming.

The three layers. The application model has been mapped using Apex. This will map Database classes onto a wider range of legacy systems and will be more flexible in the creation of your database schema.

The Data layer. The data is managed by Oracle Database.

Exception Handling:

A good way to track exceptions is to implement an exception handler at the application level. This will allow consolidation of the logging and notification parts of the exception handling in one convenient place.

The global exception handler can handle both specific exceptions that UUIS trap in the code as well as generic un-handled exceptions. After global exception handler has done its work, it would redirect the users of the application to a friendly error page that tells them that something has gone wrong and instructions on what to do.

All unhandled exceptions would be logged in the Windows Application Event Log or log files with some detail on the nature of the failure.

general search

The general search included that retrieve records from the database according to user specified search criteria. Further, the search may encompass other information collections like on-screen data.

Users & Permissions

This application can assign, remove and editing the user permission. There are four levels of user from the access permission respective:

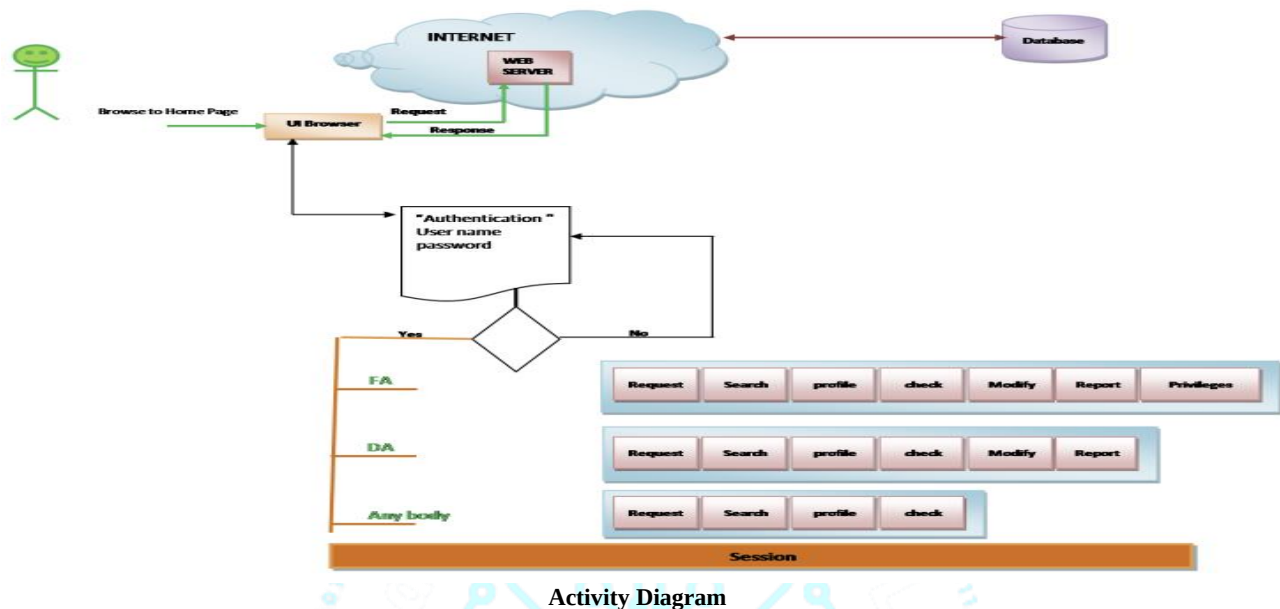
- 1) Level 1-the contractor and general user, who are only able to make request and view their own requests;
- 2) Level 2-Division user who are able to manage the requests, view and search assets in their own department;
- 3) Level 3- Accounts department who are able to manage the requests, view search assets in their own

faculty;

4) Level 4- administrators who are able to manage the requests, view audit log and search assets in the whole application.

Audit log

This Application can list all the user activities in the web application.



Activity Diagram

Performance Requirement

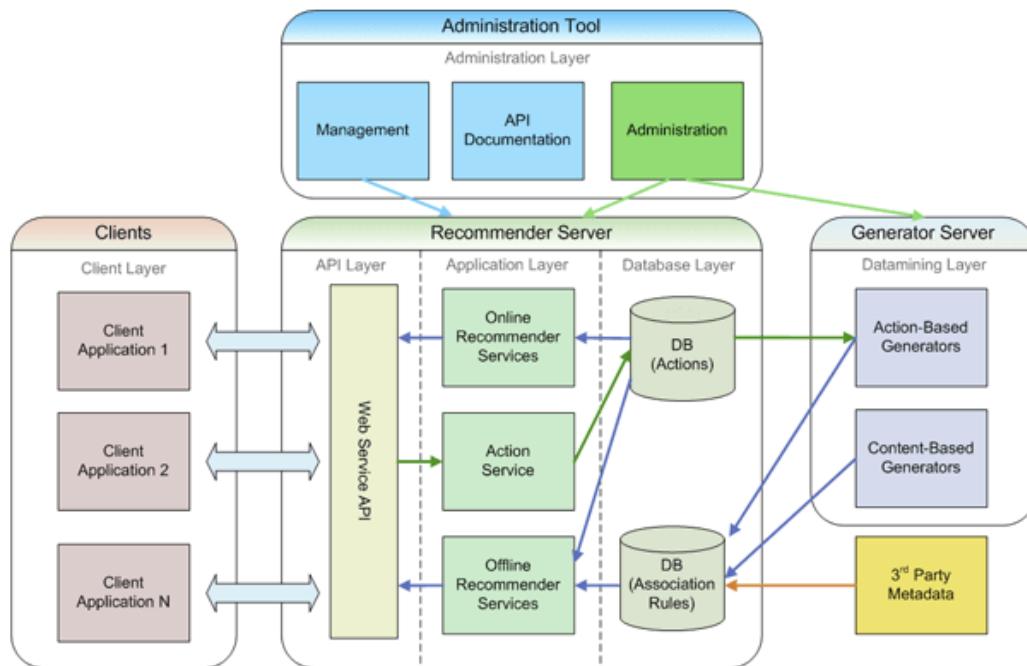
Performance should not be an issue because all of our server queries involve small pieces of data. Changing screens will require very little computation and thus will occur very quickly. Server updates should only take a few seconds as long as the phone can maintain a steady Signal. The cost-division algorithms used by in application will be highly efficient, taking only a fraction of a second to compute.

No	Requirement Text / Data	Acceptability	Limitations
1	All report loading from server response time should me limit	Avg. 10 to 15 Sec	For a big data need high band with
2	Blob data process	Avg. 30 to 60 Sec	Need high band with

Security Requirements

No	Requirement Text / Data	Acceptability
1	The system shall only allow authorized users to access the login, lost password, and registration Page.	Page Security
2	The system shall require a login and password to gain access to the application	Page Security
3	The system will permit role-based access into entire application	Role based access

Application Architecture



1. Oracle Apex Definition:

Oracle Apex is a Database-centric web application development framework.



Develop desktop and mobile web apps

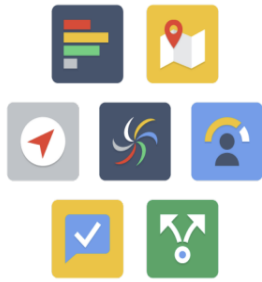


Visualize and maintain database data



Leverage SQL Skills and database capabilities

2. Oracle Apex Use Case:



Developing opportunistic
& self service web apps



Extending enterprise
application solutions



Migrating file based and
client server apps to the web

3. Distinguishing Characteristics:



App Development IDE is
a web browser.
No client software
needed



App definitions are stored in
the database as meta data.
Declarative – No code
generation

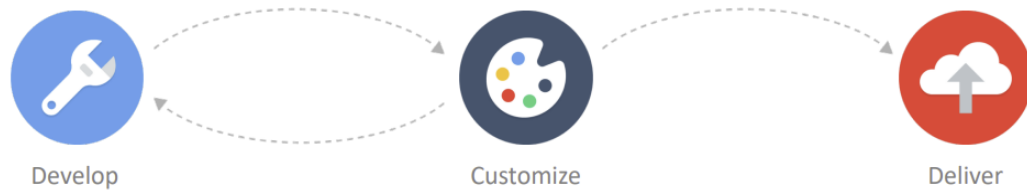


Page generation is efficient
with only one request and
one response.
Data processing done in the
Database

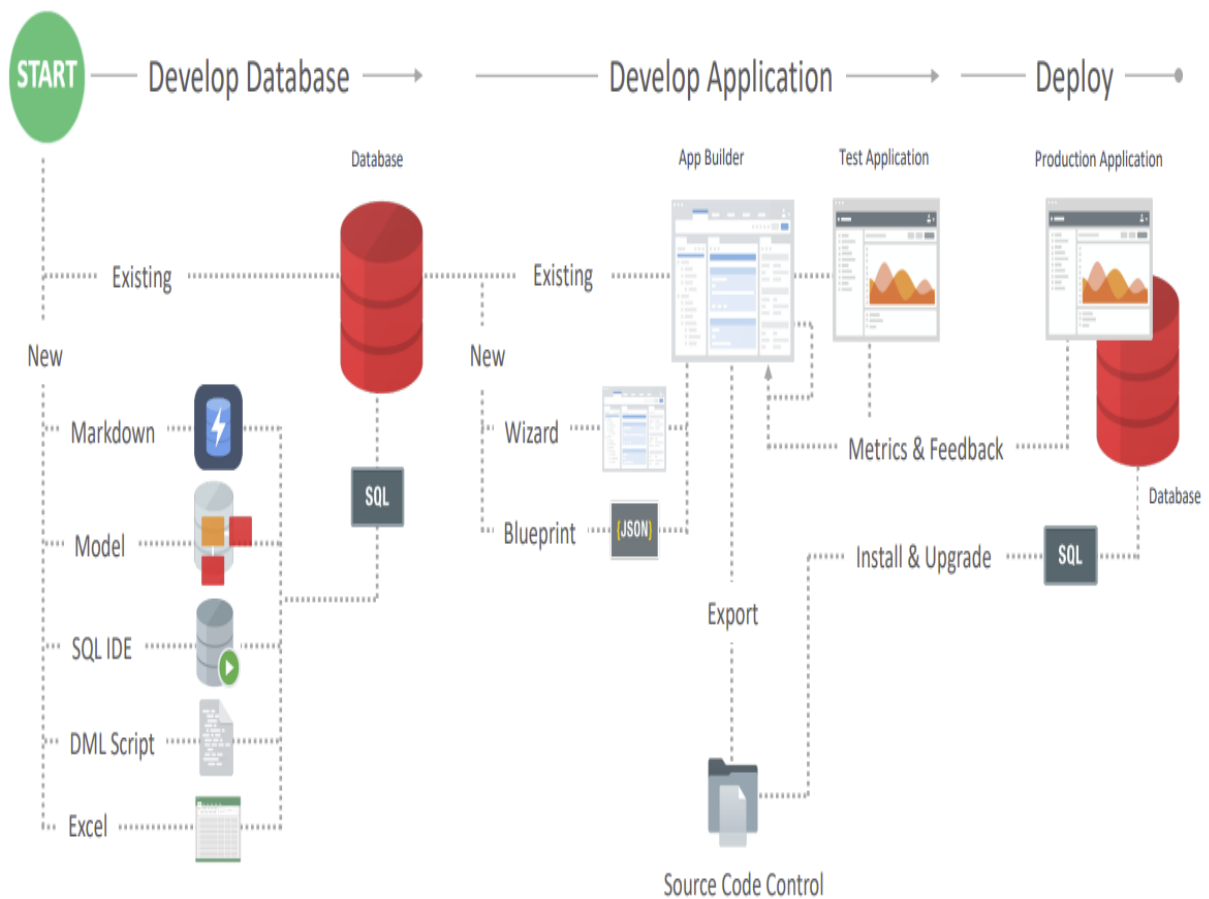
4. Rapidly develop, customize, and deliver:

Pre-built controls for security, authentication,
database interaction, validation, session management
and more...

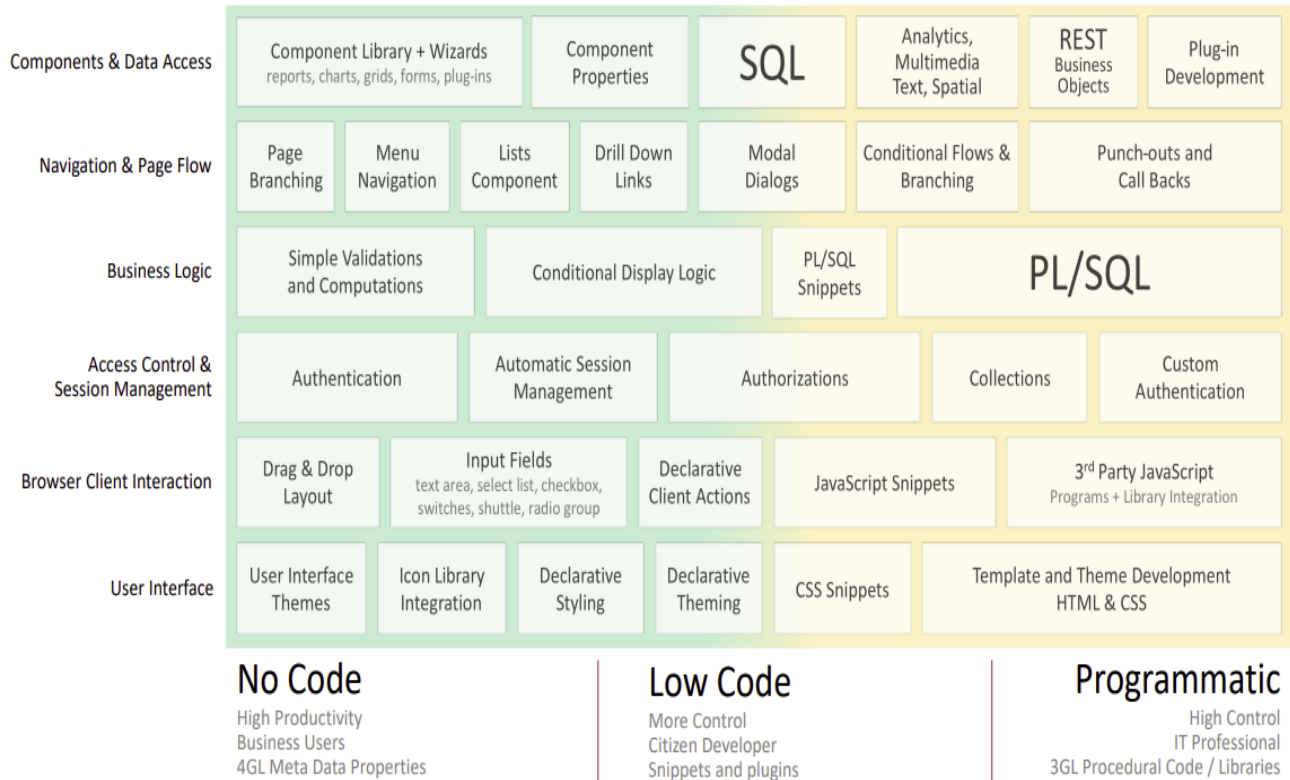
Go from prototype to production in minutes



5. Application Development and Deploy Process:



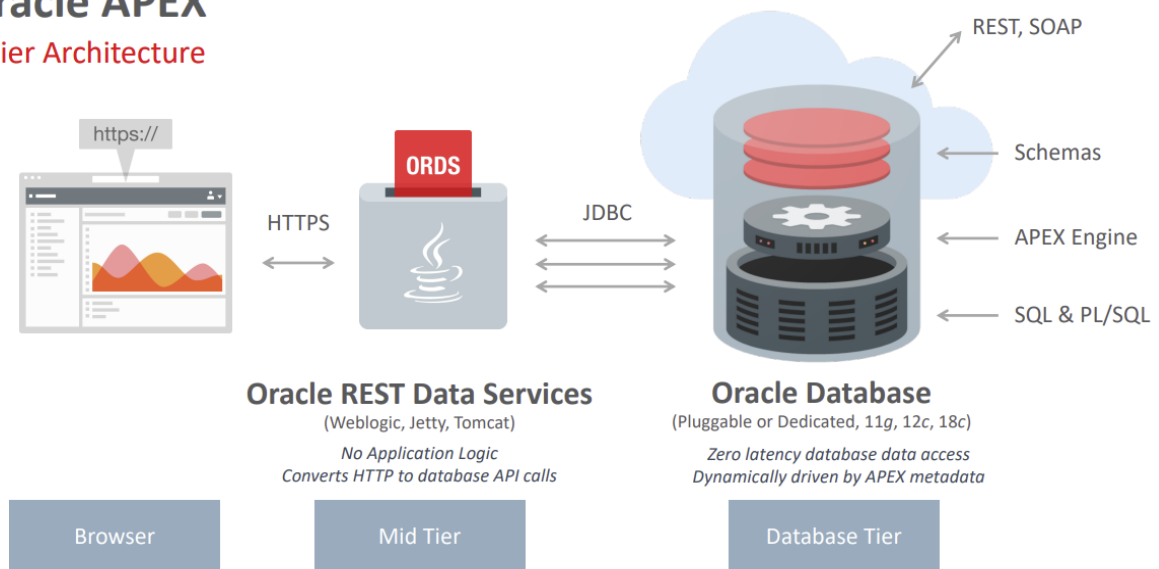
6. High productivity AppDev components on the no code to highly programmatic spectrum:



7. Tier Architecture:

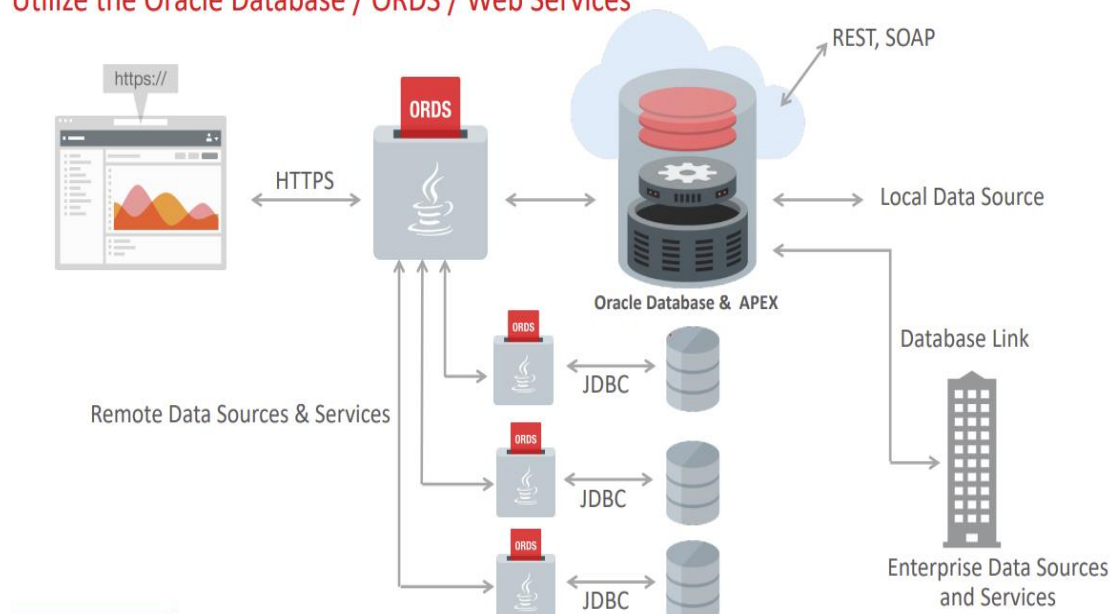
Oracle APEX

3 Tier Architecture

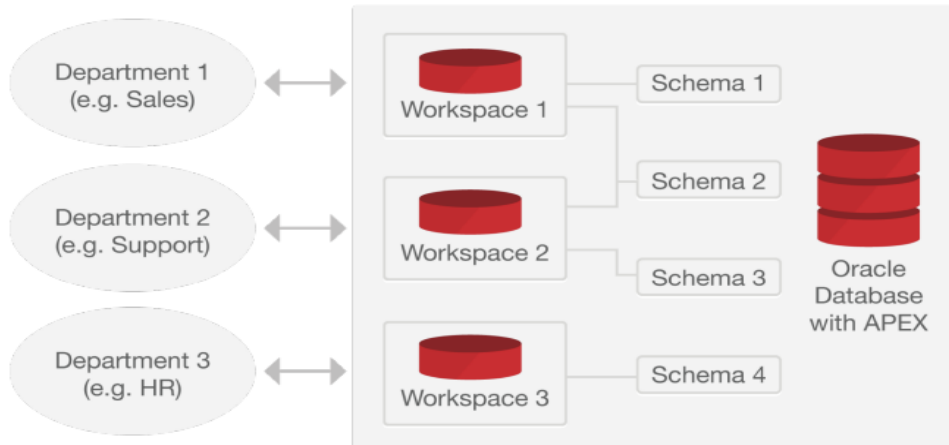


8. Data Sources Process: Data Sources

Utilize the Oracle Database / ORDS / Web Services



9. Single Database Instance / Multiple Workspaces: Easily managed with optional self-service provisioning.



- Workspaces used to define application definitions / Schemas hold data
- Many-to-many relationship between Workspaces and Schemas
- Instance Administrators manage the environment and schema access
- Departments can request more space, and access to a new schema

10. Available for maintaining SQL, PL/SQL, JavaScript, HTML, CSS, and large text sections:

- Syntax Highlighting
- Undo / Redo
- Find/Replace
- Auto completion
- Code validation
- Used in Page Designer, Component View and SQL Workshop

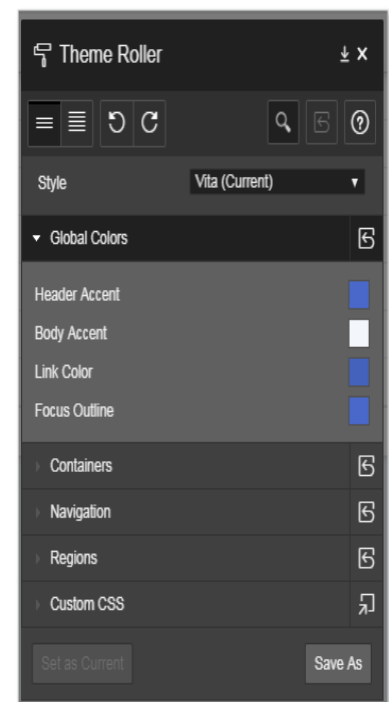


```
1 select
2   'Monthly Sales' as label,
3   '$' || trim(to_char(nvl(sum(o.order_total),0),'999G999G999G999G990')) as value,
4   'f?p=||:APP_ID||:4:||:APP_SESSION||:::4,RIR:IRGTE_ORDER_DATE:':P1_THIS_MONTH as url
5 from demo_orders o
6 where order_timestamp >= to_date(to_char(sysdate,'YYYYMM')||'01','YYYYMMDD')
7 union all
8 select
9   'Monthly Orders' as label,
10  trim(to_char(count(distinct o.order_id),'999G999G999G999G990')) as value,
11  'f?p=||:APP_ID||:4:||:APP_SESSION||:::4,RIR:IRGTE_ORDER_DATE:':P1_THIS_MONTH as url
12 from demo_orders o
13 where order_timestamp >= to_date(to_char(sysdate,'YYYYMM')||'01','YYYYMMDD')
14 union all
15 select 'Total Products' as label,
16        trim(to_char(count(distinct o.product_name),'999G999G999G999G990')) as value.
```

11. Use Universal Theme:

Theme Styles and Theme Roller

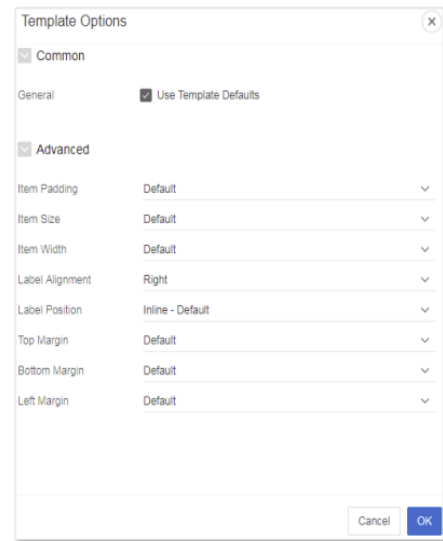
- Theme Styles provide a variety of different color schemes and styles for a single theme
- Defined as CSS file that is included in addition to the theme's base CSS file
- Universal Theme includes several pre-built styles
- Additional styles can be easily generated using the built-in Theme Roller utility



12. Use Template Options:

Template Options

- Declarative CSS modifiers for HTML5 templates
- Enables developers to declaratively apply styles like borders, padding, icons, colors, size, etc.
- Uses easy-to-understand, descriptive names
- Available for most template types:
 - Page, Region, Report, Breadcrumb, List, Item, Button
- Significantly reduces the number of required templates for a theme



13. Use Charting Engine:

Charting Engine

Declarative support for Oracle JET Data Visualization Components

- Based on Oracle JET Data Visualization Components
- Oracle Jet (JavaScript Extension Toolkit) is open source
- JavaScript-based solution, data being provided in JSON format
- Highly Customizable
 - Using 'JavaScript code' to provide function to be called when chart is initialized
 - Using Dynamic Actions to add interactive behavior
- Built-in Accessibility Support
- Built-in Responsive Design Support

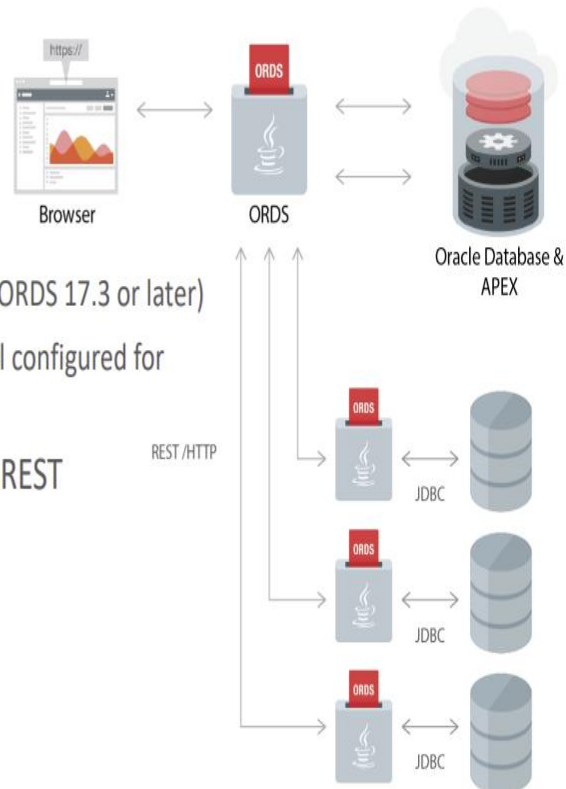


14.REST Enabled SQL Support:

REST Enabled SQL Support

Remote Database Connections

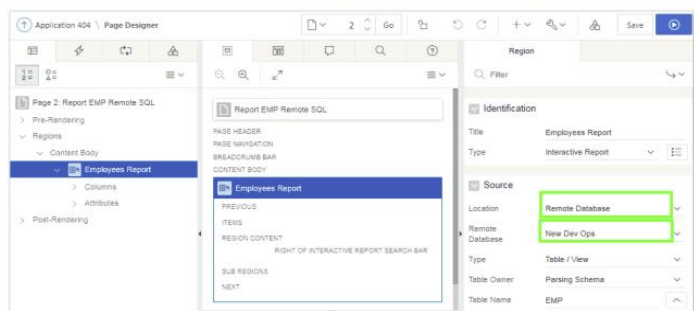
- Executing dynamic SQL or PL/SQL on remote databases using ORDS and REST
 - Relies on the ORDS "REST Enabled SQL" feature (ORDS 17.3 or later)
 - Requires ORDS instance having a connection pool configured for target database.
- Oracle APEX passes SQL query to ORDS over REST
- Self-describing JSON response
- Response contains a JSON object with:
 - Result set meta data (column names, data types)
 - The result data
 - Information about pagination



REST Enabled SQL Support

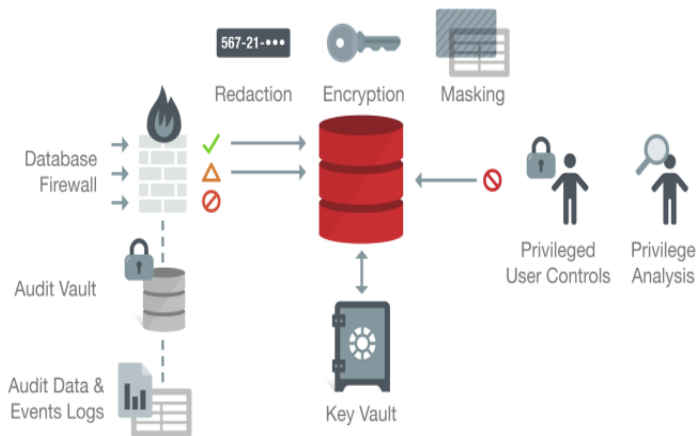
Using Remote Database Connections in Oracle APEX

- Create and maintain References to REST Enabled SQL Instances (Oracle REST Data Services) in Shared Components: A name, Endpoint URL, and Authentication information
- Use a REST Enabled SQL Service and therefore a remote Oracle database as the Data Source for Oracle APEX components like:
 - Classic Reports, Interactive Reports
 - Reflow Report, Toggle Columns Report
 - Tree Region, JET Charts
 - CSS Calendar
 - PL/SQL Process



15. Security features of the Database:

Security features of the Oracle Database



Works with APEX

Transparent Data Encryption	✓
Database Vault	✓
Audit Vault	✓
Database Firewall	✓
Key Vault	✓
Data Redaction	✓
Data Masking	✓
Auditing	✓

16. High Performance features of the Database:

High Performance features of the Oracle Database

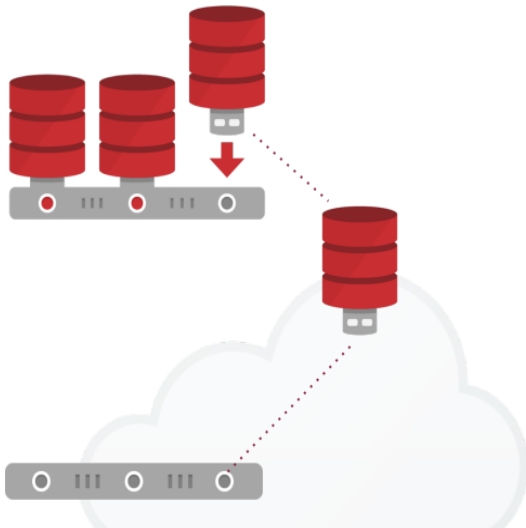


Works with APEX

Multi version read consistency	✓
Row level locking	✓
In-Memory	✓
Compression	✓
Partitioning	✓
Hadoop, Big Data SQL, R	✓
Cost based query optimizer	✓
Scale-out with Oracle RAC	✓
Exadata Smartscan & Infiniband	✓

17. Simplified consolidation and deployment with Oracle Multitenant:

Works with APEX



PDB Deployment	✓
PDB Cloning	✓
PDB Self Service App	✓
PDB Migration	✓
High Density Consolidation	✓
Automated Provisioning and Configuration	✓
PDB Resource Management	✓

18. Database Application Development Features:

Works with APEX

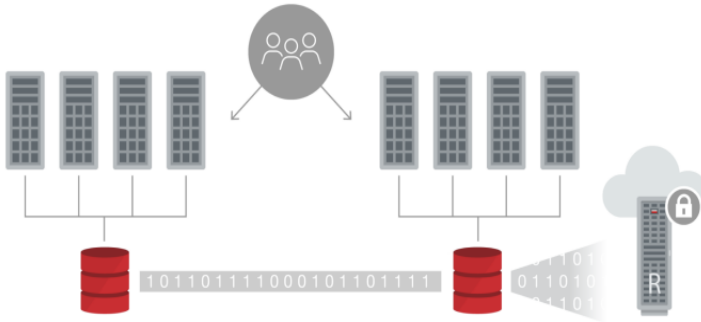


SQL and PL/SQL	✓
Java in Database	✓
Analytics	✓
Text	✓
Regular Expressions	✓
Spatial	✓
Pattern Matching	✓
Database XML	✓
Database JSON	✓
AQ	✓

19.High Availability features of the Database:

High Availability features of the Oracle Database

Works with APEX



Real Application Clusters	✓
Data Guard	✓
Active Data Guard	✓
Golden Gate	✓
Secure Backup	✓
Online Reorganization	✓
Edition-Based Redefinition	✓
Flashback, Total Recall	✓
Automated Rolling Upgrade of Database	✓
Failover to the Cloud	✓



Oracle Application Express

✓

✓

✓

Sign In

[Reset Password](#)

- After filling up credential information press "Sign In" and you will see the apex home screen

192.168.2.116:8181/devserver/f?p=4050:3:14638149179334:::
Search

ORACLE Application Express
Manage Requests
Manage Instance
Manage Workspaces
Monitor Activity



Manage Requests



Manage Instance



Manage Workspaces



Monitor Activity

Database Table Description

- **Uses Database Object:**

There are use some database object such as:

1. Table
2. Index
3. Trigger
4. Procedure
5. Function
6. Sequence
7. Lob
8. View
9. Package
10. Constraint
11. Job

Uses Object

Total uses object

SL	Object Name	Total
01	Table	355
02	Index	157
03	Trigger	366
04	Procedure	657
05	Function	33
06	Sequence	360
07	Lob	87
08	View	450
09	Package	300
10	Constraint	256
11	Job	35

DFD

SL	Table Name	Table Description	Table Space
01	APP_REMARKS	Remarks Information	TS_TRANSACTION
02	ASSING_HEADS	Budget Head Information	TS_TRANSACTION
03	App_Users	User Information	TS_TRANSACTION
04	BDG_ADD_DTL	Budget Information	TS_TRANSACTION
05	BDG_APP_MST	APP Information	TS_TRANSACTION
06	BDG_DED_DTL	Budget Deduction Information	TS_TRANSACTION
07	BDG_MD_DTL	MD Approval Information	TS_TRANSACTION
08	BDG_MET_LETTER	Letter Information	TS_TRANSACTION
09	BDG_REQ_DTL	Requisition Information	TS_TRANSACTION
10	BDG_MEETINGS	Metting Information	TS_TRANSACTION
11	BDG_TRANSFER_ADD	Transfer Amount Information	TS_TRANSACTION
12	BGD_CALCULATIVE_DATA	Calculation Information	TS_TRANSACTION
13	BGD_CLEARANCE	Budget Clearance Information	TS_TRANSACTION
14	BILL_VOUCHERS	Voucher Information	TS_TRANSACTION
15	FINANCIAL_YEARS	FY Information	TS_TRANSACTION
16	LEDGER_HEADS	Leager Information	TS_TRANSACTION

• Database Object Structure:

1. Table Structure

Description: USER Information TABLE SPACE: MASTER_DATA_TS

SL#	Column name	Type & Length	Description
1.	User	VARCHAR2 (10)	User Name
2.	PASSWORD	NUMBER (10)	Password
3.	USER_TYPE	VARCHAR2 (10)	User Type

Description: LEDGER_HEADS TABLE SPACE: MASTER_DATA_TS

SL#	Column name	Type & Length	Nullable
4.	LEDGER_HEAD_ID	NUMBER(10,0)	No
5.	U_ID	VARCHAR2(50)	Yes
6.	NAME_ENG	VARCHAR2(255)	No
7.	NAME_BNG	VARCHAR2(255)	Yes
8.	PARENT_ID	NUMBER	Yes

Description: BILL_VOUCHERS_REVISE
TABLE SPACE: MASTER_DATA_TS

SL#	Column name	Type & Length	Nullable
1.	BILL_VOUCHERS_RVS_ID	NUMBER	No
2.	BILL_VOUCHERS_ID	NUMBER	Yes
3.	VOUCHER_NO	VARCHAR2(14)	Yes
4.	VOUCHER_DT	DATE	Yes
5.	U_ID	VARCHAR2(100)	Yes
6.	HEAD_ID	NUMBER	Yes
7.	FI_YEAR_ID	NUMBER	Yes

Description: BOARD_SGR_DTL
TABLE SPACE: MASTER_DATA_TS

SL#	Column name	Type & Length	Nullable
1	ID	NUMBER	Yes
2	BOARD_MST_ID	NUMBER	Yes
3	HEAD_ID	NUMBER	Yes
4	FI_YEAR_ID	NUMBER	Yes
6	BRANCH_ID	NUMBER	Yes

Description: BILL_RECEVIERS
TABLE SPACE: MASTER_DATA_TS

SL#	Column name	Type & Length	Nullable
1.	BILL_RECEVIERS_ID	NUMBER(10,0)	No
2.	U_ID	VARCHAR2(50)	Yes
3.	NAME_ENG	VARCHAR2(255)	No
4.	NAME_BNG	VARCHAR2(255)	No
5.	BRANCH_ID	NUMBER	Yes

Description: BILL_VOUCHERS
TABLE SPACE: MASTER_DATA_TS

SL#	Column name	Type & Length	Nullable
1.	BILL_VOUCHERS_ID	NUMBER(10,0)	No
2.	VOUCHER_NO	VARCHAR2(14)	Yes
3.	VOUCHER_DT	DATE	Yes
4.	U_ID	VARCHAR2(100)	Yes
5.	HEAD_ID	NUMBER	Yes

Description: BGD_CLEARANCE
TABLE SPACE: MASTER_DATA_TS

SL#	Column name	Type & Length	Nullable
1.	CLR_ID	NUMBER	No
2.	MEMO_NO	VARCHAR2(50)	Yes
3.	NAME_OF_ENG	VARCHAR2(3600)	Yes
4.	BRANCH_ID	NUMBER(10,0)	Yes
5.	FI_YEAR	NUMBER(5,0)	Yes
6.	HEAD_ID	NUMBER(10,0)	Yes

Description: CONT_SCRTY_BILL_MST
TABLE SPACE: MASTER_DATA_TS

SL#	Column name	Type & Length	Nullable
1.	BILL_MST_ID	NUMBER	No
2.	ZONE_ID	NUMBER	Yes
3.	WO_ID	NUMBER	Yes
4.	TOTAL_AMT	NUMBER	Yes
5.	CONT_ID	NUMBER	Yes
6.	PRESENT_USER	VARCHAR2(55)	Yes
7.	STATUS	VARCHAR2(1)	Yes

Description: BDG_ZONE_FNL_APP_MST
TABLE SPACE: MASTER_DATA_TS

SL#	Column name	Type & Length	Nullable
1.	FNL_APP_MST_ID	NUMBER(10,0)	No
2.	REQ_ACC_MST_ID	NUMBER(10,0)	Yes
3.	MEETING_SERIAL_NO	NUMBER(10,0)	Yes
4.	REQ_MST_ID	NUMBER(10,0)	Yes
5.	MEMO_NO	VARCHAR2(255)	Yes
6.	U_ID	VARCHAR2(100)	Yes
7.	REQ_DT	DATE	Yes

2. Index Structure

```
CREATE UNIQUE INDEX "TREEDATA_PK" ON "TREEDATA" ("NODE_ID")
```

3. Job Structure

```
BEGIN
    DBMS_SCHEDULER.CREATE_SCHEDULE (
        schedule_name => 'my_stats_schedule',
        start_date     => SYSTIMESTAMP,
        end_date       => SYSTIMESTAMP + INTERVAL '30' day,
        repeat_interval => 'FREQ=HOURLY; INTERVAL=4',
        comments       => 'Every 4 hours');
END;
```

4. Trigger Structure

```
5. CREATE OR REPLACE EDITIONABLE TRIGGER "TRG_BDG_ADD_MST"
6. BEFORE INSERT OR UPDATE OR DELETE
7. ON BUDGET.BDG_ADD_MST
8. FOR EACH ROW
9. BEGIN
10. IF INSERTING
11. THEN
12. :NEW.CREATED_BY := UPPER (NVL (v ('APP_USER'), USER));
13. :NEW.CREATED_AT := SYSDATE;
14. ELSIF UPDATING
15. THEN
16. :NEW.UPDATED_BY := UPPER (NVL (v ('APP_USER'), USER));
17. :NEW.UPDATED_AT := SYSDATE;
18. END IF;
19. END;
20.
21. /
```

```

create or replace PROCEDURE    PROC_INT_TNS_ADD_APP (
    PI_BDG_ADD_APP_MST_ID IN  NUMBER,
    PO_ERROR              OUT VARCHAR2)
IS
    HEAD_EXP      EXCEPTION;
    FI_YEAR_EXP   EXCEPTION;
    BRANCH_EXP    EXCEPTION;
    BOOKING_EXP   EXCEPTION;
    BAL_AMT_EXP   EXCEPTION;
    V_REQ_MST     NUMBER;
    V_REQ_DTL     NUMBER;
    V_BDG_BAL_AMT NUMBER;
    V_HEAD_ID     NUMBER;
    V_FI_YEAR_ID  NUMBER;
    V_BRANCH_ID   NUMBER;
    V_BOOKING_AMOUNT NUMBER;

    CURSOR C1 IS
        SELECT A.FI_YEAR_ID,
               B.BRANCH_ID,
               B.HEAD_ID,
               B.ADD_APP_AMT
        FROM BDG_ADD_APP_MST A, BDG_ADD_APP_DTL B
        WHERE  A.BDG_ADD_APP_MST_ID = B.BDG_ADD_APP_MST_ID
              AND A.BDG_ADD_APP_MST_ID = PI_BDG_ADD_APP_MST_ID;
BEGIN
    FOR REC IN C1
    LOOP
        V_HEAD_ID := REC.HEAD_ID;
        V_FI_YEAR_ID := REC.FI_YEAR_ID;
        V_BRANCH_ID := REC.BRANCH_ID;
        V_BOOKING_AMOUNT := REC.ADD_APP_AMT;

        IF V_HEAD_ID IS NULL
        THEN
            RAISE HEAD_EXP;
        END IF;

        IF V_FI_YEAR_ID IS NULL
        THEN
            RAISE FI_YEAR_EXP;
        END IF;

        IF V_BRANCH_ID IS NULL
        THEN
            RAISE BRANCH_EXP;
        END IF;

        IF V_BOOKING_AMOUNT IS NULL
        THEN
            RAISE BOOKING_EXP;
        END IF;

        SELECT COUNT (*)
        INTO V_REQ_MST
        FROM BDG_ZONE_FNL_APP_MST
        WHERE BRANCH_ID = V_BRANCH_ID AND FI_YEAR_ID = V_FI_YEAR_ID;

        SELECT COUNT (*)
        INTO V_REQ_DTL
        FROM BDG_ZONE_FNL_APP_DTL
        WHERE  HEAD_ID = V_HEAD_ID
              AND REQ_MST_ID =
                (SELECT REQ_MST_ID
                 FROM BDG_ZONE_FNL_APP_MST
                 WHERE  BRANCH_ID = V_BRANCH_ID
                      AND FI_YEAR_ID = V_FI_YEAR_ID);

        SELECT BDG_BAL_AMT

```

```

    INTO V_BDG_BAL_AMT
    FROM BDG_ZONE_FNL_APP_DTL
    WHERE HEAD_ID = V_HEAD_ID
    AND REQ_MST_ID =
      (SELECT REQ_MST_ID
       FROM BDG_ZONE_FNL_APP_MST
       WHERE BRANCH_ID = V_BRANCH_ID
        AND FI_YEAR_ID = V_FI_YEAR_ID);

    IF V_BOOKING_AMOUNT > V_BDG_BAL_AMT
    THEN
      RAISE BAL_AMT_EXP;
    END IF;

    IF V_REQ_MST = 1 AND V_REQ_DTL = 1
    THEN
      UPDATE BDG_ZONE_FNL_APP_DTL A
      SET BGD_TNS_ADD_AMT =
        NVL (BGD_TNS_ADD_AMT, 0) + NVL (V_BOOKING_AMOUNT, 0),
        BDG_BAL_AMT =
        NVL (BDG_BAL_AMT, 0) + NVL (V_BOOKING_AMOUNT, 0)
      WHERE HEAD_ID = V_HEAD_ID
        AND REQ_MST_ID =
          (SELECT REQ_MST_ID
           FROM BDG_ZONE_FNL_APP_MST
           WHERE BRANCH_ID = V_BRANCH_ID
            AND FI_YEAR_ID = V_FI_YEAR_ID);
    ELSIF V_REQ_MST = 0
    THEN
      po_error := 'This zone approve amount does not found';
    ELSIF V_REQ_MST > 1
    THEN
      po_error := 'Multiple approve amount found';
    ELSIF V_REQ_DTL = 0
    THEN
      po_error := 'This head amount does not found';
    ELSIF V_REQ_DTL > 1
    THEN
      po_error := 'This head does not found';
    END IF;
  END LOOP;

  IF PO_ERROR IS NULL
  THEN
    COMMIT;
    po_error := '1';
  ELSE
    ROLLBACK;
  END IF;
/*
DECLARE
  V_ERROR VARCHAR2 (200);
BEGIN
  BUDGET.PROC_INT_TNS_DED_APP (PI_BDG_DED_APP_MST_ID => '1',
                              PO_ERROR => V_ERROR);
  DBMS_OUTPUT.PUT_LINE (V_ERROR);
END;
*/
EXCEPTION
  WHEN HEAD_EXP
  THEN
    PO_ERROR := 'Head Id must be entry value';
    ROLLBACK;
  WHEN FI_YEAR_EXP
  THEN
    PO_ERROR := 'Financial Id must be entry value';
    ROLLBACK;
  WHEN BRANCH_EXP
  THEN
    PO_ERROR := 'Branch Id must be entry value';

```

```

        ROLLBACK;
    WHEN BOOKING_EXP
    THEN
        PO_ERROR := 'Booking amount must be entry value';
        ROLLBACK;
    WHEN BAL_AMT_EXP
    THEN
        PO_ERROR := 'Booking amount more then Balance amount';
        ROLLBACK;
    WHEN OTHERS
    THEN
        PO_ERROR := SQLCODE || ' ' || SQLERRM;
        ROLLBACK;
END PROC_INT_TNS_ADD_APP;

```

Function Structure

```

create or replace FUNCTION FNC_MENU_ROLE_STATUS(
P_ROLE_ID NUMBER,
P_MENU_ID NUMBER,
P_STATUS VARCHAR2)
RETURN CHAR
IS
V_RET_STATUS CHAR(1) := NULL;
V_PER_STATUS VARCHAR2(20);
BEGIN
V_PER_STATUS := P_STATUS;
IF V_PER_STATUS = 'VIEW_S' THEN
SELECT NVL(VIEW_STATUS,'N') INTO V_RET_STATUS
FROM ROLE_MENUS
WHERE ROLE_ID = P_ROLE_ID
AND MENU_ID = P_MENU_ID;
RETURN V_RET_STATUS;
END IF;
IF V_PER_STATUS = 'INSERT_S' THEN
SELECT NVL(INSERT_STATUS,'N') INTO V_RET_STATUS
FROM ROLE_MENUS
WHERE ROLE_ID = P_ROLE_ID
AND MENU_ID = P_MENU_ID;
RETURN V_RET_STATUS;
END IF;
IF V_PER_STATUS = 'UPDATE_S' THEN
SELECT NVL(UPDATE_STATUS,'N') INTO V_RET_STATUS
FROM ROLE_MENUS
WHERE ROLE_ID = P_ROLE_ID
AND MENU_ID = P_MENU_ID;
RETURN V_RET_STATUS;
END IF;
IF V_PER_STATUS = 'DELETE_S' THEN
SELECT NVL(DELETE_STATUS,'N') INTO V_RET_STATUS
FROM ROLE_MENUS
WHERE ROLE_ID = P_ROLE_ID
AND MENU_ID = P_MENU_ID;
RETURN V_RET_STATUS;
END IF;
EXCEPTION
WHEN OTHERS THEN
RETURN 'N';
END;
```

Sequence Structure

```
CREATE SEQUENCE "SEQ_IMPREST_BEFTN" MINVALUE 1 MAXVALUE  
99999999999999999999999999999999 INCREMENT BY 1 START WITH 2641 CACHE 20  
NOORDER NOCYCLE NOKEEP NOSCALE GLOBAL
```

Lov Structure

```
SELECT COUNT(*) TOT
INTO :G_BILL_APP_FROM_CONT
FROM V_BILL_APP_FROM_CONT A
where (A.ZONE_ID IN (SELECT BRANCH_ID FROM APP_USERS WHERE LOGIN_ID =
:APP_USER AND USER_LEVEL IN (1,4))
OR A.ZONE_ID IN (SELECT DISTINCT BRANCH_ID FROM APP_USERS WHERE
'ADMIN'=:APP_USER));
```

View Structure

```
CREATE OR REPLACE FORCE EDITIONABLE VIEW "ALL_PENDING_BILL_CASH_MST_VW"
("MODULE_SHORT_NAME", "MODULE_NAME", "PAY_BANK_AC", "NO_OF_VOUCHERS", "AMOUNT",
"ISSUE_CHEQUE", "ISSUE_ADVICE") AS
SELECT
'CONT_BILL',
'Contractor Bill',
b.bank_name
|| ' - '
|| b.sender_account_no "Pay Bank A/C",
COUNT(a.bill_cont_acc_mst_id) "No of Vouchers",
SUM(a.tot_pay_amount) "Amount",
a.sender_id "Issue Cheque",
a.sender_id "Issue Advice"
FROM
bill_cont_acc_reg_book a,
wasa_sender_acc b
WHERE
a.sender_id = b.sender_id
AND A.bill_cont_acc_reg_book_id NOT IN (
SELECT
bill_cont_acc_reg_book_id
FROM
bill_cont_acc_cr_bk
)
GROUP BY
b.bank_name,
b.sender_account_no,
a.sender_id
UNION ALL
SELECT
'MOBILE_BILL',
'Mobile Bill',
b.bank_name
|| ' - '
|| b.sender_account_no "Pay Bank A/C",
COUNT(a.mobile_allow_acc_mst_id) "No of Vouchers",
SUM(a.total_pay_amount) "Amount",
a.sender_id "Issue Cheque",
a.sender_id "Issue Advice"
FROM
mobile_allow_acc_reg_book a,
wasa_sender_acc b
WHERE
a.sender_id = b.sender_id
AND a.mobile_allow_acc_reg_book_id NOT IN (
SELECT
mobile_allow_acc_reg_book_id
FROM
mobile_allow_acc_cash
)
GROUP BY
b.bank_name,
b.sender_account_no,
a.sender_id
UNION ALL
SELECT
```

```

        'ROAD_CUTTING_BILL',
        'Road Cutting Bill',
        b.bank_name
        || ' - '
        || b.sender_account_no "Pay Bank A/C",
        COUNT(a.road_cutting_acc_mst_id) "No of Vouchers",
        SUM(a.total_pay_amount) "Amount",
        a.sender_id "Issue Cheque",
        a.sender_id "Issue Advice"
FROM
    road_cutting_acc_reg_book a,
    wasa_sender_acc b
WHERE
    a.sender_id = b.sender_id
    AND a.road_cutting_acc_reg_book_id NOT IN (
        SELECT
            road_cutting_acc_reg_book_id
        FROM
            road_cutting_acc_cash
        )
GROUP BY
    b.bank_name,
    b.sender_account_no,
    a.sender_id /

```

Package Structure Specification

```

create or replace PACKAGE      PKG_FUEL
IS
    PROCEDURE PRC_INT_FIX_FUEL_INS (
        PI_FUEL_VEH_BILL_MST_ID IN NUMBER,
        PI_U_ID                IN VARCHAR2,
        PI_LEDGER_HEAD_ID      IN NUMBER,
        PI_REMARKS              IN VARCHAR2,
        PI_FILE_NO              IN VARCHAR2,
        PI_VOUCHER_DT           IN DATE,
        PI_RECEVIER_ID          IN NUMBER,
        PI_USER                  IN VARCHAR2,
        PI_MEMO_NO              IN VARCHAR2,
        PI_MEMO_DT              IN DATE,
        PO_ERROR                OUT VARCHAR2);

    /*
SET SERVEROUTPUT ON;

DECLARE
    V_ERROR VARCHAR2 (500);
BEGIN
    BUDGET.PKG_FUEL.PRC_INT_FIX_FUEL_INS (
        PI_FUEL_VEH_BILL_MST_ID =>",
        PI_U_ID                =>",
        PI_LEDGER_HEAD_ID      =>",
        PI_REMARKS              =>",
        PI_FILE_NO              =>",
        PI_VOUCHER_DT           =>",
        PI_RECEVIER_ID          =>",
        PI_USER                  =>",
        PI_MEMO_NO              =>",
        PI_MEMO_DT              =>",
        PO_ERROR                =>V_ERROR);
    DBMS_OUTPUT.PUT_LINE (V_ERROR);
END;
*/

    PROCEDURE PRC_BILL_VOUCHERS_FUEL_INS (PI_U_ID                VARCHAR2,
        PI_HEAD_ID          NUMBER,
        PI_BOOKING_AMOUNT   NUMBER,
        PI_LEDGER_HEAD_ID   NUMBER,
        PI_BRANCH_ID        NUMBER,
        PI_REMARKS           VARCHAR2,
        PI_FILE_NO           VARCHAR2,
        PI_FI_YEAR_ID        NUMBER,
        PI_VOUCHER_DT         DATE,
        PI_RECEVIER_ID       NUMBER,
        PI_USER              VARCHAR2,
        PI_MEMO_NO           VARCHAR2,
        PI_MEMO_DT           DATE,
        PI_FUEL_VEH_BILL_MST_ID NUMBER,
        PI_VOUCHER_NO_COMMON VARCHAR2,
        PO_ERROR             OUT VARCHAR2);

```

```
PROCEDURE PRC_INT_FIX_FUEL_UPD (PI_NOTESHEET_ID    NUMBER,
                                PO_ERROR           OUT VARCHAR2);
```

```
END;
```

Package Body

```
create or replace PACKAGE BODY    PKG_FUEL
IS
```

```
PROCEDURE PRC_INT_FIX_FUEL_INS (
    PI_FUEL_VEH_BILL_MST_ID IN    NUMBER,
    PI_U_ID                 IN    VARCHAR2,
    PI_LEDGER_HEAD_ID       IN    NUMBER,
    PI_REMARKS              IN    VARCHAR2,
    PI_FILE_NO              IN    VARCHAR2,
    PI_VOUCHER_DT           IN    DATE,
    PI_RECEVIER_ID          IN    NUMBER,
    PI_USER                 IN    VARCHAR2,
    PI_MEMO_NO              IN    VARCHAR2,
    PI_MEMO_DT              IN    DATE,
    PO_ERROR                OUT VARCHAR2)
IS
```

```
HEAD_EXP      EXCEPTION;
FI_YEAR_EXP   EXCEPTION;
BRANCH_EXP    EXCEPTION;
BOOKING_EXP   EXCEPTION;
BAL_AMT_EXP   EXCEPTION;
V_REQ_MST     NUMBER;
V_REQ_DTL     NUMBER;
V_BDG_BAL_AMT NUMBER;
V_HEAD_ID     NUMBER;
V_FI_YEAR_ID  NUMBER;
V_BRANCH_ID   NUMBER;
V_BOOKING_AMOUNT  NUMBER;
```

```
V_ESTMATE_CHECK_FLAG  NUMBER;
V_REVISED_CHECK_FLAG NUMBER;
```

```
V_BDG_BAL_AMT_RVSD  NUMBER;
V_REQ_MST_RVSD      NUMBER;
V_REQ_DTL_RVSD      NUMBER;
```

```
V_BDG_DED_AMT_ESTMT  NUMBER;
V_VOUCHER_NO_COMMON  VARCHAR2 (50);
V_COUNT              NUMBER;
```

```
CURSOR C1 IS
```

```
SELECT B.BRANCH_ID    ZONE_ID,
       A.FI_YEAR_ID,
       A.HEAD_ID,
       B.NET_AMT
FROM CDDL_BILL.FUEL_VEH_BILL_MST A,
     CDDL_BILL.FUEL_VEH_BILL_DTL B
WHERE  A.FUEL_VEH_BILL_MST_ID = B.FUEL_VEH_BILL_MST_ID
AND A.FUEL_VEH_BILL_MST_ID = PI_FUEL_VEH_BILL_MST_ID;
```

```
BEGIN
```

```
BEGIN
SELECT COUNT (*) INTO V_COUNT FROM BUDGET.BL_VCR_FUEL_VEH_BILL;
```

```
IF V_COUNT > 0
THEN
SELECT NVL (MAX (TO_NUMBER (VOUCHER_NO_COMMON)), 0) + 1
INTO V_VOUCHER_NO_COMMON
FROM BUDGET.BL_VCR_FUEL_VEH_BILL;
```

```
ELSE
V_VOUCHER_NO_COMMON := '1';
END IF;
```

```
EXCEPTION
WHEN OTHERS
THEN
V_VOUCHER_NO_COMMON := '1';
END;
```

```
FOR REC IN C1
```

```
LOOP
V_HEAD_ID := REC.HEAD_ID;
V_FI_YEAR_ID := REC.FI_YEAR_ID;
V_BRANCH_ID := REC.ZONE_ID;
V_BOOKING_AMOUNT := REC.NET_AMT;
```

```
IF V_HEAD_ID IS NULL
THEN
RAISE HEAD_EXP;
END IF;
```

```
IF V_FI_YEAR_ID IS NULL
THEN
RAISE FI_YEAR_EXP;
```

```

END IF;

IF V_BRANCH_ID IS NULL
THEN
    RAISE BRANCH_EXP;
END IF;

IF V_BOOKING_AMOUNT IS NULL
THEN
    RAISE BOOKING_EXP;
END IF;

SELECT COUNT (*)
INTO V_ESTIMATE_CHECK_FLAG
FROM BDG_ESTMT_RVSD_YR_CHECK
WHERE ESTIMATE_CHECK_FLAG = 'Y' AND FI_YEAR_ID = V_FI_YEAR_ID;

SELECT COUNT (*)
INTO V_REVISIED_CHECK_FLAG
FROM BDG_ESTMT_RVSD_YR_CHECK
WHERE REVISED_CHECK_FLAG = 'Y' AND FI_YEAR_ID = V_FI_YEAR_ID;

IF V_ESTIMATE_CHECK_FLAG = 1
THEN
    SELECT COUNT (*)
    INTO V_REQ_MST
    FROM BDG_ZONE_FNL_APP_MST
    WHERE BRANCH_ID = V_BRANCH_ID AND FI_YEAR_ID = V_FI_YEAR_ID;

    SELECT COUNT (*)
    INTO V_REQ_DTL
    FROM BDG_ZONE_FNL_APP_DTL
    WHERE HEAD_ID = V_HEAD_ID
    AND REQ_MST_ID =
        (SELECT REQ_MST_ID
         FROM BDG_ZONE_FNL_APP_MST
         WHERE BRANCH_ID = V_BRANCH_ID
         AND FI_YEAR_ID = V_FI_YEAR_ID);

    SELECT BDG_BAL_AMT
    INTO V_BDG_BAL_AMT
    FROM BDG_ZONE_FNL_APP_DTL
    WHERE HEAD_ID = V_HEAD_ID
    AND REQ_MST_ID =
        (SELECT REQ_MST_ID
         FROM BDG_ZONE_FNL_APP_MST
         WHERE BRANCH_ID = V_BRANCH_ID
         AND FI_YEAR_ID = V_FI_YEAR_ID);

    IF V_BOOKING_AMOUNT > V_BDG_BAL_AMT
    THEN
        RAISE BAL_AMT_EXP;
    END IF;

    IF V_REQ_MST = 1 AND V_REQ_DTL = 1
    THEN
        PRC_BILL_VOUCHERS_FUEL_INS (
            PI_U_ID          => PI_U_ID,
            PI_HEAD_ID       => V_HEAD_ID,
            PI_BOOKING_AMOUNT => V_BOOKING_AMOUNT,
            PI_LEDGER_HEAD_ID => PI_LEDGER_HEAD_ID,
            PI_BRANCH_ID     => V_BRANCH_ID,
            PI_REMARKS       => PI_REMARKS,
            PI_FILE_NO       => PI_FILE_NO,
            PI_FI_YEAR_ID    => V_FI_YEAR_ID,
            PI_VOUCHER_DT    => PI_VOUCHER_DT,
            PI_RECEVIER_ID   => PI_RECEVIER_ID,
            PI_USER          => PI_USER,
            PI_MEMO_NO       => PI_MEMO_NO,
            PI_MEMO_DT       => PI_MEMO_DT,
            PI_FUEL_VEH_BILL_MST_ID => PI_FUEL_VEH_BILL_MST_ID,
            PI_VOUCHER_NO_COMMON => V_VOUCHER_NO_COMMON,
            PO_ERROR        => PO_ERROR);

        UPDATE BDG_ZONE_FNL_APP_DTL A
        SET BDG_FUEL_FIX_AMT =
            NVL (BDG_FUEL_FIX_AMT, 0)
            + NVL (V_BOOKING_AMOUNT, 0),
            BDG_BAL_AMT =
            NVL (BDG_BAL_AMT, 0)
            - NVL (V_BOOKING_AMOUNT, 0)
        WHERE HEAD_ID = V_HEAD_ID
        AND REQ_MST_ID =
            (SELECT REQ_MST_ID
             FROM BDG_ZONE_FNL_APP_MST
             WHERE BRANCH_ID = V_BRANCH_ID
             AND FI_YEAR_ID = V_FI_YEAR_ID);
    
```

```

UPDATE CDDL_BILL.FUEL_GEN_ACC_MST
SET BUDGET_BOOK_FLAG = 'Y',
    VOUCHER_NO_COMMON = V_VOUCHER_NO_COMMON    --,
--BUDGET_BOOKED_BY = :APP_USER
WHERE FUEL_GEN_MST_ID = PI_FUEL_VEH_BILL_MST_ID;
ELSIF V_REQ_MST = 0
THEN
    po_error := 'This zone approve amount does not found';
ELSIF V_REQ_MST > 1
THEN
    po_error := 'Multiple approve amount found';
ELSIF V_REQ_DTL = 0
THEN
    po_error := 'This head amount does not found';
ELSIF V_REQ_DTL > 1
THEN
    po_error := 'This head does not found';
END IF;
ELSIF V_REVISD_CHECK_FLAG = 1
THEN
    SELECT COUNT (*)
    INTO V_REQ_MST
    FROM BDG_ZONE_FNL_APP_MST
    WHERE BRANCH_ID = V_BRANCH_ID AND FI_YEAR_ID = V_FI_YEAR_ID;

    SELECT COUNT (*)
    INTO V_REQ_MST_RVSD
    FROM BDG_ZONE_FNL_APP_MST
    WHERE BRANCH_ID = V_BRANCH_ID
        AND FI_YEAR_ID = V_FI_YEAR_ID + 1;

    SELECT COUNT (*)
    INTO V_REQ_DTL
    FROM BDG_ZONE_FNL_APP_DTL
    WHERE HEAD_ID = V_HEAD_ID
        AND REQ_MST_ID =
        (SELECT REQ_MST_ID
        FROM BDG_ZONE_FNL_APP_MST
        WHERE BRANCH_ID = V_BRANCH_ID
            AND FI_YEAR_ID = V_FI_YEAR_ID);

    SELECT COUNT (*)
    INTO V_REQ_DTL_RVSD
    FROM BDG_ZONE_FNL_APP_DTL
    WHERE HEAD_ID = V_HEAD_ID
        AND REQ_MST_ID =
        (SELECT REQ_MST_ID
        FROM BDG_ZONE_FNL_APP_MST
        WHERE BRANCH_ID = V_BRANCH_ID
            AND FI_YEAR_ID = V_FI_YEAR_ID + 1);

BEGIN
    SELECT NVL (BDG_DED_AMT, 0)
    INTO V_BDG_DED_AMT_ESTMT
    FROM BDG_ZONE_FNL_APP_DTL
    WHERE HEAD_ID = V_HEAD_ID
        AND REQ_MST_ID =
        (SELECT REQ_MST_ID
        FROM BDG_ZONE_FNL_APP_MST
        WHERE BRANCH_ID = V_BRANCH_ID
            AND FI_YEAR_ID = V_FI_YEAR_ID);
EXCEPTION
    WHEN OTHERS
    THEN
        V_BDG_DED_AMT_ESTMT := 0;
END;

SELECT NVL (BDG_BAL_AMT_RVSD, 0)
INTO V_BDG_BAL_AMT_RVSD
FROM BDG_ZONE_FNL_APP_DTL
WHERE HEAD_ID = V_HEAD_ID
    AND REQ_MST_ID =
    (SELECT REQ_MST_ID
    FROM BDG_ZONE_FNL_APP_MST
    WHERE BRANCH_ID = V_BRANCH_ID
        AND FI_YEAR_ID = V_FI_YEAR_ID + 1);

V_BDG_BAL_AMT := V_BDG_BAL_AMT_RVSD - V_BDG_DED_AMT_ESTMT;

IF V_BOOKING_AMOUNT > V_BDG_BAL_AMT
THEN
    RAISE BAL_AMT_EXP;
END IF;

IF V_REQ_MST = 1          --AND V_REQ_DTL = 1
    AND V_REQ_MST_RVSD = 1
    AND V_REQ_DTL_RVSD = 1
THEN

```

```

PRC_BILL_VOUCHERS_FUEL_INS (
    PI_U_ID      => PI_U_ID,
    PI_HEAD_ID   => V_HEAD_ID,
    PI_BOOKING_AMOUNT => V_BOOKING_AMOUNT,
    PI_LEDGER_HEAD_ID => PI_LEDGER_HEAD_ID,
    PI_BRANCH_ID => V_BRANCH_ID,
    PI_REMARKS    => PI_REMARKS,
    PI_FILE_NO    => PI_FILE_NO,
    PI_FI_YEAR_ID => V_FI_YEAR_ID,
    PI_VOUCHER_DT => PI_VOUCHER_DT,
    PI_RECEVIER_ID => PI_RECEVIER_ID,
    PI_USER       => PI_USER,
    PI_MEMO_NO    => PI_MEMO_NO,
    PI_MEMO_DT    => PI_MEMO_DT,
    PI_FUEL_VEH_BILL_MST_ID => PI_FUEL_VEH_BILL_MST_ID,
    PI_VOUCHER_NO_COMMON => V_VOUCHER_NO_COMMON,
    PO_ERROR      => PO_ERROR);

UPDATE BDG_ZONE_FNL_APP_DTL A
SET BDG_FUEL_FIX_AMT_RVSD =
    NVL (BDG_FUEL_FIX_AMT_RVSD, 0)
    + NVL (V_BOOKING_AMOUNT, 0),
    BDG_BAL_AMT_RVSD =
    NVL (BDG_BAL_AMT_RVSD, 0)
    - NVL (V_BOOKING_AMOUNT, 0)
WHERE HEAD_ID = V_HEAD_ID
AND REQ_MST_ID =
    (SELECT REQ_MST_ID
     FROM BDG_ZONE_FNL_APP_MST
     WHERE BRANCH_ID = V_BRANCH_ID
     AND FI_YEAR_ID = V_FI_YEAR_ID);

UPDATE CDDL_BILL.FUEL_GEN_ACC_MST
SET BUDGET_BOOK_FLAG = 'Y',
    VOUCHER_NO_COMMON = V_VOUCHER_NO_COMMON --,
    --BUDGET_BOOKED_BY = :APP_USER
WHERE FUEL_GEN_MST_ID = PI_FUEL_VEH_BILL_MST_ID;
ELSIF V_REQ_MST = 0
THEN
    po_error := 'This zone approve amount does not found';
ELSIF V_REQ_MST > 1
THEN
    po_error := 'Multiple approve amount found';
ELSIF V_REQ_DTL = 0
THEN
    po_error := 'This head amount does not found';
ELSIF V_REQ_DTL > 1
THEN
    po_error := 'This head does not found';
END IF;
END IF;
END LOOP;

IF PO_ERROR IS NULL
THEN
    COMMIT;
    po_error := '1';
ELSE
    ROLLBACK;
END IF;
EXCEPTION
WHEN HEAD_EXP
THEN
    PO_ERROR := 'Head Id must be entry value';
    ROLLBACK;
WHEN FI_YEAR_EXP
THEN
    PO_ERROR := 'Financial Id must be entry value';
    ROLLBACK;
WHEN BRANCH_EXP
THEN
    PO_ERROR := 'Branch Id must be entry value';
    ROLLBACK;
WHEN BOOKING_EXP
THEN
    PO_ERROR := 'Booking amount must be entry value';
    ROLLBACK;
WHEN BAL_AMT_EXP
THEN
    PO_ERROR := 'Booking amount more then Balance amount';
    ROLLBACK;
WHEN OTHERS
THEN
    PO_ERROR := SQLCODE || ' ' || SQLERRM;
    ROLLBACK;
END PRC_INT_FIX_FUEL_INS;

PROCEDURE PRC_BILL_VOUCHERS_FUEL_INS (
    PI_U_ID      VARCHAR2,

```

```

PI_HEAD_ID          NUMBER,
PI_BOOKING_AMOUNT   NUMBER,
PI_LEDGER_HEAD_ID   NUMBER,
PI_BRANCH_ID        NUMBER,
PI_REMARKS          VARCHAR2,
PI_FILE_NO          VARCHAR2,
PI_FI_YEAR_ID       NUMBER,
PI_VOUCHER_DT       DATE,
PI_RECEVIER_ID      NUMBER,
PI_USER             VARCHAR2,
PI_MEMO_NO          VARCHAR2,
PI_MEMO_DT          DATE,
PI_FUEL_VEH_BILL_MST_ID  NUMBER,
PI_VOUCHER_NO_COMMON VARCHAR2,
PO_ERROR            OUT VARCHAR2)
IS
V_BILL_VOUCHERS_ID NUMBER;
V_YEAR             VARCHAR2 (5);
V_VC               VARCHAR2 (14);
V_COUNT            NUMBER := 0;
V_BDG_MNST_AMT     NUMBER;
V_PRIV_BILL_AMOUNT NUMBER;
BEGIN
SELECT NVL (MAX (NVL (FUEL_VEH_BILL_MST_ID, 0)), 0) + 1
  INTO V_BILL_VOUCHERS_ID
  FROM BUDGET.BL_VCR_FUEL_VEH_BILL;

SELECT TRIM (NAME_SHORT)
  INTO V_YEAR
  FROM BUDGET.FINANCIAL_YEARS
 WHERE ID = PI_FI_YEAR_ID;

SELECT COUNT (*)
  INTO V_COUNT
  FROM BUDGET.BL_VCR_FUEL_VEH_BILL
 WHERE FI_YEAR_ID = PI_FI_YEAR_ID;

IF V_COUNT > 0
THEN
  SELECT  V_YEAR
        || '-'
        || LPAD (
            MAX (
              TO_NUMBER (
                NVL (SUBSTR (NVL (VOUCHER_NO, 0), 7, 8),
                0)))
            + 1,
            8,
            0)
        INTO V_VC
        FROM BUDGET.BL_VCR_FUEL_VEH_BILL
        WHERE FI_YEAR_ID = PI_FI_YEAR_ID;
ELSE
  V_VC := V_YEAR || '-' || '00000001';
END IF;

BEGIN
SELECT NVL (BDG_MNST_AMT, 0)
  INTO V_BDG_MNST_AMT
  FROM BUDGET.BDG_ZONE_FNL_APP_DTL A,
       BUDGET.BDG_ZONE_FNL_APP_MST B
 WHERE  A.FNL_APP_MST_ID = B.FNL_APP_MST_ID
        AND HEAD_ID = PI_HEAD_ID
        AND BRANCH_ID = PI_BRANCH_ID
        AND FI_YEAR_ID = PI_FI_YEAR_ID;
EXCEPTION
  WHEN OTHERS
  THEN
    V_BDG_MNST_AMT := 0;
END;

BEGIN
SELECT NVL (SUM (NVL (BOOKING_AMOUNT, 0)), 0)
  INTO V_PRIV_BILL_AMOUNT
  FROM BL_VCR_FUEL_VEH_BILL
 WHERE  HEAD_ID = PI_HEAD_ID
        AND BRANCH_ID = PI_BRANCH_ID
        AND FI_YEAR_ID = PI_FI_YEAR_ID;
EXCEPTION
  WHEN OTHERS
  THEN
    V_PRIV_BILL_AMOUNT := 0;
END;

INSERT INTO BL_VCR_FUEL_VEH_BILL (BL_VCR_FUEL_VEH_BILL_ID,
                                U_ID,
                                HEAD_ID,
                                BOOKING_AMOUNT,
                                LEDGER_HEAD_ID,

```

```

        BRANCH_ID,
        REMARKS,
        FILE_NO,
        FI_YEAR_ID,
        BUG_AMOUNT,
        PRIV_BILL_AMOUNT,
        REMAINING_AMOUNT,
        VOUCHER_NO,
        VOUCHER_DT,
        RECEVIER_ID,
        ACCOUNTANT_ID,
        MEMO_NO,
        MEMO_DT,
        SUBMIT_STATUS,
        SUBMIT_BY,
        SUBMIT_AT,
        ORG_ID,
        STATUS,
        FUEL_VEH_BILL_MST_ID,
        VOUCHER_NO_COMMON)
VALUES (
    V_BILL_VOUCHERS_ID,
    PI_U_ID,
    PI_HEAD_ID,
    PI_BOOKING_AMOUNT,
    PI_LEDGER_HEAD_ID,
    PI_BRANCH_ID,
    PI_REMARKS,
    PI_FILE_NO,
    PI_FI_YEAR_ID,
    V_BDG_MNST_AMT,
    NVL (V_PRIV_BILL_AMOUNT, 0),
    NVL (V_BDG_MNST_AMT, 0)
    - NVL (PI_BOOKING_AMOUNT, 0)
    - NVL (V_PRIV_BILL_AMOUNT, 0),
    V_VC,
    PI_VOUCHER_DT,
    PI_RECEVIER_ID,
    PI_USER,
    PI_MEMO_NO,
    PI_MEMO_DT,
    'Y',
    PI_USER,
    SYSDATE,
    1,
    'Y',
    PI_FUEL_VEH_BILL_MST_ID,
    PI_VOUCHER_NO_COMMON);
/*UPDATE CDDL_BILL.BILL_CONT_ACC_MST
SET BUDGET_BOOK_FLAG = 'Y', BUDGET_VOUCHER_ID = V_BILL_VOUCHERS_ID
WHERE BILL_MST_ID = PI_IMPREST_MST_ID;*/
EXCEPTION
WHEN OTHERS
THEN
    PO_ERROR := SQLCODE || ' ' || SQLERRM;
END PRC_BILL_VOUCHERS_FUEL_INS;

```

```

PROCEDURE PRC_INT_FIX_FUEL_UPD (PI_NOTESHEET_ID    NUMBER,
                                PO_ERROR           OUT VARCHAR2)

```

```

IS
    HEAD_EXP           EXCEPTION;
    FI_YEAR_EXP        EXCEPTION;
    BRANCH_EXP         EXCEPTION;
    BOOKING_EXP        EXCEPTION;
    BAL_AMT_EXP        EXCEPTION;
    V_REQ_MST          NUMBER;
    V_REQ_DTL          NUMBER;
    V_BDG_BAL_AMT      NUMBER;
    V_HEAD_ID          NUMBER;
    V_FI_YEAR_ID       NUMBER;
    V_BRANCH_ID        NUMBER;
    V_BOOKING_AMOUNT   NUMBER;

    V_ESTMATE_CHECK_FLAG NUMBER;
    V_REVISIED_CHECK_FLAG NUMBER;

    V_BDG_BAL_AMT_RVSD  NUMBER;
    V_REQ_MST_RVSD      NUMBER;
    V_REQ_DTL_RVSD      NUMBER;

    V_BDG_DED_AMT_ESTMT NUMBER;
    V_VOUCHER_NO_COMMON VARCHAR2 (50);
    V_COUNT             NUMBER;

```

```

CURSOR C1 IS
SELECT B.BRANCH_ID    ZONE_ID,
       B.FI_YEAR_ID,
       B.HEAD_ID,

```

```

        B.BOOKING_AMOUNT FUEL_AMT,
        B.FUEL_VEH_BILL_MST_ID
    FROM CDDL_BILL.FUEL_NOTESHEET_PRSNL A, BL_VCR_FUEL_VEH_BILL B
    WHERE A.FUEL_VEH_MST_ID = B.FUEL_VEH_BILL_MST_ID
        AND A.NOTESHEET_ID = PL_NOTESHEET_ID;
BEGIN
    FOR REC IN C1
    LOOP
        V_HEAD_ID := REC.HEAD_ID;
        V_FI_YEAR_ID := REC.FI_YEAR_ID;
        V_BRANCH_ID := REC.ZONE_ID;
        V_BOOKING_AMOUNT := REC.FUEL_AMT;

        IF V_HEAD_ID IS NULL
        THEN
            RAISE HEAD_EXP;
        END IF;

        IF V_FI_YEAR_ID IS NULL
        THEN
            RAISE FI_YEAR_EXP;
        END IF;

        IF V_BRANCH_ID IS NULL
        THEN
            RAISE BRANCH_EXP;
        END IF;

        IF V_BOOKING_AMOUNT IS NULL
        THEN
            RAISE BOOKING_EXP;
        END IF;

        SELECT COUNT (*)
        INTO V_ESTIMATE_CHECK_FLAG
        FROM BDG_ESTMT_RVSD_YR_CHECK
        WHERE ESTIMATE_CHECK_FLAG = 'Y' AND FI_YEAR_ID = V_FI_YEAR_ID;

        SELECT COUNT (*)
        INTO V_REVISED_CHECK_FLAG
        FROM BDG_ESTMT_RVSD_YR_CHECK
        WHERE REVISED_CHECK_FLAG = 'Y' AND FI_YEAR_ID = V_FI_YEAR_ID;

        IF V_ESTIMATE_CHECK_FLAG = 1
        THEN
            SELECT COUNT (*)
            INTO V_REQ_MST
            FROM BDG_ZONE_FNL_APP_MST
            WHERE BRANCH_ID = V_BRANCH_ID AND FI_YEAR_ID = V_FI_YEAR_ID;

            SELECT COUNT (*)
            INTO V_REQ_DTL
            FROM BDG_ZONE_FNL_APP_DTL
            WHERE HEAD_ID = V_HEAD_ID
                AND REQ_MST_ID =
                (SELECT REQ_MST_ID
                 FROM BDG_ZONE_FNL_APP_MST
                 WHERE BRANCH_ID = V_BRANCH_ID
                     AND FI_YEAR_ID = V_FI_YEAR_ID);

            SELECT BDG_FUEL_FIX_AMT
            INTO V_BDG_BAL_AMT
            FROM BDG_ZONE_FNL_APP_DTL
            WHERE HEAD_ID = V_HEAD_ID
                AND REQ_MST_ID =
                (SELECT REQ_MST_ID
                 FROM BDG_ZONE_FNL_APP_MST
                 WHERE BRANCH_ID = V_BRANCH_ID
                     AND FI_YEAR_ID = V_FI_YEAR_ID);

            IF V_BOOKING_AMOUNT > V_BDG_BAL_AMT
            THEN
                RAISE BAL_AMT_EXP;
            END IF;

            IF V_REQ_MST = 1 AND V_REQ_DTL = 1
            THEN
                UPDATE BDG_ZONE_FNL_APP_DTL A
                SET BDG_FUEL_FIX_AMT =
                    NVL (BDG_FUEL_FIX_AMT, 0)
                    - NVL (V_BOOKING_AMOUNT, 0),
                    BDG_DED_AMT =
                    NVL (BDG_DED_AMT, 0)
                    + NVL (V_BOOKING_AMOUNT, 0)
                WHERE HEAD_ID = V_HEAD_ID
                    AND REQ_MST_ID =
                    (SELECT REQ_MST_ID
                     FROM BDG_ZONE_FNL_APP_MST

```

```

        WHERE BRANCH_ID = V_BRANCH_ID
          AND FI_YEAR_ID = V_FI_YEAR_ID);

CDDL_BILL.FUEL_VEH_BILL_ACC_REG_CASH_INS (
  PI_FUEL_VEH_BILL_MST_ID => REC.FUEL_VEH_BILL_MST_ID,
  PI_APPROVE_FLAG => "",
  PO_ERROR => PO_ERROR);
ELSIF V_REQ_MST = 0
THEN
  po_error := 'This zone approve amount does not found';
ELSIF V_REQ_MST > 1
THEN
  po_error := 'Multiple approve amount found';
ELSIF V_REQ_DTL = 0
THEN
  po_error := 'This head amount does not found';
ELSIF V_REQ_DTL > 1
THEN
  po_error := 'This head does not found';
END IF;
ELSIF V_REVISSED_CHECK_FLAG = 1
THEN
  SELECT COUNT (*)
  INTO V_REQ_MST
  FROM BDG_ZONE_FNL_APP_MST
  WHERE BRANCH_ID = V_BRANCH_ID AND FI_YEAR_ID = V_FI_YEAR_ID;

  SELECT COUNT (*)
  INTO V_REQ_MST_RVSD
  FROM BDG_ZONE_FNL_APP_MST
  WHERE BRANCH_ID = V_BRANCH_ID
    AND FI_YEAR_ID = V_FI_YEAR_ID + 1;

  SELECT COUNT (*)
  INTO V_REQ_DTL
  FROM BDG_ZONE_FNL_APP_DTL
  WHERE HEAD_ID = V_HEAD_ID
    AND REQ_MST_ID =
      (SELECT REQ_MST_ID
       FROM BDG_ZONE_FNL_APP_MST
       WHERE BRANCH_ID = V_BRANCH_ID
        AND FI_YEAR_ID = V_FI_YEAR_ID);

  SELECT COUNT (*)
  INTO V_REQ_DTL_RVSD
  FROM BDG_ZONE_FNL_APP_DTL
  WHERE HEAD_ID = V_HEAD_ID
    AND REQ_MST_ID =
      (SELECT REQ_MST_ID
       FROM BDG_ZONE_FNL_APP_MST
       WHERE BRANCH_ID = V_BRANCH_ID
        AND FI_YEAR_ID = V_FI_YEAR_ID + 1);

/*BEGIN
  SELECT NVL (BDG_DED_AMT, 0)
  INTO V_BDG_DED_AMT_ESTMT
  FROM BDG_ZONE_FNL_APP_DTL
  WHERE HEAD_ID = V_HEAD_ID
    AND REQ_MST_ID =
      (SELECT REQ_MST_ID
       FROM BDG_ZONE_FNL_APP_MST
       WHERE BRANCH_ID = V_BRANCH_ID
        AND FI_YEAR_ID = V_FI_YEAR_ID);
EXCEPTION
  WHEN OTHERS
  THEN
    V_BDG_DED_AMT_ESTMT := 0;
END;*/

SELECT NVL (BDG_FUEL_FIX_AMT_RVSD, 0)
INTO V_BDG_BAL_AMT_RVSD
FROM BDG_ZONE_FNL_APP_DTL
WHERE HEAD_ID = V_HEAD_ID
  AND REQ_MST_ID =
    (SELECT REQ_MST_ID
     FROM BDG_ZONE_FNL_APP_MST
     WHERE BRANCH_ID = V_BRANCH_ID
      AND FI_YEAR_ID = V_FI_YEAR_ID + 1);

V_BDG_BAL_AMT := V_BDG_BAL_AMT_RVSD; -- V_BDG_DED_AMT_ESTMT;

IF V_BOOKING_AMOUNT > V_BDG_BAL_AMT
THEN
  RAISE BAL_AMT_EXP;
END IF;

IF V_REQ_MST = 1          --AND V_REQ_DTL = 1
  AND V_REQ_MST_RVSD = 1

```

```

AND V_REQ_DTL_RVSD = 1
THEN
UPDATE BDG_ZONE_FNL_APP_DTL A
SET BDG_FUEL_FIX_AMT_RVSD =
    NVL (BDG_FUEL_FIX_AMT_RVSD, 0)
    - NVL (V_BOOKING_AMOUNT, 0),
    BDG_DED_AMT_RVSD =
    NVL (BDG_DED_AMT_RVSD, 0)
    + NVL (V_BOOKING_AMOUNT, 0)
WHERE HEAD_ID = V_HEAD_ID
AND REQ_MST_ID =
(SELECT REQ_MST_ID
FROM BDG_ZONE_FNL_APP_MST
WHERE BRANCH_ID = V_BRANCH_ID
AND FI_YEAR_ID = V_FI_YEAR_ID);

CDDL_BILL.FUEL_VEH_BILL_ACC_REG_CASH_INS (
    PI_FUEL_VEH_BILL_MST_ID => REC.FUEL_VEH_BILL_MST_ID,
    PI_APPROVE_FLAG => "",
    PO_ERROR => PO_ERROR);
ELSIF V_REQ_MST = 0
THEN
    po_error := 'This zone approve amount does not found';
ELSIF V_REQ_MST > 1
THEN
    po_error := 'Multiple approve amount found';
ELSIF V_REQ_DTL = 0
THEN
    po_error := 'This head amount does not found';
ELSIF V_REQ_DTL > 1
THEN
    po_error := 'This head does not found';
END IF;
END IF;
END LOOP;

IF PO_ERROR IS NULL
THEN
    COMMIT;
    po_error := '1';
ELSE
    ROLLBACK;
END IF;
/*
DECLARE
    V_ERROR VARCHAR2 (200);
BEGIN
    BUDGET.PRC_INT_FIX_IMP_UPD (PI_NOTESHEET_ID => '1',
        PO_ERROR => V_ERROR);
    DBMS_OUTPUT.PUT_LINE (V_ERROR);
END;
*/
EXCEPTION
WHEN HEAD_EXP
THEN
    PO_ERROR := 'Head Id must be entry value';
    ROLLBACK;
WHEN FI_YEAR_EXP
THEN
    PO_ERROR := 'Financial Id must be entry value';
    ROLLBACK;
WHEN BRANCH_EXP
THEN
    PO_ERROR := 'Branch Id must be entry value';
    ROLLBACK;
WHEN BOOKING_EXP
THEN
    PO_ERROR := 'Booking amount must be entry value';
    ROLLBACK;
WHEN BAL_AMT_EXP
THEN
    PO_ERROR := 'Booking amount more then Balance amount';
    ROLLBACK;
WHEN OTHERS
THEN
    PO_ERROR := SQLCODE || ' ' || SQLERRM;
    ROLLBACK;
END PRC_INT_FIX_FUEL_UPD;
END;

```

22. Constraint Type Use

- a. Primary Key
- b. Check
- c. Foreign Key
- d. Unique

1. Create Tablespace Code

create tablespace TS_APEX_TRANSACTION

Datafile 'D:\APP\ORACLE\ORADATA\ORCLDB\TS_APEX_TRANSACTION.DBF' size 1024M
autoextend on;

```
Connected to:
Oracle Database 11g Enterprise Edition Release 11.2.0.1.0 - 64bit Production
With the Partitioning, OLAP, Data Mining and Real Application Testing options

SQL> CREATE TABLESPACE APEX_TS DATAFILE 'C:\Orahme11g\oradata\RND\apex_ts.dbf'
  2  size 4000M
  3  extent management local
  4  segment space management auto;

Tablespace created.

SQL> _
```

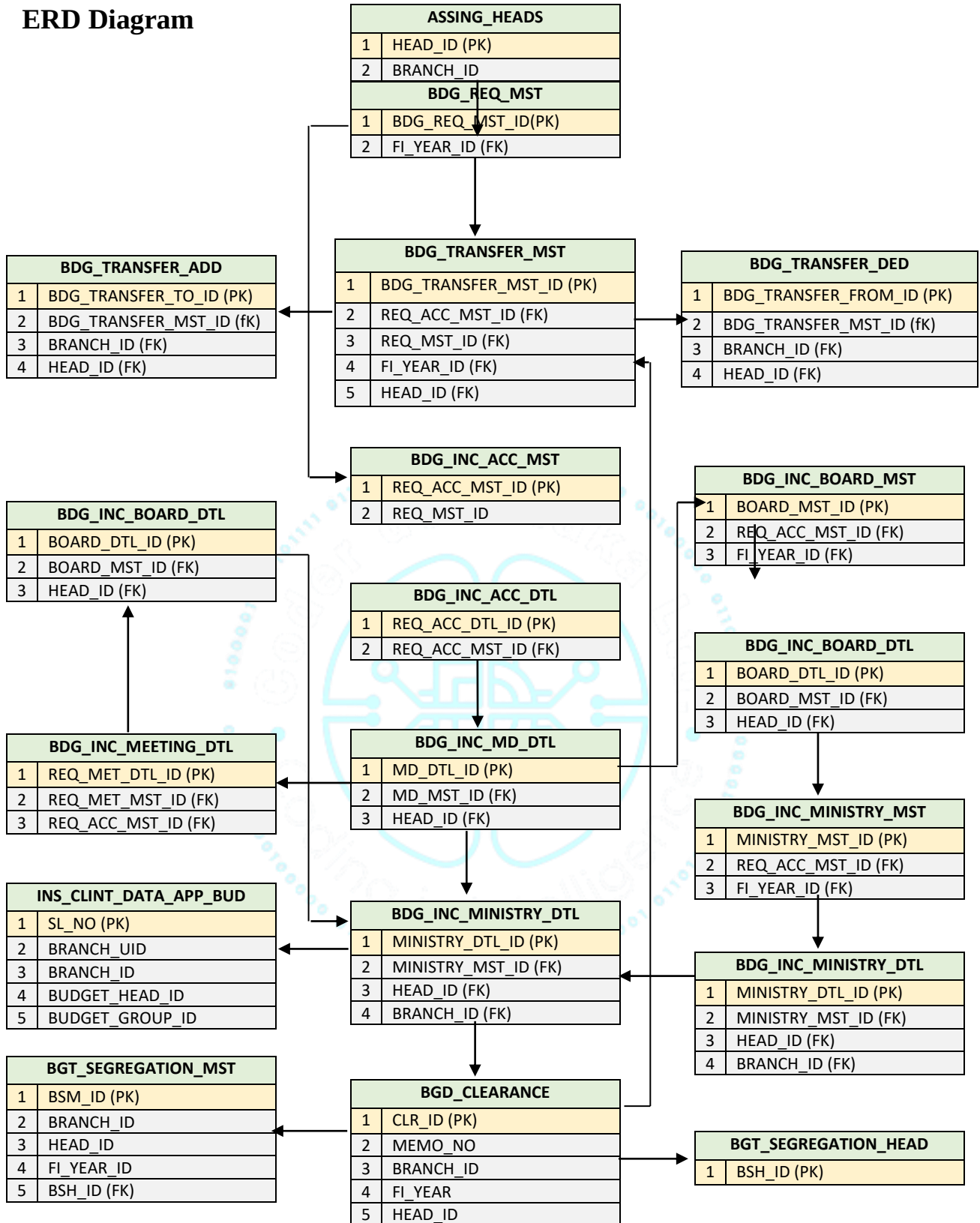
Application Backup

1. Click App
2. Click Export/Import
3. Click Export
4. Next
5. Export

Oracle Apex Import

1. Click App Builder
2. Click Import
3. Select Application
4. Next

ERD Diagram



Conclusion:

This all document has provided a comprehensive guide to the Budget Management System Software developed for Dhaka Water Supply and Sewerage Authority (WASA). The software offers a range of features and functionalities that enable efficient Payment, tracking, and reporting. By implementing this software, Dhaka WASA can streamline its Payment processes, enhance financial control, and make informed decisions based on accurate and real-time financial information.

Each section provides detailed instructions and guidance to ensure successful implementation and utilization of the software.

