# 

# Midterm Report

# of

# Practice School

# Clearance Optimization Engine

# SUBMITTED

# BY

# HEMASAI AISHWARYA V 130905580

# Under the Guidance of:

# PRAKASH K AITHAL SUNILCHOWDARY

# ASSISTANT PROFESSOR SENIOR MANAGER

# SENIOR SCALE

# DEPARTMENT OF CS&E STORES TEAM

# Manipal Institute of Technology ORACLE RETAIL

# 

# Midterm Report

# of

# Practice School

# Clearance Optimization Engine

# SUBMITTED

# BY

# HEMASAI AISHWARYA V 130905580

# Under the Guidance of:

# PRAKASH K AITHAL SUNILCHOWDARY

# ASSISTANT PROFESSOR SENIOR MANAGER

# SENIOR SCALE

# DEPARTMENT OF CS&E STORES TEAM Manipal Institute of Technology ORACLE RETAIL

**TABLE OF CONTENTS**

|  |  |  |
| --- | --- | --- |
| **Sl. No.** | **Title** | **Page No.** |
|  |  |  |
|  | About the Organization and Guide  Internship Offer Letter | 3  4 |
|  | Abstract | 5 |
| 1. | Introduction | 6 |
|  | 1.1.General Introduction to the topic | 6 |
|  | 1.2.Organization | 6 |
|  | 1.3.Area of Computer Science | 7 |
|  | 1.4.Hardware and Software Requirements | 7 |
| 2. | Problem Definition | 9 |
| 3. | Objectives | 10 |
| 4. | Background | 11 |
| 5. | Methodology | 16 |
| 6. | Implementation Details | 20 |
| 7. | Progress | 25 |
| 8. | References | 26 |

**DETAILS OF THE ORGANIZATION**

**Postal Address:**

Oracle India Pvt Ltd.

Prestige Technology Park -III, Valence Building,

Sy.No. 110/1,2&3

Amane Bellandur Khane Village,

Varthur Hobli,

Marathahalli Ring Road,

Bangalore 560103 Karnataka , India

**DETAILS OF THE EXTERNAL GUIDE**

Sunil Chowdary

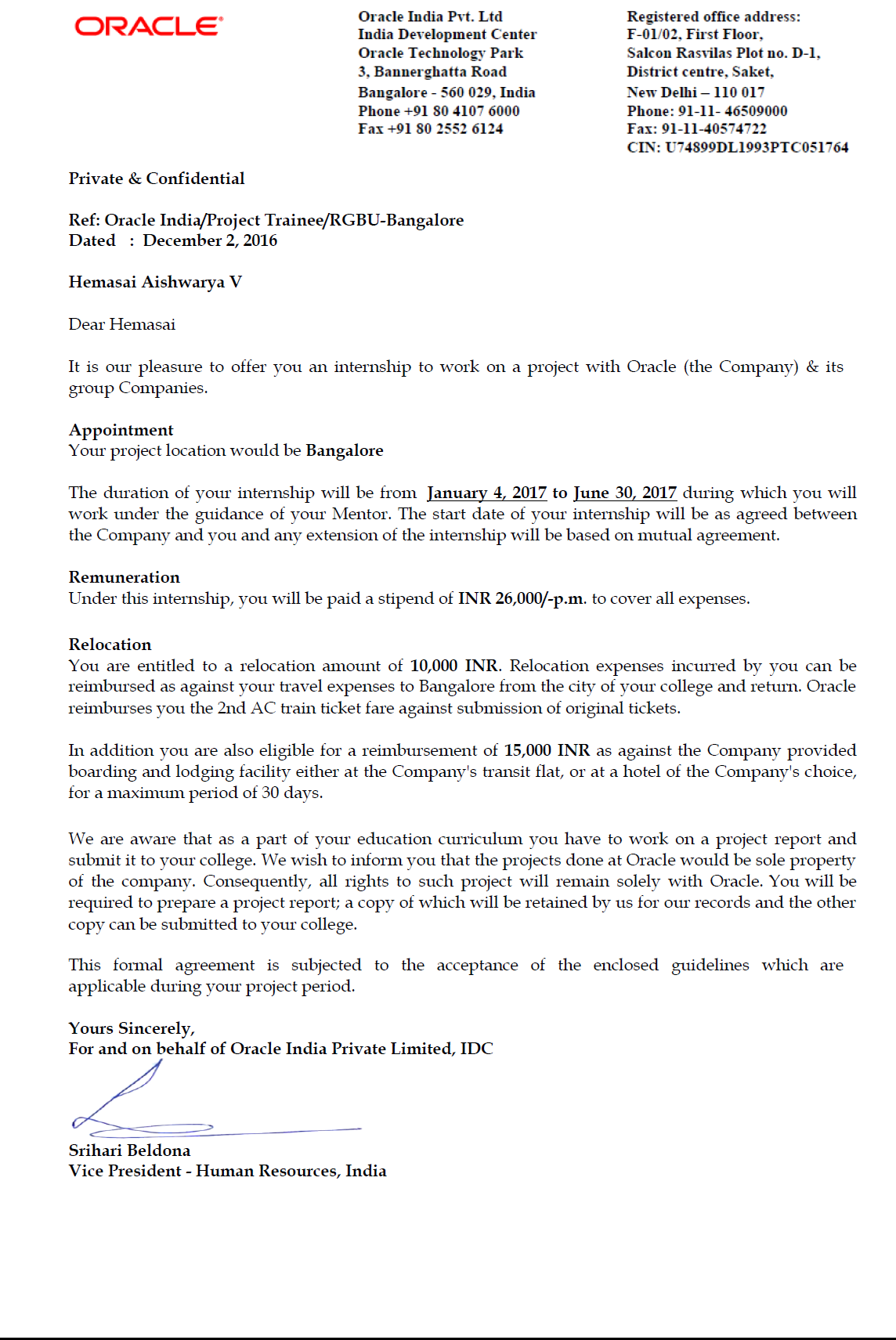
Senior Manager, Software Development

RGBURD ISC Apps Sustaining Eng

[sunil.chowdary@oracle.com](mailto:sunil.chowdary@oracle.com)

Ph No. +918066050782,

+919886353999

** INTERNSHIP OFFER LETTER**

ABSTRACT

Retail Industry is said to be highly competitive and mature among the various industries in any economy. With the increasing standards of living and the online shopping options, Retailers have a difficult task to have an edge over others in their targeted markets. The companies must ensure that they provide desirable products while managing inventory and controlling costs to succeed in the business. Retailers often have stores in various parts of the country and also spread across continents and it is imperative that they would have to carry out varied tasks such as store operations management, warehouse management, demand forecasting, merchandise financial planning, assortment planning and category management in order to optimize and cut down costs and efficiently plan for higher profits. Oracle Retail provides the retailers with innovative solutions in Planning, Management and Forecasting areas in order to help the retailer.

Issues and feedback’s from customers are very usual for a very well established product in a product development company. Our Stores Team works for resolving issues and incorporating the code enhancements into all the products sold.

This project basically deals with clearance optimization engine patch release 14.0.5. This includes upgrading LUX and Weblogic. For the COE product there may be issues or feedbacks raised by the internal organization or from the customers; so after the issue is logged, it needs to be fixed.

BugDB and Internet Explorer Developer tools were used for bug resolution process. Tortoise SVN was used to check in the changes made.

CHAPTER 1

INTRODUCTION

* 1. **GENERAL INTRODUCTION TO THE TOPIC**

The Clearance Optimization Engine (COE) provides the What If RMI interface that Item Planning Configured for COE can access using the RPAS special expression. This allows the IP application to produce in-season price recommendations and forecasts that account for planned promotions and future markdowns in the product life cycle. The forecast includes a sales plan and an optimal price plan. COE produces its recommendations during the weekly model run. The results of the model run are stored in the database. These results can be extracted using the sendback files. They are then available for use by IP.Users have the ability to perform real-time What-If scenarios from within Item Planning and alter plans in order to see the results of those changes. The changes include planning a markdown, changing future prices, changing an order, changing the exit date, changing the salvage value, and changing the sell-through target.

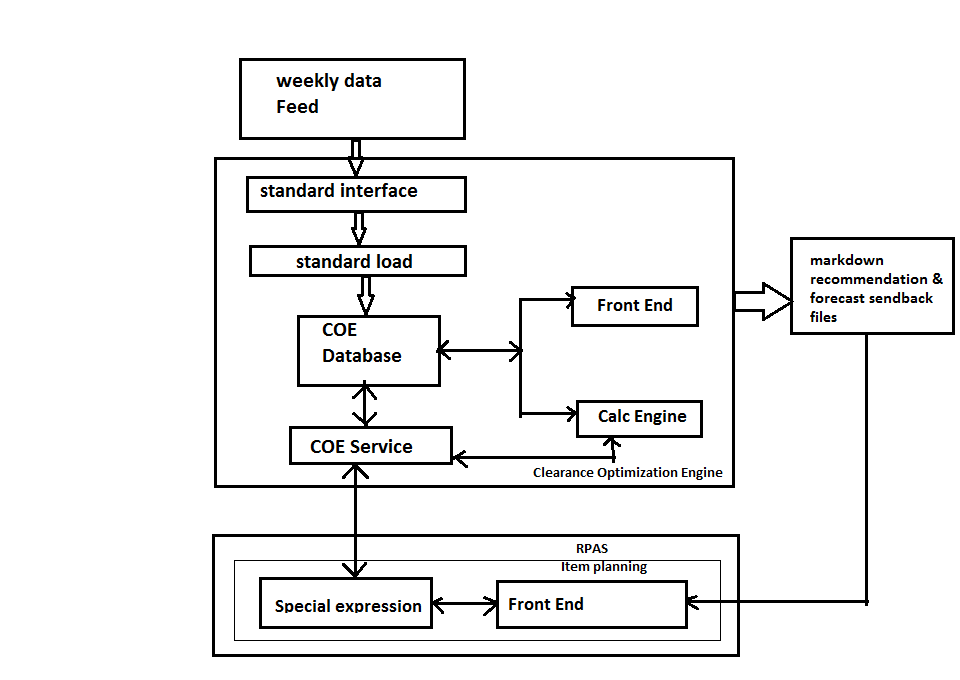


Fig 1.1 Relationship Between COE and Item Planning

**1.2 ORGANIZATION**

Oracle offers the industry's most complete and integrated suite of software applications for insight-driven retailing. Only Oracle provides all the key components like database, middleware, and applications to transform the retail business.

Oracle Retail is the result of Oracle's strategic acquisition of best-of-breed applications, as well as the realization of its long-term vision for the retail sector. These applications provide retailers in key vertical segments including fashion, grocery and hardliners with more insight, further integration, and greater value to drive customer relationships.

The future of retailing requires that business adapts to changing consumer needs and use intelligence to drive repeat shoppers at each channel. With Oracle Retail, retailers can keep pace with their customers and transform the way business is done. Only Oracle allows creating a unified view of the information that delivers economic value to the whole business, from in-store operations to corporate strategy to supply chain and logistical management. Oracle does this by providing accurate, actionable information, fully integrated between store, merchandising, and financial systems. With the newest release of Oracle Retail, Oracle continues to transform the economics of retail by providing innovative leadership in technology and delivering measurable results for retailers of all sizes. Oracle Retail provides a user friendly application environment so that the customers are been provided with an easy working atmosphere.

**1.3 AREA OF COMPUTER SCIENCE**

**1.3.1 Sustenance Engineering**

Product development companies need to continuously look to develop new products / build new versions. Even when a new software product or enterprise business system is operating successfully it can create further challenges and demands. New features are required and new technology must be exploited. Despite this success, budgetary constraints are often placed on the upgrade effort once the initial capital has been expended.

**1.3.2 Software Development**

Software development is the process of computer programming, documenting, testing, and bug fixing involved in creating and maintaining applications and frameworks resulting in a software product. Software development is a process of writing and maintaining the source code.

**1.4 HARDWARE AND SOFTWARE REQUIREMENTS**

**1.4.1 Hardware Requirements:**

* Processor: Intel i5.
* RAM volume: 8GB.
* Hard Drive Volume: 500 GB.

**1.4.2 Software Requirements:**

Operating System: OBI Windows 7.

Browser : Microsoft Internet Explorer 9.0 onwards

###### Oracle JavaScript Extension Toolkit (JET) :

###### Empowers developers by providing a modular open source toolkit based on modern JavaScript, CSS3 and HTML5 design and development principles.

###### Oracle WebLogic Server 12c R2 :

It is the industry's best application server for building and deploying enterprise Java EE applications with support for new features for lowering cost of operations, improving performance, enhancing scalability and supporting the Oracle Applications portfolio.

**Lightweight User Interface (UI) Extensions (LUX):**

It is a collection of UI patterns, components, utilities, templates, and tools for building client-side applications. LUX relies on HTML, CSS, JavaScript, and WAI-ARIA.

CHAPTER 2

PROBLEM DEFINITION

The clearance optimization engine or COE is already developed. It had many base releases and patch releases. Now, We are working on COE Patch release 14.0.5 .For working on COE 14.0.5, we need to use the latest versions of LUX and Weblogic Server. Change in the LUX and Weblogic server versions can also cause some errors. These errors should be rectified and necessary changes should be made to the code for making them compatible.

This is basically done for making COE compatible with the other oracle retail products. This would help the customers to continue using COE as well as the other oracle products.

The other issue is bug fixing.

CHAPTER 3

OBJECTIVES

The project is basically pure research on particular software product and enhances the performance and minimizes the errors in the software. This task needs quick adaptation of software code, its structure, algorithms and performance. Based on your research about the software, one can decide that whether the issue reported by customer is actually a bug or it’s just the actual functionality of the software. In short thorough understanding of software architecture is the first thing which is required.

The other task involves problem finding and solving ability. In this project sustaining engineer has to develop the exact environment that customer has and then perform the test on the product COE. Once problem is found out, the researcher has to open the code, batches or user interface. He/she has to learn and understand each and every aspect of the issue; and finally accomplishes the solution.

The main objectives :

* To contribute in developing the product (COE) 14.0.5 patch release.
* To resolve the issues (bugs/defects) and feedbacks of customers in an efficient and effective way by optimizing the recourses.

The objective of the project is reliability. The organization can provide the software to the customer, which is reliable and robust. The second most important objective is reusability. Generally organization uses the same code which is in the previous versions with little modifications at the time of new version release. So if the old code is error free then the new version will automatically become reliable. Third benefit from this project is income. If customer demands more functionality which is specific to his business requirements, he will demand this functionality from the organization. The project allows the intern to interact with software architects and software researchers. The interns also get the chance to communicate with the end users of the software and can get the feedback of their work

CHAPTER 4

BACKGROUND

**4.1 SUSTAINING ENGINEERING IN CLEARANCE OPTIMIZATION ENGINE**

The work of Sustaining Engineering starts as soon as the software product releases. Job of Stores team is to maintain the product, after it is delivered to the customer. During this maintenance phase, hundreds of bugs are fixed and new functionalities are added to the given product owing to customer requirements. Using Sustaining Engineering we can maintain and enhance existing software and can extend the value of current products. The process of Sustaining Engineering identifies the software solution to help the developer to discover and capture the untapped features of the existing product. This helps Oracle to develop a more new reliable and flawless product for its customers.

Sustaining engineering thus extends the life of a product. The process involves the following steps:-

* Requirement Analysis
* Software solution and implementation
* Source code control and configuration management
* Documentation
* Quality Assurance test plan design and execution
* Formal release procedures

The basic hierarchy of the sustaining engineering is shown below:

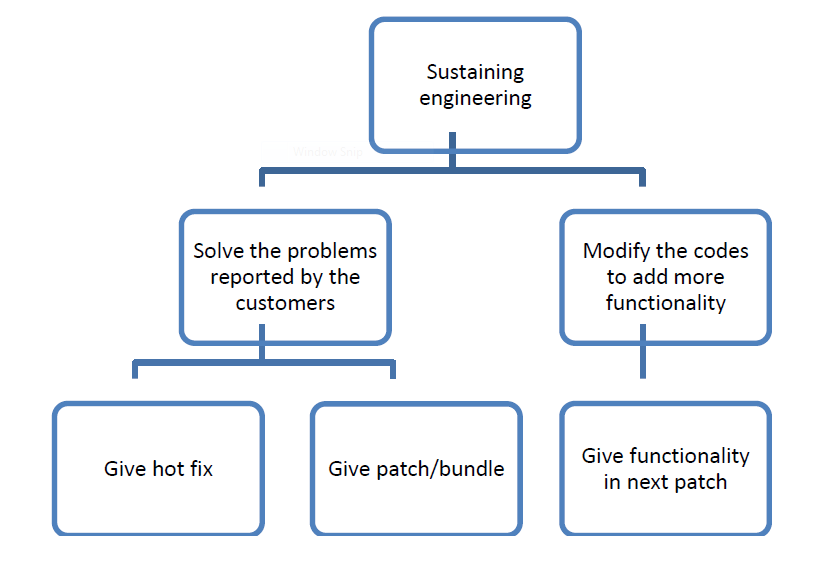


Fig 4.1 Hierarchy of the sustaining engineering

**4.2 LIGHTWEIGHT USER INTERFACE EXTENSIONS (LUX)**

The GBU Lightweight UI Extensions (LUX) provide common, re-usable business functionality and helper utilities (patterns, utility libraries, and UI components) that GBU teams can use to more quickly create client-side, web-standards based applications. GBU LUX is built using the Oracle JET framework, UI components and additional 3rd party open source libraries. LUX is managed / curated by the GBU Architecture Team.

LUX was originally started as the GBU Lightweight UI Framework, focused on providing a common UI framework for use.

Key goals included:

* Reduce duplication of effort across the GBU's by providing a common client side UI framework.
* Use standard web Technology (HTML5, CSS5, Javascript) and well supported open source libraries (Jquery, JqueryUI, Knockout, Requirejs, etc.) rather than reimplementing needed functionality.
* Support for teams to "contribute" their work so that it can be supported and easily

utilized by other teams.

**4.2.1 Reasons For Using LUX**

Since our main requirement was the data grid, LUX grid supports a range of column types and features such as framing with groups, sorting, menus

and inline editing.

Features available in the LUX Data Grid but not available in the OJET Data Grid

include:

* Editing of data within grid cells (eg. Text, Numerics, time / date, lists, checkboxes,etc.)
* Row manipulation (cut / past, copy, add new, delete, copy multiple, etc.).
* Row and column Drag and Drop
* Column Grouping
* Filter and row Sorting
* Row and Cell Level events (select, edit, save, move, etc.)

Find and Filter (including type down)

**4.3 WEBLOGIC SERVER 12C**

Oracle WebLogic Server is a scalable, enterprise-ready Java Platform, Enterprise Edition (Java EE) application server. The WebLogic Server infrastructure supports the deployment of many types of distributed applications and is an ideal foundation for building applications based on Service Oriented Architectures (SOA). SOA is a design methodology aimed at maximizing the reuse of application services.

The WebLogic Server complete implementation of the Java EE 7.0 specification provides a standard set of APIs for creating distributed Java applications that can access a wide variety of services, such as databases, messaging services, and connections to external enterprise systems. End-user clients access these applications using Web browser clients or Java clients. In addition to the Java EE implementation, WebLogic Server enables enterprises to deploy mission-critical applications in a robust, secure, highly available, and scalable environment. These features allow enterprises to configure clusters of WebLogic Server instances to distribute load, and provide extra capacity in case of hardware or other failures. New diagnostic tools allow system administrators to monitor and tune the performance of deployed applications and the WebLogic Server environment itself. You can also configure WebLogic Server to monitor and tune application throughput automatically without human intervention. Extensive security features protect access to services, keep enterprise data secure, and prevent malicious attacks.

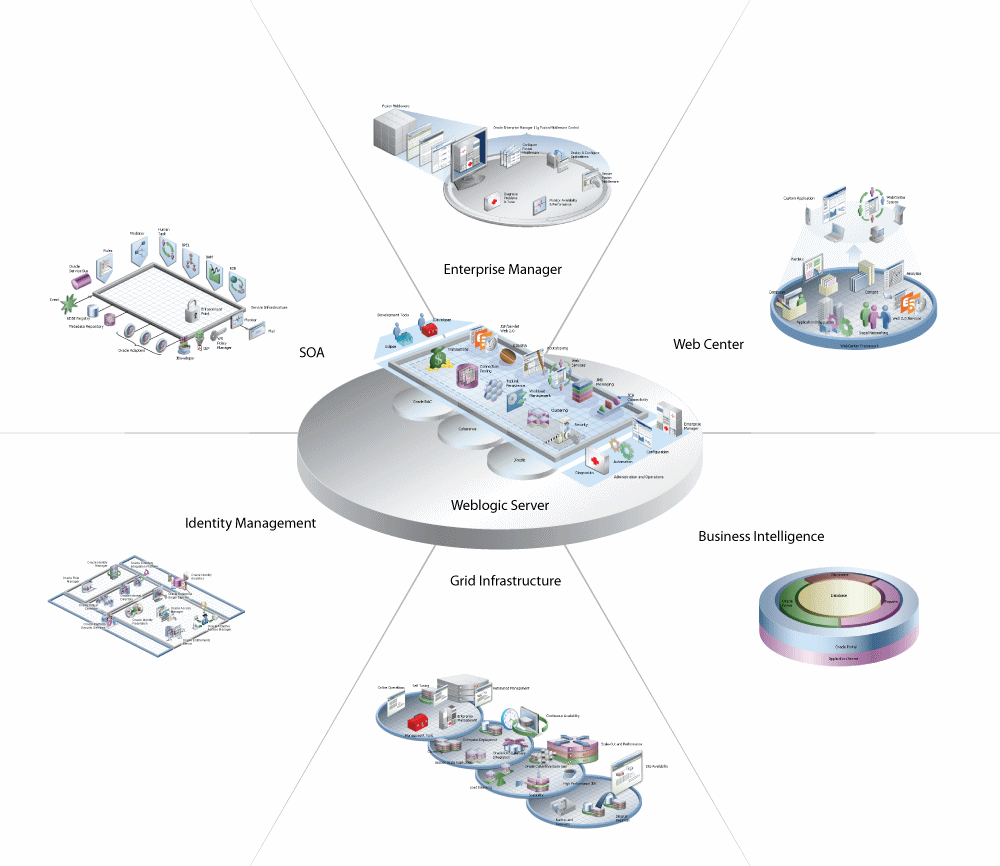


Fig 4.2 Oracle Fusion Middleware Overview

**4.4 ORACLE JET**

Oracle JavaScript Extension Toolkit (JET) empowers developers by providing a modular open source toolkit based on modern JavaScript, CSS3 and HTML5 design and development principles. Oracle JET is targeted at intermediate to advanced JavaScript developers working on client-side applications. It's a collection of open source JavaScript libraries along with a set of Oracle contributed JavaScript libraries that make it as simple and efficient as possible to build applications that consume and interact with Oracle products and services, especially Oracle Cloud services.

CHAPTER 5

METHODOLOGY

In this chapter, how the work is accomplished in Stores team is explained in detail. This involves the planning and execution of software projects for the purpose of existing product enhancement including Bug Fixes. There are different languages and technologies used at each and every level of the product. If there are any issues for this sold product, then that should be resolved and incorporated in all the products, which are sold. Such product maintenance projects are very frequently in competition with new product development for internal engineering resources.

We have followed the following stages of sustaining engineering in resolving the issues:

**Stage I: Logging of issues and feedbacks by customer:**

After the COE product is sold, there is a marketing team in the organization, which collects feedbacks and issues faced by the customers. The information might be passed through phone, email or by person. The feedback may be to add new functionality to the COE or remove existing functionality or modify the existing functionality. After collecting the issues they pass it on to strategizing team.

**Stage II: Segregating the issues which are to be solved:**

The strategizing team analyses the issues and feedback of customers and issues faced, and then prioritizes the problems based on the urgency, feasibility and many other criteria. The urgency in retail domain means, the severity of the issue faced by the retailer (customer) in terms of data loss or any other business impact.

**Stage III: Segregating the issues based on required date:**

The GRID team then decides which bugs or functionalities should be fixed early than the next version release date of the product. Based on the required date, the bugs or functionalities fall in two buckets:

(i) Before next version release date.

(ii) Not before next version release date.

**Stage IV: Segregation issues based on major functionality:**

In the sustenance engineering team, the specific bugs would be allocated to specific teams based on the functionality, which is getting impacted by the bug. So, one of the team member would be assigned this bug or functionality change.

**Stage V: Fixing the issue or adding functionality:**

1. **Analysis:** As a part of the bug or feedback passed to us by GRID people, they send us relevant information and documents regarding the issue. These are called bug documents. Then we analyze the issue, which means first we completely understand the functionalities of product, related to the bug. We can find the information about all the functionalities from the well-versed internal oracle documents. After getting an in-depth overview the first step we do is to try to recreate the issue in our local COE product and try to see whether the issue persists at our end also. If there is something happening other than what they have mentioned, then we would do some research whether it is behaving as per the expected functionality or deviating from it.
2. **Design:** After analyzing the issue you would come to know whether or not the design of the system is going to be affected to fix the issue. Or there might be a need to design a new module to fix the issue. In any case then we should talk to the design architect (DA) and confirm whether the design change is really required, because we never know the design change might not be useful for all the customers. Any design change or code fix should be done keeping all the customers in mind, not for an individual customer. If the change is specific to any one customer then the team decides not to go ahead for fixing it
3. **Development/Code Enhancement:** Here we need to get the modules of the

COE,which are going to be changed in case if the fix does not require creating new modules.

1. **Debugging the Code:** This is basically done using the internet explorer developer

tools.

**Tools used:**

The F12 tools is a suite of on-demand tools that is built into every installation of Windows Internet Explorer. F12 tools is available anytime, on any page, and enables website developers to quickly debug JavaScript, HTML, and Cascading Style Sheets(CSS), as well track down performance problems on a webpage or network.

The several features provided are:

## The DOM Explorer tool (CTRL+1)

The DOM Explorer tool shows the structure of your webpage as it's being rendered in the browser and makes it possible to edit your HTML and styles in a live page. You can do this without having to edit and reload your sources, so you can quickly solve display issues or experiment with new ideas.

## The Debugger tool (CTRL+3)

You use the Debugger tool to examine what your code is doing, when it's doing it, and how it's doing it. Pause code in mid-execution, step through it line-by-line, and watch the state of variables and objects at each step.

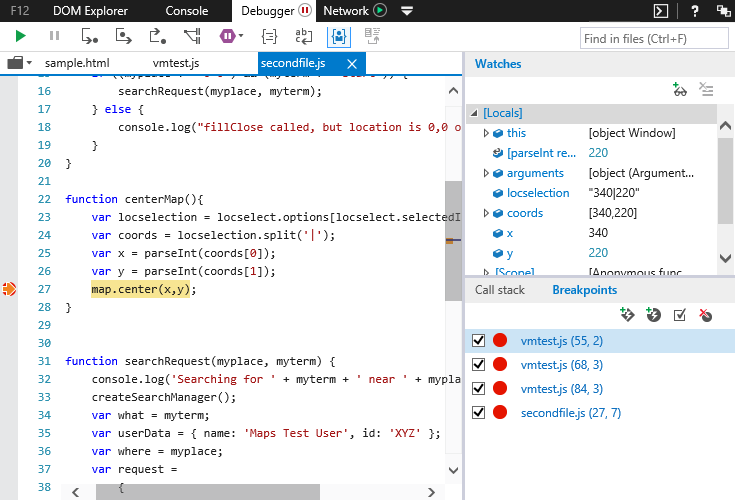


Fig 5.1 Debugger Tool

## The Console tool (CTRL+2)

The Console tool provides a way to interact with your running code:

* Change variable values or inject code into a live site with the Console's command line.
* Use the Console Debugging API to send out debug information.

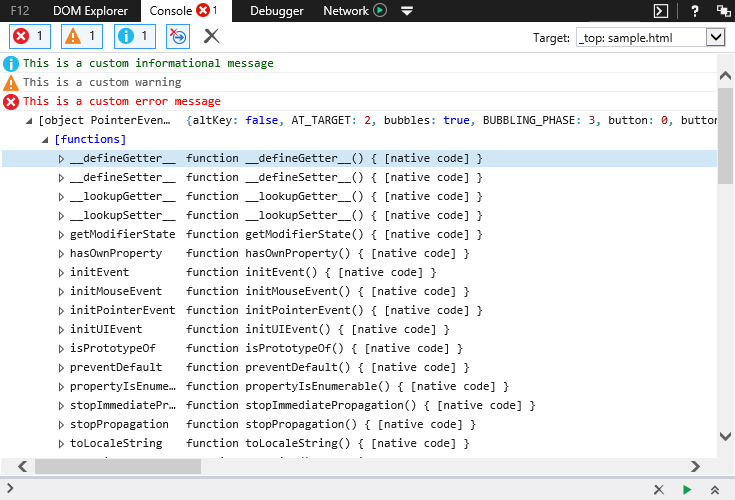


Fig 5.2 Console Tool

CHAPTER 6

IMPLEMENTATION DETAILS

**6.1 COE PATCH RELEASE 14.0.5**

* A remote server is created for developing COE 14.0.5
* Cygwin is used to run the installer**.** It is a Unix-like environment and command-line interface for Microsoft Windows. Cygwin provides native integration of Windows-based applications, data, and other system resources with applications, software tools, and data of the Unix-like environment. Thus it is possible to launch Windows applications from the Cygwin environment, as well as to use Cygwin tools and applications within the Windows operating context.
* LUX 2.0 is upgraded to 2.3
* Upgrading weblogic 11g to 12c is still in progress
* 1st code drop is already done

**6.1.1 Code Inspection**

Code inspection for a model is carried out after the module is successfully compiled

and the all the syntax errors have been eliminated. Code inspection are extremely

cost- effective strategies for reduction in coding errors and to produce high quality code.

The aim of code inspection is to discover some common types of errors caused due

to oversight and improper programming.

Following is a list of some classical programming errors which can be checked during code inspection:

* Improper storage allocation and deallocation
* Improper modification of loop variables.
* Use of incorrect logical operators or incorrect precedence among operators.

**Code**

When embarking on a change, we took a copy of the current code base on which to

work. As other developers submit changed code to the source code repository, this

copy gradually ceases to reflect the repository code. Not only can the existing code

base change, but new code can be added as well as new libraries, and other resources

that create dependencies, and potential conflicts.

The longer a branch of code remains checked out, the greater the risk of multiple integration conflicts and failures when the developer branch is reintegrated into the mainline. When developers submit code to the repository they must first update their code to reflect the changes in the repository since they took their copy. The more changes

the repository contains, the more work developers must do before submitting their

own changes. Eventually, the repository may become so different from the developers' baselines that they enter what is sometimes referred to as "merge hell", or "integration hell", where the time it takes to integrate exceeds the time it took to make their original changes. Continuous integration involves integrating early and often, so as to avoid the pitfalls of "integration hell". The practice aims to reduce rework and thus reduce cost and

time. We used TortoiseSVN, it is a free software released under the GNU General Public

License. After all the code is written and the other javascripts are edited to get the value

from the corresponding grid we do the following:

config.js: contains all the paths for the used libraries.

**Define block:**

It defines a well-scoped object that avoids polluting the global namespace. It can explicitly list its dependencies and get a handle on those dependencies without needing to refer to global objects, but instead receive the dependencies as arguments to the function that defines the module. The RequireJS syntax for modules allows them to be loaded as fast as possible, even out of order, but evaluated in the correct dependency order, and since global variables are not created, it makes it possible to load multiple versions of a module in a page.

define([lib1, lib2, … , libn /\*(listed in config)\*/ ], function(obj1,obj2, …, objn){

Code to make the grid changes visible instantaneously visible using knockout;

Once a grid is configured, it must be loaded with data. This can be done by statically linking to an existing JSON file, passing in dynamically generated JSON, or by calling a REST service that

provides JSON;

Event handlers like link clicks, updated and all;

Returns the object;

})

**Require block:**

Uses the define block.

require([knockout and the above one], functions(ko, loader) {

bindings are applied;

data is loaded;

})

**6.1.2 Build**

There is a number of steps required to transform the source

into a deployable and useable software solution. The following is a hypothetical build

process you might use with a simple software system

* Get the source. You may need to download or fetch the source from a source code
* repository. For this, you might need to know the tag or version of the source code you want to build.
* Prepare a build area.
* Configure the build. In this step, you will determine what optional components can be built based on the current environment. You might want to set build numbers and version numbers to be included in the build.
* Validate the source code.
* Compile the source code
* Build the compiled code into libraries potentially including non-code resources such as properties, images and sound files.
* Run the system's tests to validate the build.
* Build the documentation for the software. This may range from something as simple as collecting text files up to processing content through some form of publishing system to produce the documentation in its final form.
* Package up all of the components of the software \_ code, resources, images,

documentation, etc. \_ into a deployable package. You might need to produce

several packages in different formats for different target users.

* Deploy the software to some standard location for use or distribution.

**6.2 BUG FIXES**

|  |  |  |
| --- | --- | --- |
| **BUG ID** | **ISSUE** | **STATUS** |
| 1 | Version is displayed as 14.0.4 instead of 14.0.5 | Corrected |
| 2 | The application wasn’t successfully running | Corrected |
| 3 | &gt; is appearing instead of > in a table | Corrected |
| 4 | Scroll is half visible when the grid is in filter mode | Corrected |
| 5 | Last checkbox is not getting selected | Corrected |
| 6 | Secondary window dialogs are not resizing | Partially corrected |
| 7 | Some filtering operations are not working as expected | Partially corrected |
| 8 | Horizontal scroll is half visible | Corrected |

Table 6.1 Solved bugs

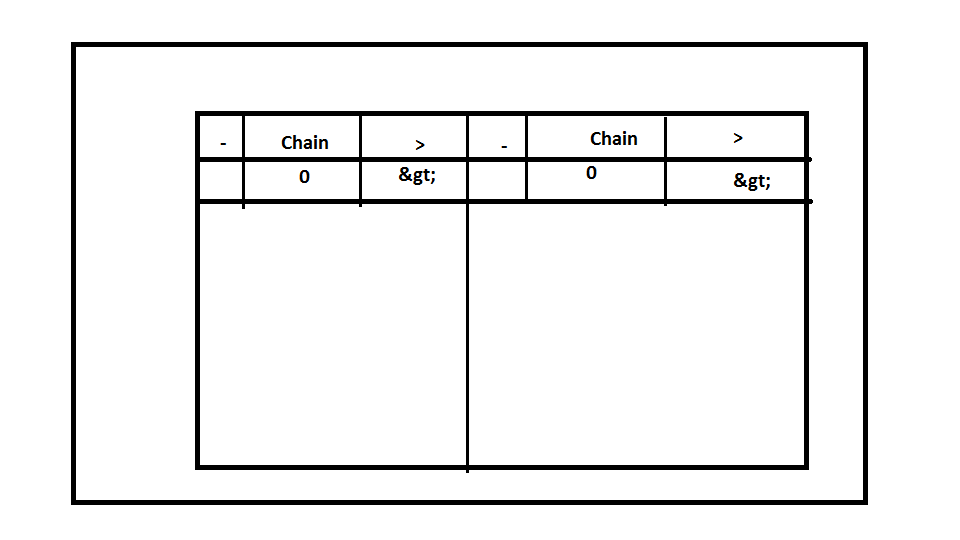


Fig 6.1 Bug3: In COE 14.0.5

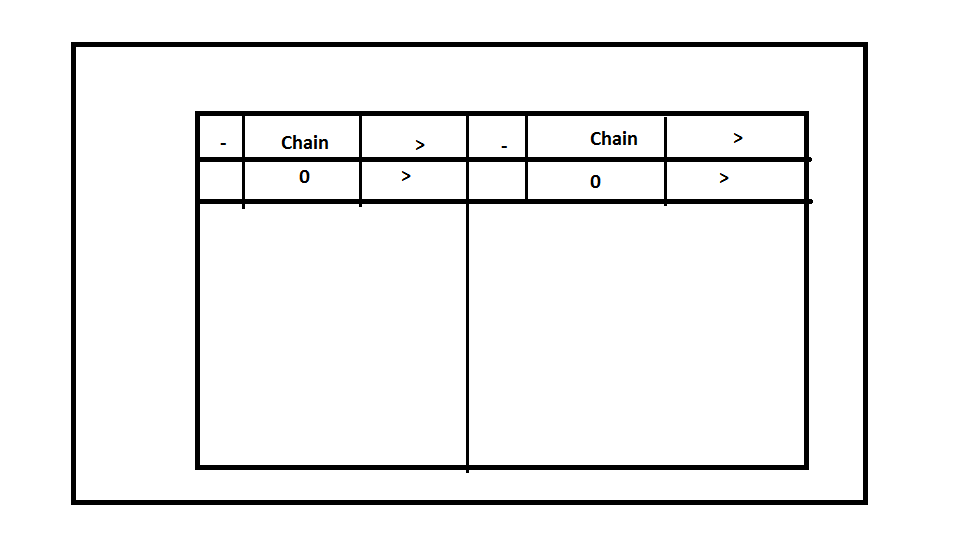


Fig 6.2 Bug3: In COE 14.0.4

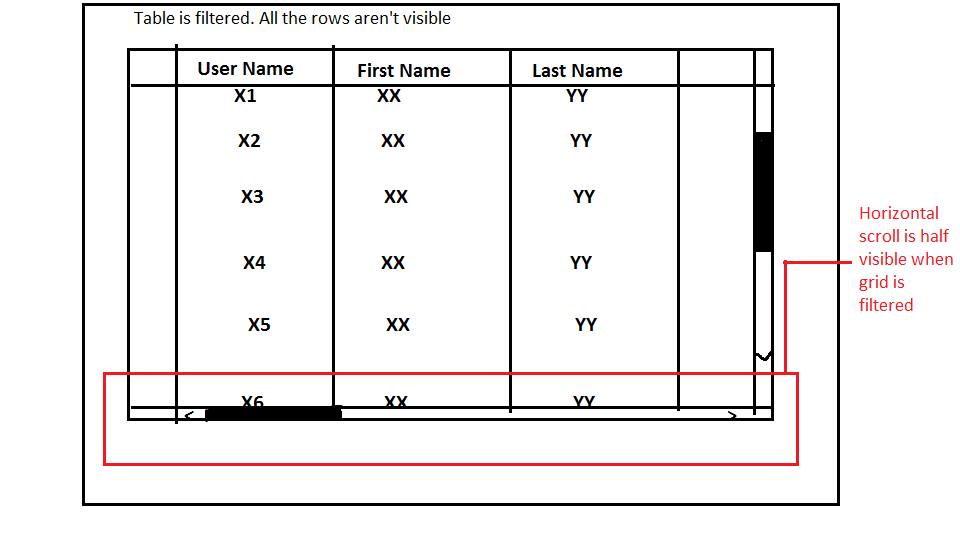


Fig 6.3 Bug4

CHAPTER 7

PROGRESS AND REMAINING WORK

**7.1 COMPLETED WORK**

* Few bugs are fixed and closed
* LUX 2.0 is upgraded to 2.3 in COE 14.0.5 Patch release.
* 1st code drop is done

**7.2 REMAINING WORK**

|  |  |
| --- | --- |
| **Bug id** | **Issue** |
| 8 | Wrong column count is displayed in print and export |
| 9 | Vertical scroll is missing in the filter table |
|  | Other newly logged bugs |

Table 7.1 Unsolved bugs

* 2nd and 3rd code drop is pending.
* Weblogic server is not completely upgraded

CHAPTER 8

REFERENCES

[1] "Oracle Retail Clearance Optimization Engine Documentation Library".

Docs.oracle.com. N.p., 2017. Web. 6 Apr. 2017.

[2] "VPAT - GBU Lightweight UI Extensions (LUX) 1.1.1". Oracle.com. N.p., 2017.

Web. 6 Apr. 2017.

[3] "Understanding Administration Console Extensions". Docs.oracle.com. N.p., 2017.

Web. 6 Apr. 2017.

[4] "F12 Devtools Guide". Docs.microsoft.com. N.p., 2017. Web. 6 Apr. 2017.

[5] "Oracle JET 3.0.0". Docs.oracle.com. N.p., 2017. Web. 6 Apr. 2017.