

VERITAS™

March 29, 2017

Statement of Employment for Laxmi Shinganwade

To Whom It May Concern:

This letter is to confirm that Laxmi Shinganwade is a Full time employee of Veritas Software Technologies India Pvt Ltd hired on January 16, 2017 and is currently a Intern - Master's Level. The employee's length of service may include time with acquired companies.

Please note, this statement is issued on the employee's request.

In accordance with our normal practice, this statement is given without acceptance of any liability on the part of Veritas Software Technologies India Pvt Ltd. It is not our practice to comment further in employment statements.

Sincerely,



Leucadia Milly Sandeep
India Human Resources
Veritas Software Technologies India Private

Veritas Software Technologies India Private Limited, EON Zone, 0-05 Floor, Wing-4, Cluster A, Plot No. 1, Survey No. 77,
MIDC Knowledge Park, Kharadi, Pune – 411014, India, Corporate Identification Number: U72200PN2015FTC154978

Acknowledgment

I present with pride and pleasure the project report on “Release Management” aimed to the supplement attachment has required under the regulation of the “SavitribaiPhule Pune University”. I

would like to express my sincere gratitude to Mrs.Yamini Nagpal, VERITAS Technologies LLC India Private Limited, who has helped in making this project. I am highly indebted to the project guide Dr.Minakshi More, MCA Coordinator, I.M.C.C. and Ms.Shruti M for their guidance and constant inspection in each and every stage of project. I wish to offer most sincere thanks to Dr. Santosh Deshpande, HOD, Computer Department, IMCC for encouraging and providing all the facilities. I wish to offer sincere thanks to Dr.ManasiBhate, TPH for providing the necessary co-operation. Particular thanks to Dr. V. H. Inamdar, Director, I.M.C.C. who allotted us enough time for the project. Finally, I am extremely thankful to the teaching and non-teaching staff for their kind and whole co-operation.

Laxmi D. Shinganwade

INDEX

Sr. No	Title	Page No.
1	INTRODUCTION	
1.1	Company Profile	1
1.2	Existing System and Need for System	4
1.3	Scope of Work	5
1.4	Operating environment	10
1.5	Detail Description of technology used	11
2	PROPOSED SYSTEM	
2.1	Proposed System	17
2.2	Objective of System	19
2.3	User Requirements	22
3	ANALYSIS AND DESIGN	
3.1	Object Diagram	23
3.2	Class Diagram	24
3.3	Use Case Diagram	25
3.4	Activity Diagram	30
3.5	Sequence Diagram	35
3.6	Entity Relationship Diagram	39
3.7	Module Hierarchy Diagram	40
3.8	Component Diagram	41

3.9	Deployment Diagram	41
3.10	Module Specification	42
3.11	User Interface Design	44
3.12	Data Dictionary	49
3.13	Table Specification	52
3.14	Test Procedures and Implementations	55
4	USER MANUAL	
4.1	User Manual	65
4.2	Operational Manual	67
4.3	Flow Charts	69
	Drawbacks and Limitations	71
	Proposed Enhancement	72
	Conclusion	74
	Bibliography	
	Annexure 1(User Interface Screens)	
	Annexure 2(Output Reports With Data)	
	Annexure 3(Sample Program Code)	

CHAPTER 1
INTRODUCTION

1.1 Company Profile



Veritas Technologies LLC develops and delivers backup and recovery, business continuity, information governance, and storage management solutions worldwide. The company provides backup and recovery products comprising system recovery solutions for protecting desktops, laptops, servers, and virtual machines; and NetBackup appliances that provide backup and recovery for data centers, remote offices, and virtual environments. Its business continuity products include InfoScale Enterprise, which provides business continuity and software-defined storage for critical services across data centers; Backup Exec, a backup and recovery solution for mid-sized organizations; and Disaster Recovery Orchestrator that offers disaster recovery for business applications.

The company's information governance products include Data Insight, a file analysis solution that provides the tracking and reporting necessary to deliver organizational accountability for file usage and security. It also provides education services; business critical services; consulting services; managed services that include monitoring, management, and support of backup and archiving environments; and appliance installation and custom configuration services. It offers unstructured data growth, multi-vendor hybrid cloud, healthcare, and government solutions. Its software and hardware products enable Fortune 500 companies to protect, identify, and manage data within environments ranging from traditional data centers to private, public, and hybrid clouds. The company was incorporated in 2015 and is based in Mountain View, California.

The exponential growth of data and the resources needed to manage it is one of the most pressing issues facing business today. And it's not just the amount of data. It's where it lives and how it travels between private clouds, public clouds and back to on

premises. In these increasingly complex IT environments, it's important to focus on what's constant: the data.

Every one of our information management solutions – from business continuity to back up and recovery to software defined storage and information governance – is designed around the principle that information is more important than infrastructure. Veritas has the privilege to help the world's organizations - including 86% of the global Fortune 500 - collect, protect, analyze and optimize their data, even in the most demanding environments.

Chief Executive Officer: Bill Coleman is the Chief Executive Officer at Veritas. Coleman is an industry veteran who has spent 25 years running prominent Silicon Valley companies including his years as the founder/CEO of BEA Systems.

1.2 Existing System and Need for System

A poorly understood and underdeveloped part of the software process is software release management, which is the process through which software is made available to and obtained by its users.

We are having issues like - it takes too much time for the prod deployment(even it is minor), errors are found at the last step, There is no central team which has control over new changes and releases, no proper tools etc.

In this project we identify the issues encountered in software release management, and present an initial set of requirements for a software release management. We then describe a prototype of such a tool that supports both developers and users in the software release management process.

The Need of Release Management is for -

- Monitoring ERQs in a particular release
- Effort tracking
- Defect tracking
- Reporting

1.3 Scope of Work

Release management undertakes the planning, design, build, configuration and testing of requirements to create a set of release components for a live environment.

Release management builds on deployment management, which looks at how to move a change from one organization to another. Whether from a developer sandbox to integration testing or from user acceptance testing (UAT) to production, release management should be implemented alongside your deployment management process.

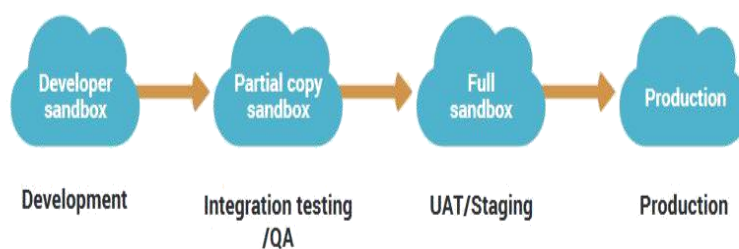
Release management can help companies meet those requirements in key areas such as user access, data availability, and increased visibility across departments.

When working with complex changes or multiple development environments, basic release management is usually a requirement due to the need for source control and integration testing. Reintegrating changes back into the production organization adds complexity to the development process due to the continually

moving goalposts of the production environment, and many businesses begin to implement some sort of formal process at this level.

With projects of this size, properly planned release management provides a way to track changes as they're moved between organizations, create a clear audit trail, and reduce conflicts. Without proper planning, however, release management can become more of a hindrance than a benefit, slowing the development process down and creating choke points for project progress.

Release management provides a high-level overview of organizational structure to best achieve successful deployments which meet users' needs, and also integrates simple deployment management into the process.



Developer sandbox –

Developer sandboxes are isolated organizations which copy metadata (but not production data) into a different environment for coding and testing.

Developer sandboxes should be used for all development work, and are completely removed from the production environment. Each team member should have access to a copy of the metadata from the production environment, and their own independent developer sandbox in which they can make their changes.

In larger projects, developers may maintain multiple environments for this purpose, and developer sandboxes should be linked to a source control system to allow easy promotion to the partial copy sandbox.

Partial copy sandbox-

Partial copy sandboxes copy both metadata and some production data, and have a larger amount of storage space to work with.

Once development work on a feature is complete, it should be checked in to your source control system and deployed to your

partial copy sandbox for integration or QA testing. In large teams or projects, a process of continuous integration, whereby committed changes are automatically built and tested, should be applied between developer sandboxes and partial copy sandboxes for fast and automated functional testing of features. This will allow for rapid iteration and bug fixing on changes and minimize the risk of clashes with other developers' environments. Having passed testing, features should be checked against project goals to ensure they're meeting customer needs before being promoted to UAT.

Full sandbox-

Full sandboxes copy the whole production organization and all data. They're useful for coding and testing changes, and for training.

Before beginning UAT, the full sandbox organization should be cloned back from production. This allows new features to be tested in relation to your entire production environment, minimizing the risks of unexpected errors or unintended effects on other aspects of the organization

Production-

The production environment is live and has users accessing data.

Before any feature is promoted to production, it should be functional (integration testing), meet the needs of the users (UAT) and not cause any disruption to the production environment (tested with real data).

1.4 Operating Environment –

Hardware and Software

HARDWARE REQUIREMENTS

- Core I3 3rd Gen or above
- 2GB RAM or above
- 500 GB hard disk or above

SOFTWARE REQUIREMENTS

- Windows /Mac OS
- Web Browser (IE/Chrome/Firefox/Safari)
- Good Internet Connection

1.5 Detail Description of Technology used

❖ Salesforce

The Salesforce.com is a secure cloud based CRM system.

A CRM (Customer Relationship Management) system is a business tool that allows you to manage your customers, partners and prospects information all in one place.

All Salesforce products run entirely in the cloud so there's no expensive setup costs, no maintenance, no storage needed and employees can work from any device with an internet connection – Smartphone, tablet or laptop.

Salesforce consists of various standard objects like account, contacts, opportunity, case, lead etc which are essential for CRM application.

Salesforce even has an option to build custom objects which helps admin or developer to build objects as per user or business requirements.

Salesforce has inbuilt features like workflow rules , approval process, schema builder etc which helps easily develop the functionality of the system.

It has various features like data loader, import wizard, export wizard etc which helps you easily import records from CSV file into your Salesforce system or export your record form Salesforce into CSV file.



Apex:

Apex is a proprietary language which has been developed by Salesforce.com

Apex is a programming language that uses Java-like syntax and acts like database stored procedures. Apex enables developers to

add business logic to system events, such as button clicks, updates of related records, and Visualforce pages. As a language,

Apex is:

Hosted—Apex is saved, compiled, and executed on the server—the Force.com platform.

Automatically upgradeable—Because compiled code is stored as metadata in the platform, Apex is automatically upgraded as part of Salesforce releases.

Object oriented—Apex supports classes, interfaces, and inheritance.

Strongly typed—Apex validates references to objects at compile time.

Multitenant aware—Because Apex runs in a multitenant platform, it guards closely against runaway code by enforcing limits, which prevent code from monopolizing shared resources.

Integrated with the database—It is straightforward to access and manipulate records. Apex provides direct access to records and their fields, and provides statements and query languages to manipulate those records.

Data focused—Apex provides transactional access to the database, allowing you to roll back operations.

Easy to use—Apex is based on familiar Java idioms.

Easy to test—Apex provides built-in support for unit test creation, execution, and code coverage. Salesforce ensures that all custom Apex code works as expected by executing all unit tests prior to any platform upgrades.

Versioned—Custom Apex code can be saved against different versions of the API.

Visual Force:-

Visualforce is a web development framework that enables developers to build sophisticated, custom user interfaces for mobile and desktop apps that can be hosted on the Force.com platform.

You can use Visualforce to build apps with user interfaces that look like the standard interface provided by Force.com, as well as your own completely custom interface.

Visualforce enables developers to extend Salesforce's built-in features, replace them with new functionality, and build completely new apps. Use powerful built-in standard controller

features, or write your own custom business logic in Apex. You can build functionality for your own organization, or create apps for sale in the AppExchange.

Visualforce app development is familiar to anyone who has built web apps. Developers create Visualforce pages by composing components, HTML, and optional styling elements.

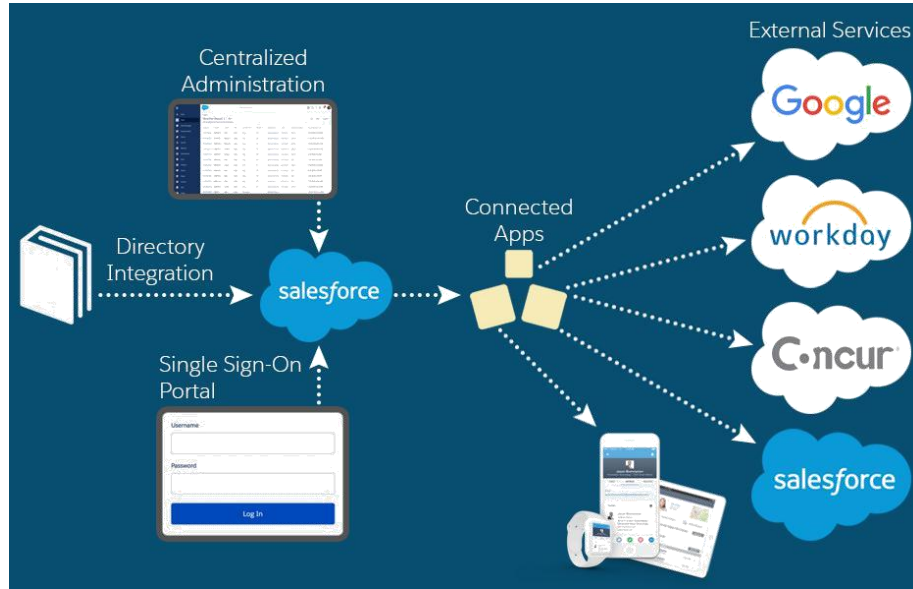
Visualforce can integrate with any standard web technology or JavaScript framework to allow for a more animated and rich user interface.

Each page is accessible by a unique URL.

When someone accesses a page the server performs any data processing required by the page, renders the page into HTML, and returns the results to the browser for display.

SOQL:-

Salesforce provides the Salesforce Object Query Language, or SOQL in short, that you can use to read saved records. SOQL is similar to the standard SQL language but is customized for the Force.com platform.



CHAPTER 2

PROPOSED SYSTEM

2.1 Proposed System

Sometimes, large companies have multiple different release management processes even for the same org. For one division, they may be using one set of processes and for another division, they could be having a totally different set of processes. It is necessary to streamline and standardize the processes across the various sets of users since they are using the same foundation, platform and resources to enable their business.

Release management has been implemented on salesforce platform..

We have monthly releases that consist of ERQs. Those ERQs are the requirements given by the business. If there is any requirement which is impacting multiple areas then that ERQ needs to be approved by ops Council. Ops council is group where all business users and managers decide whether the ERQ should be worked upon or not.

Release management is capable of managing the Release/Requirements/Test cases /Defects /Reports and Dashboards.

2.2 Objective of System

The objective of Release Management is to have consistent set of repeatable, reliable and resource-independent processes to achieve optimal business value and optimize resource utilization. Given that most organizations are focused on running shorter, high-yield business projects, it is imperative that the application delivery value chain is optimized to ensure that there are no bottlenecks in delivering the business value.

The main objective of Release Management is to track the changes between the various environments such as development environment, QA environment, UAT environment and production environment.

The system objectives are

1. The System is developed in manner that it is user friendly.
2. Required help and validation are provided at different levels.
3. To facilitate easy retrieval of data & information.
4. To reduce manual work.
5. The system is easy to handle.
6. To enhance speed of work.

7. Increasing impact of information processing for organizational decision making.
8. Information revolution and the overall development scenario.
9. Use of information processing to increase the profit.

Advantages –

Efficiency – Reducing the time it takes to find, fix, and deliver reliable, quality software.

Productivity – Teams can focus on more important items then trying to figure out what version or steps have caused an outage.

Reducing risk –The ability to identify the what, when, where, and how for a release.

Collaboration between teams – Opening a platform for the team to discuss blocking issues by using definite and transparent processes.

Configuration management – Knowing what environment setting, application requirements, and dependences exist in the production, test, and development environments.

Track Changes – Release Management is helpful to track the changes between the various environments such as development

environment, QA environment, UAT environment and
production environment.

Maintain Confidentiality of company's data

Easy access

2.3 User Requirement

Confidential project requirement

Flexible

Customize easily

Usability

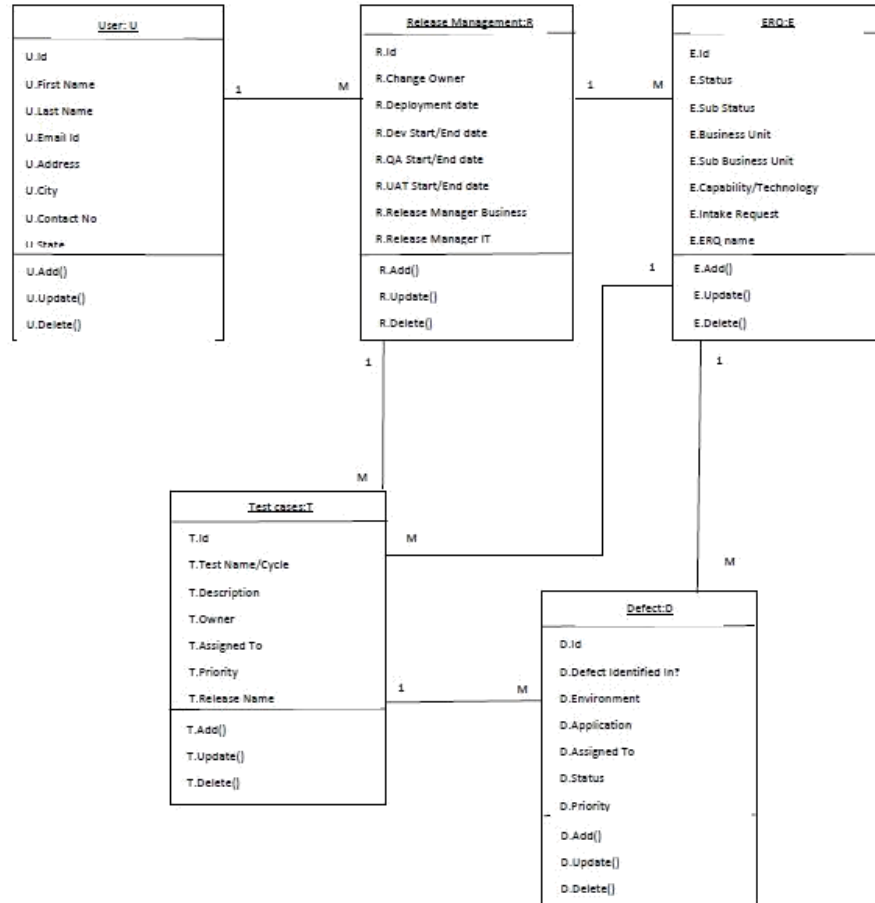
Reliability

Security

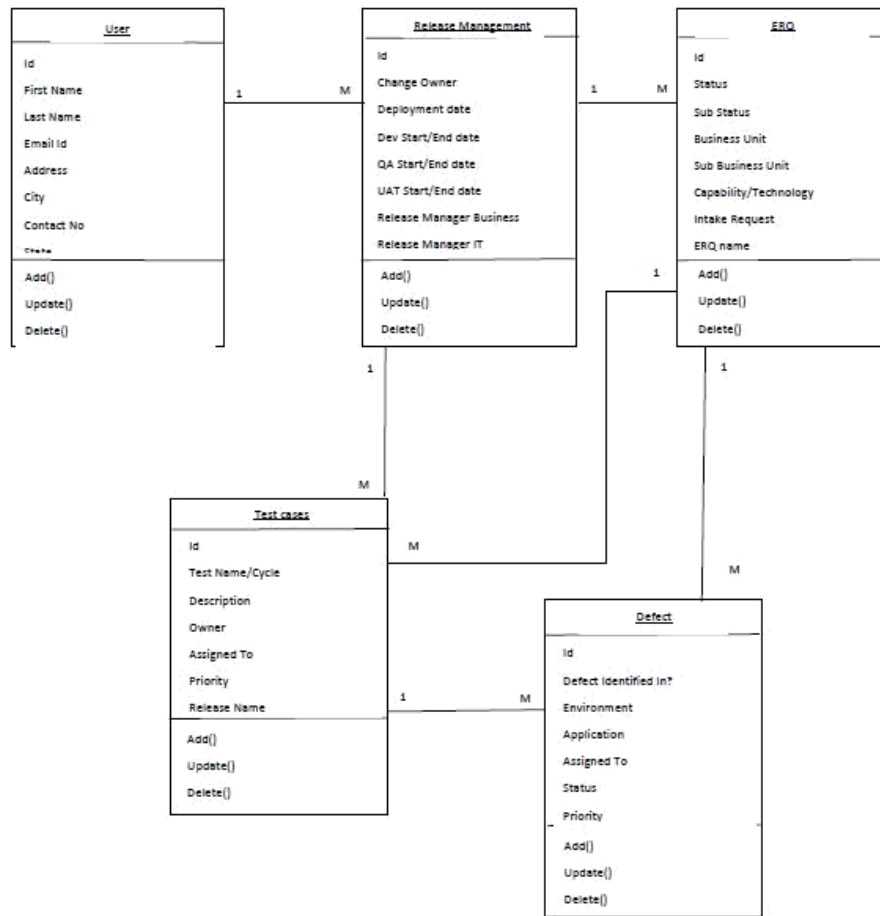
CHAPTER 3

ANALYSIS AND DESIGN

3.1 Object Diagram

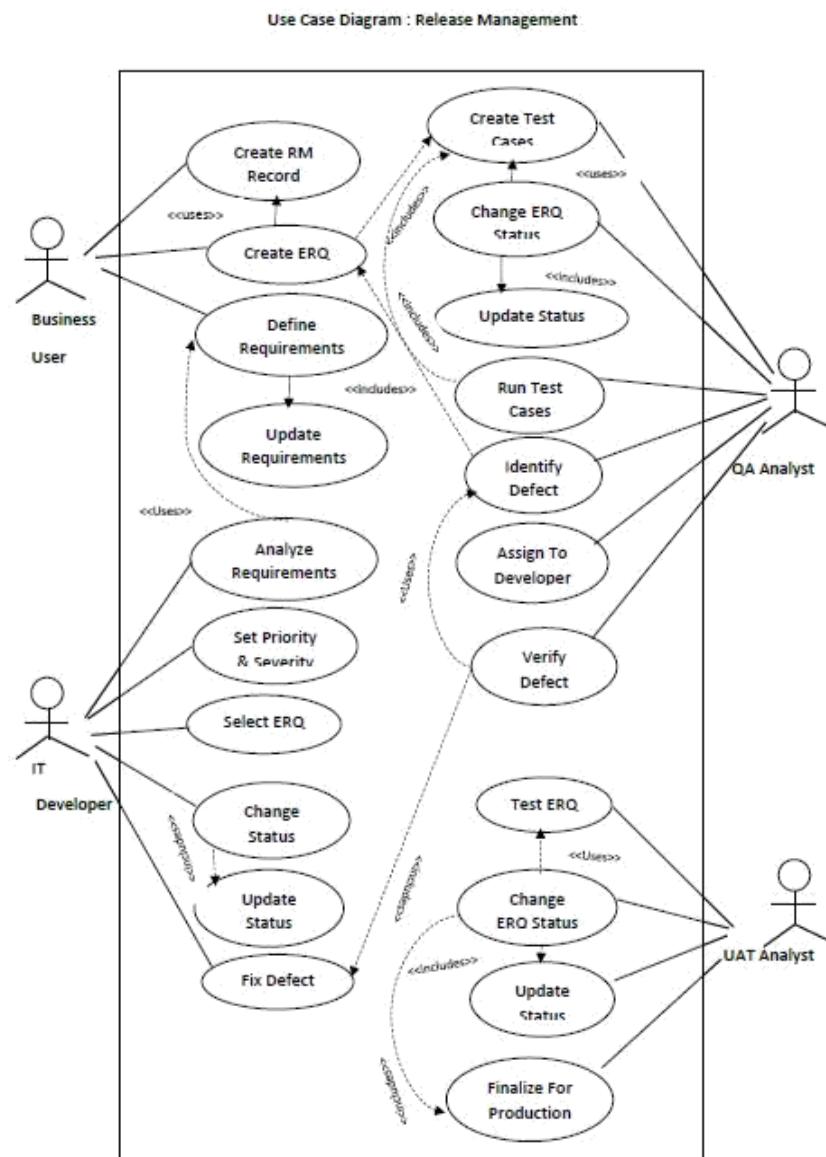


3.2 Class Diagram

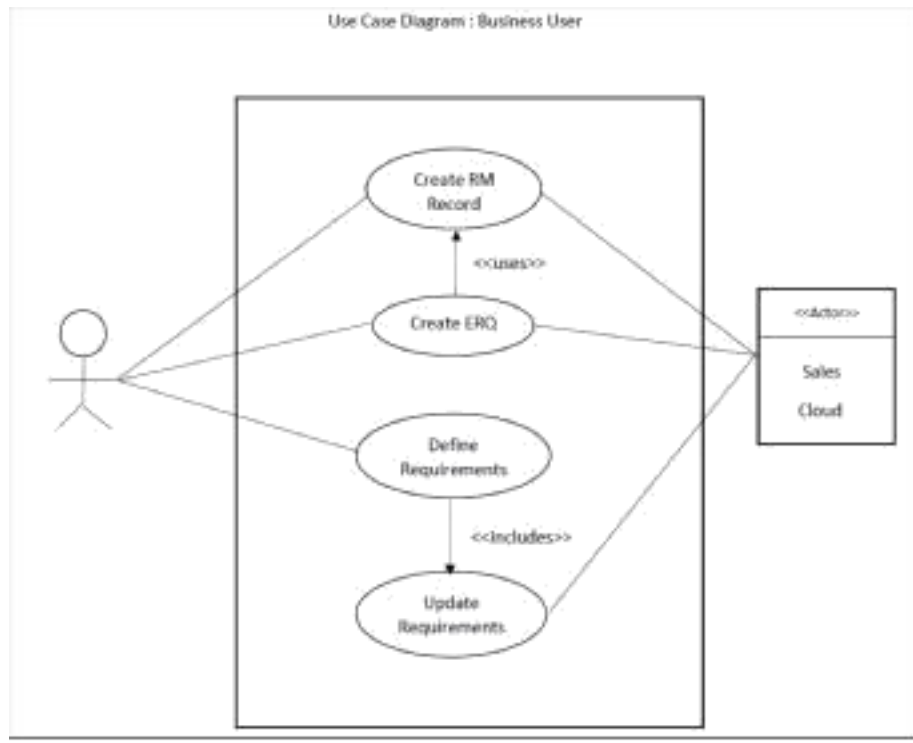


3.3 Use Case Diagram

1. Use Case Diagram

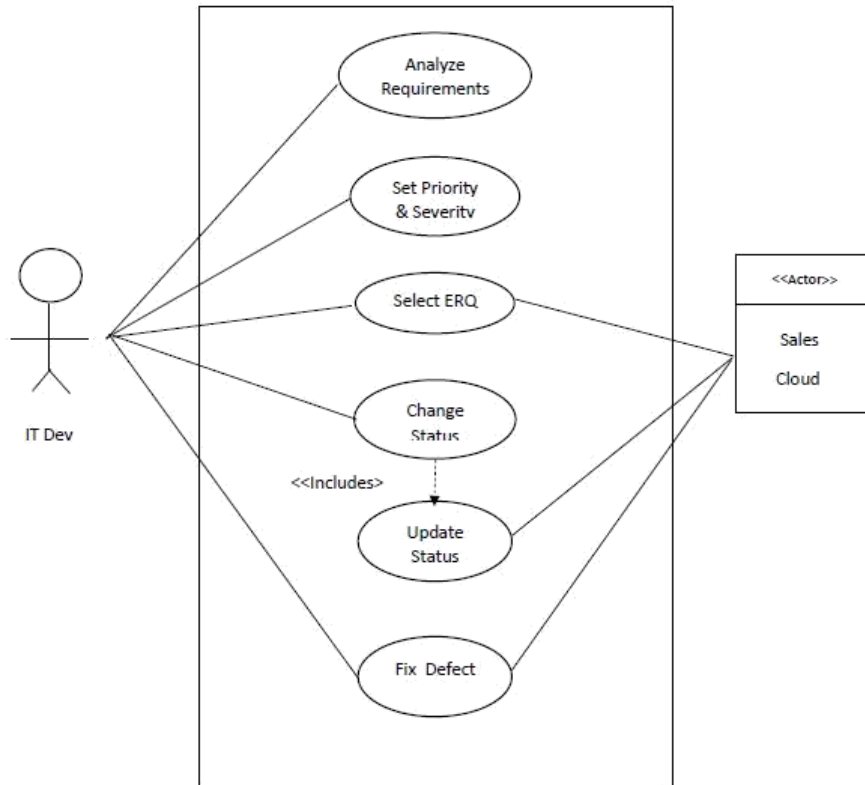


2. Use Case Diagram : Business user

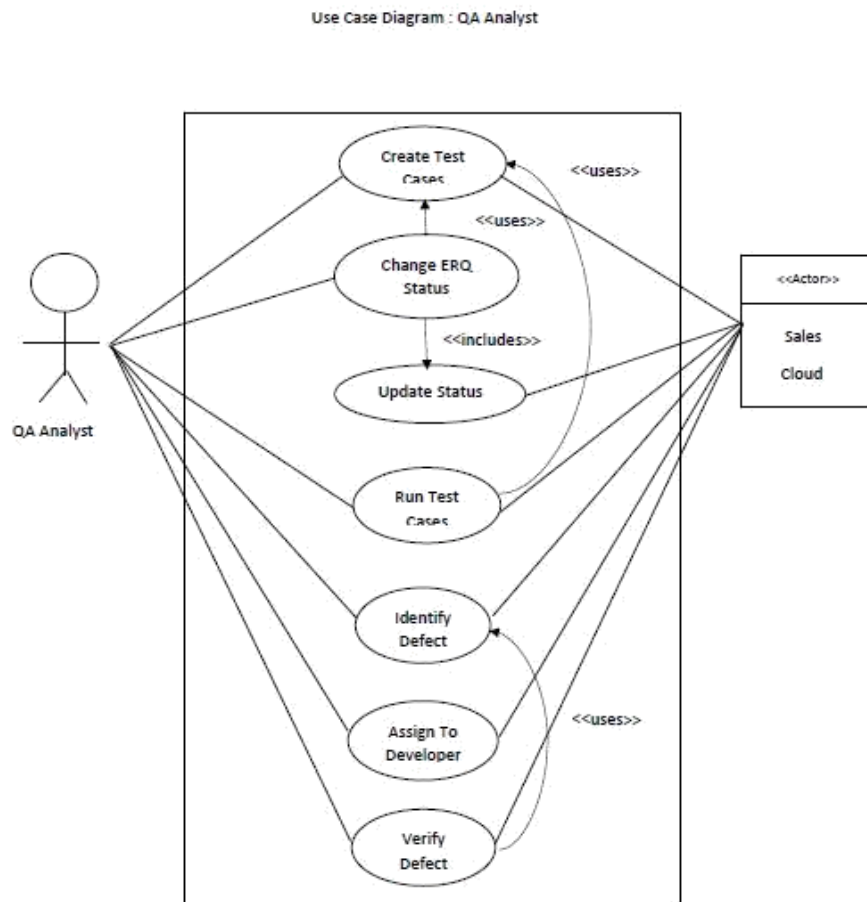


3. Use Case Diagram : IT Developer

Use Case Diagram : IT Developer

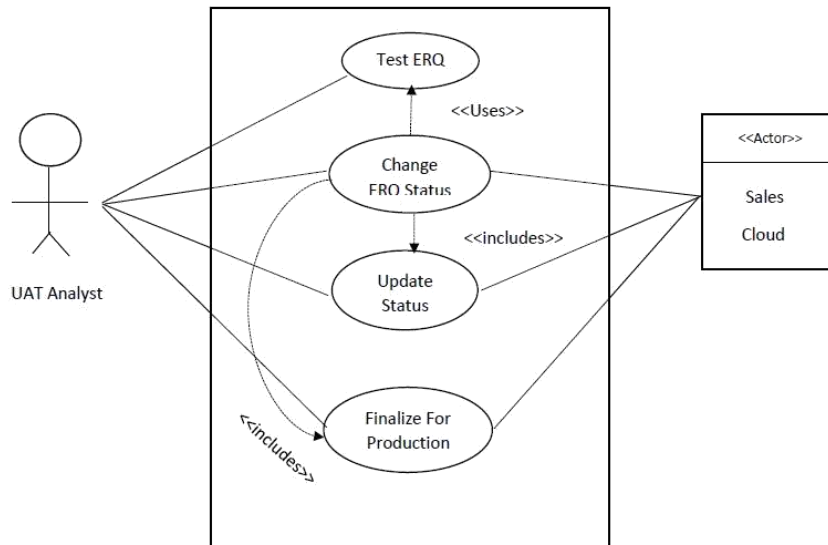


4. Use Case diagram : QA Analyst



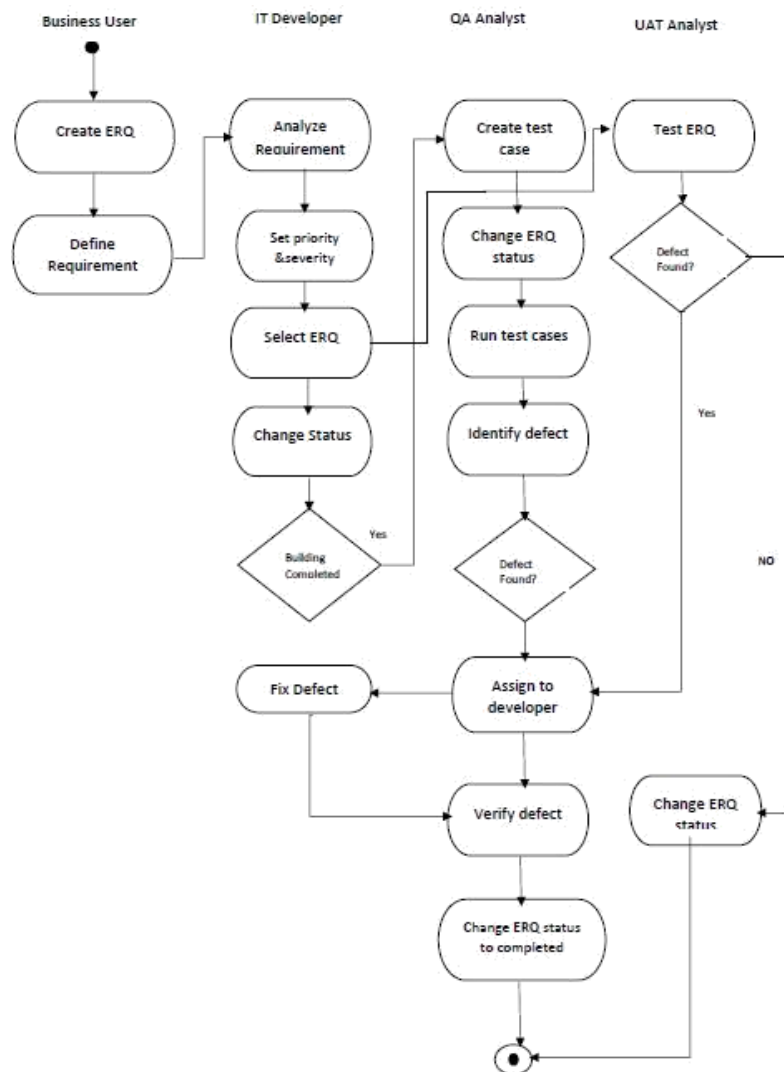
5. Use Case Diagram : UAT Analyst

Use Case Diagram : UAT Analyst



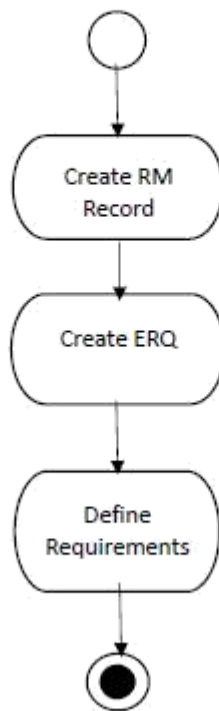
3.4 Activity Diagram

1. Activity Diagram



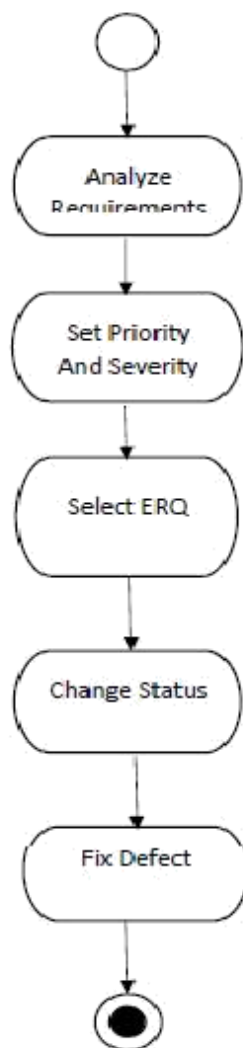
2. Activity Diagram : Business User

Activity Diagram : Business User



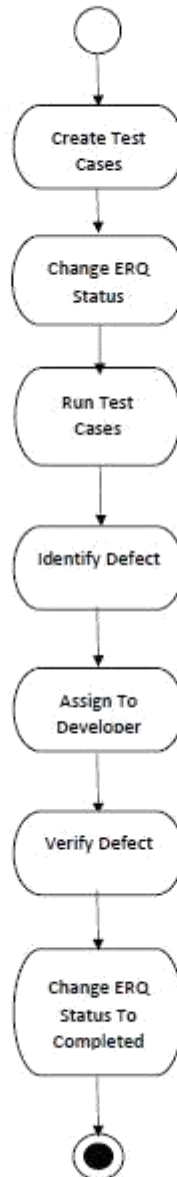
3. Activity Diagram : IT Developer

Activity Diagram : IT Developer



4. Activity Diagram : QA Analyst

Activity Diagram : QA Analyst



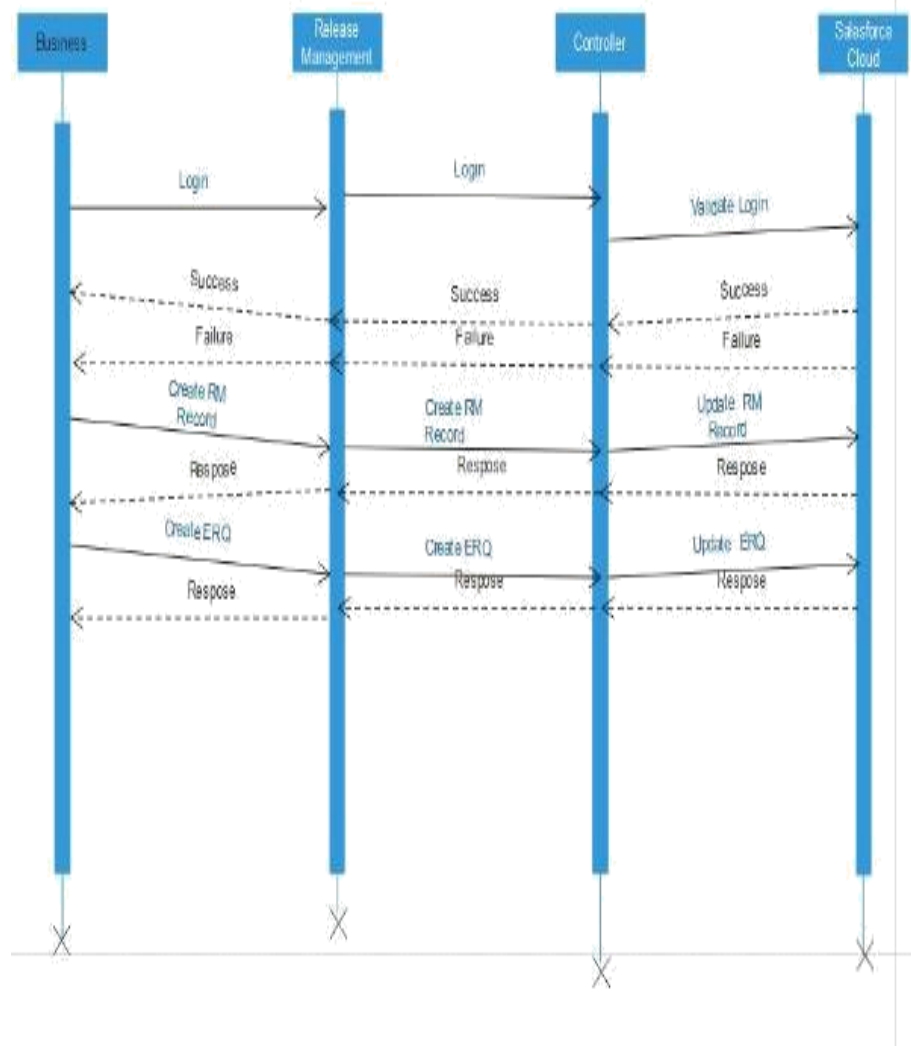
5. Activity Diagram : UAT Analyst

Activity Diagram : UAT Analyst

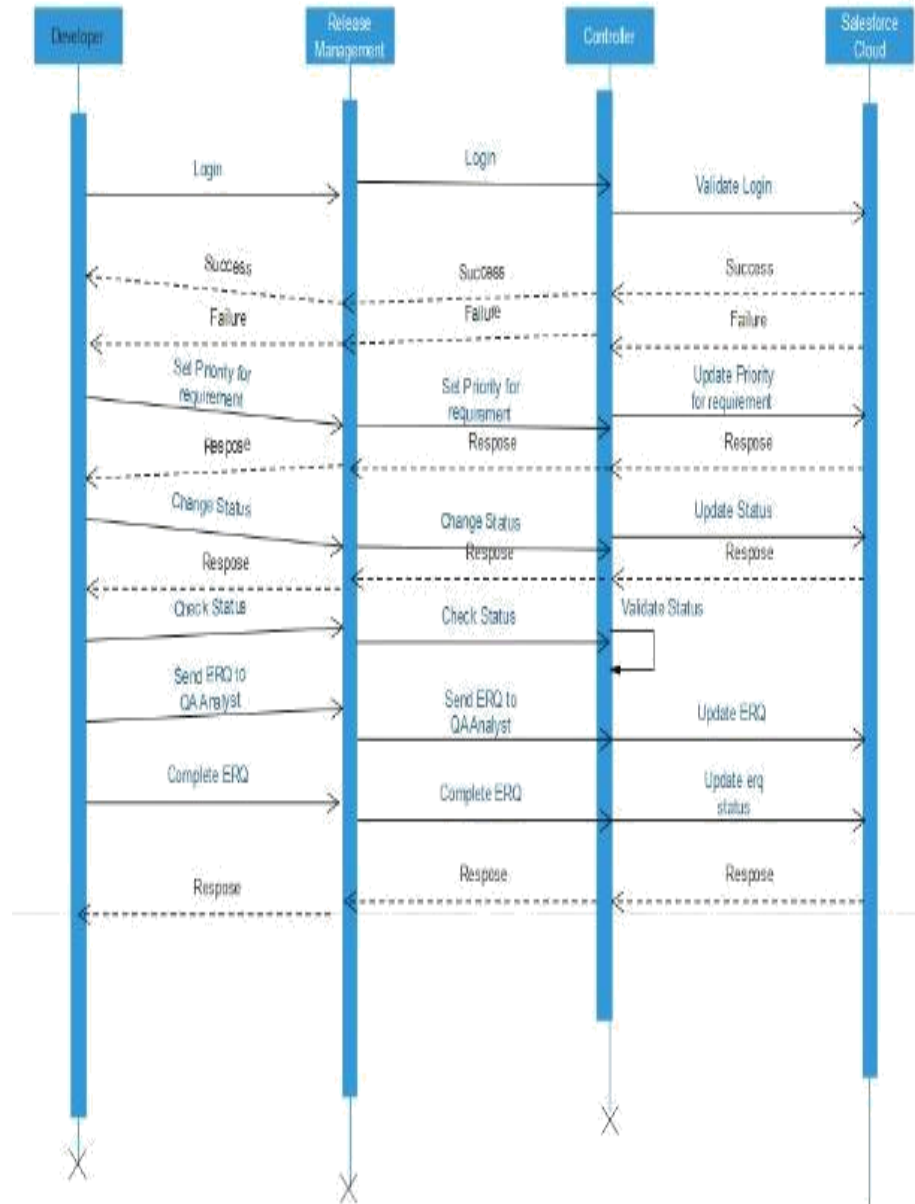


3.5 Sequence Diagram

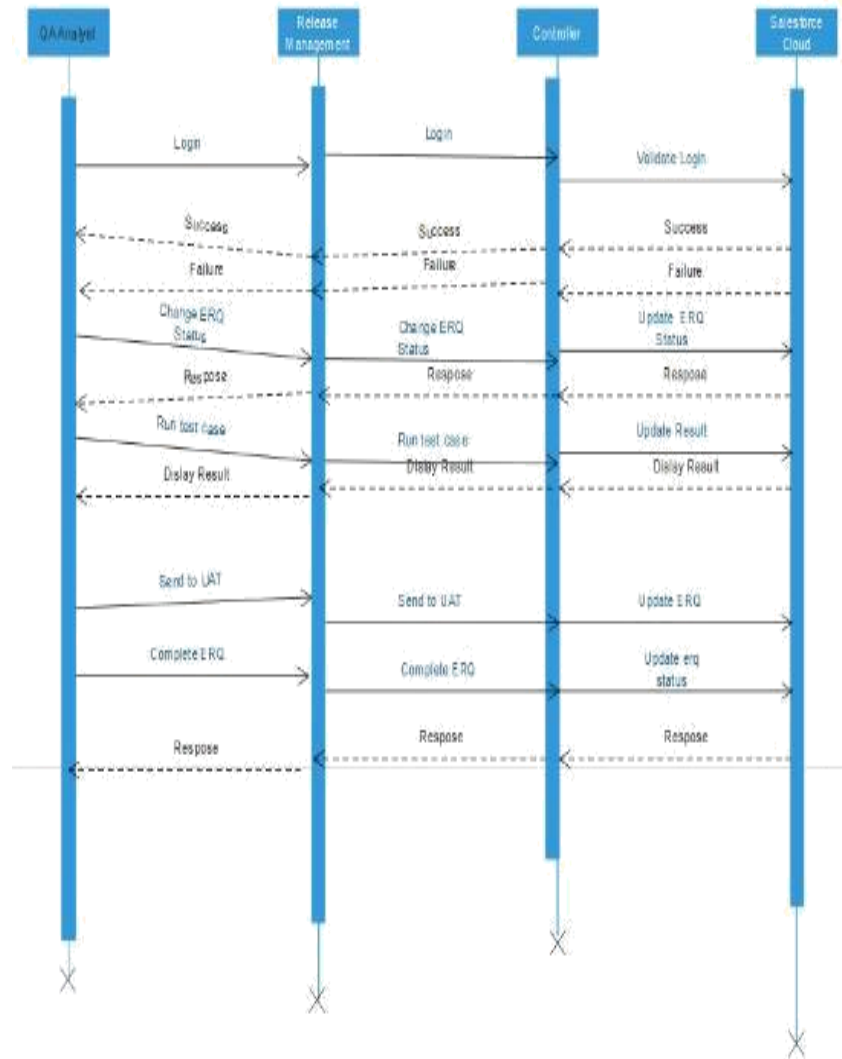
1. Sequence Diagram : Business User



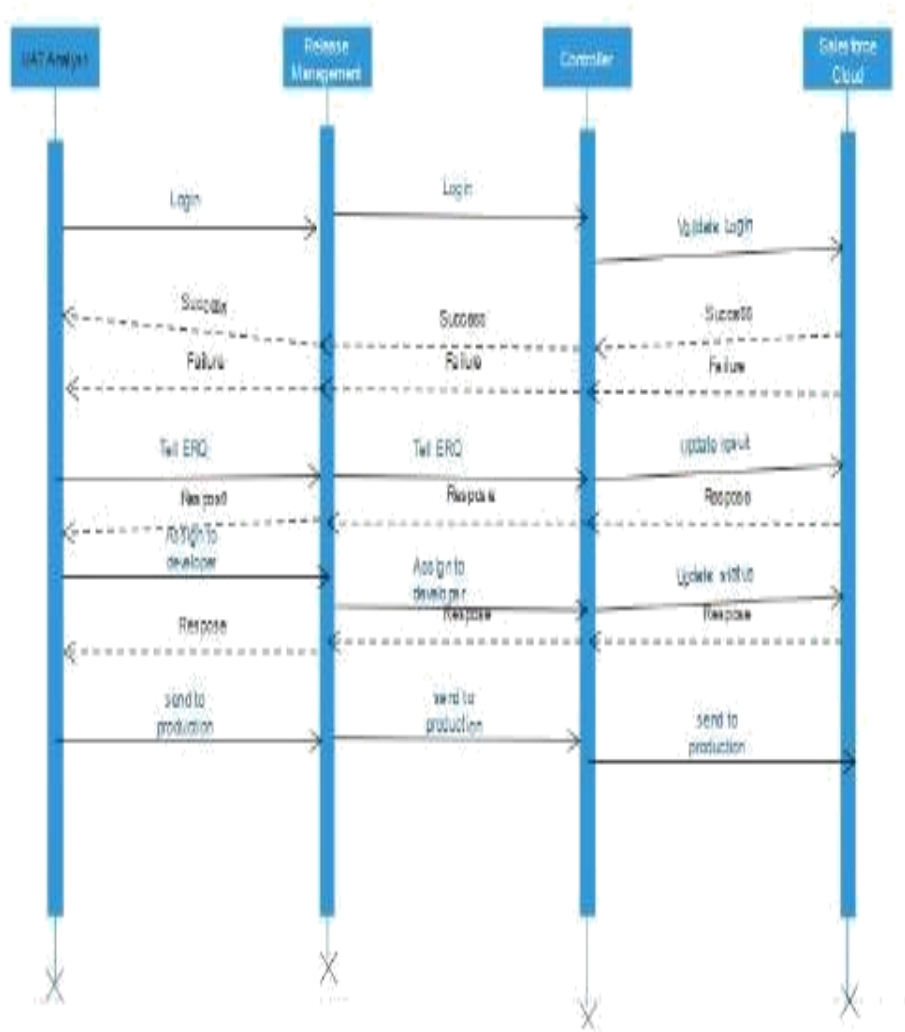
2. Sequence Diagram : IT Developer



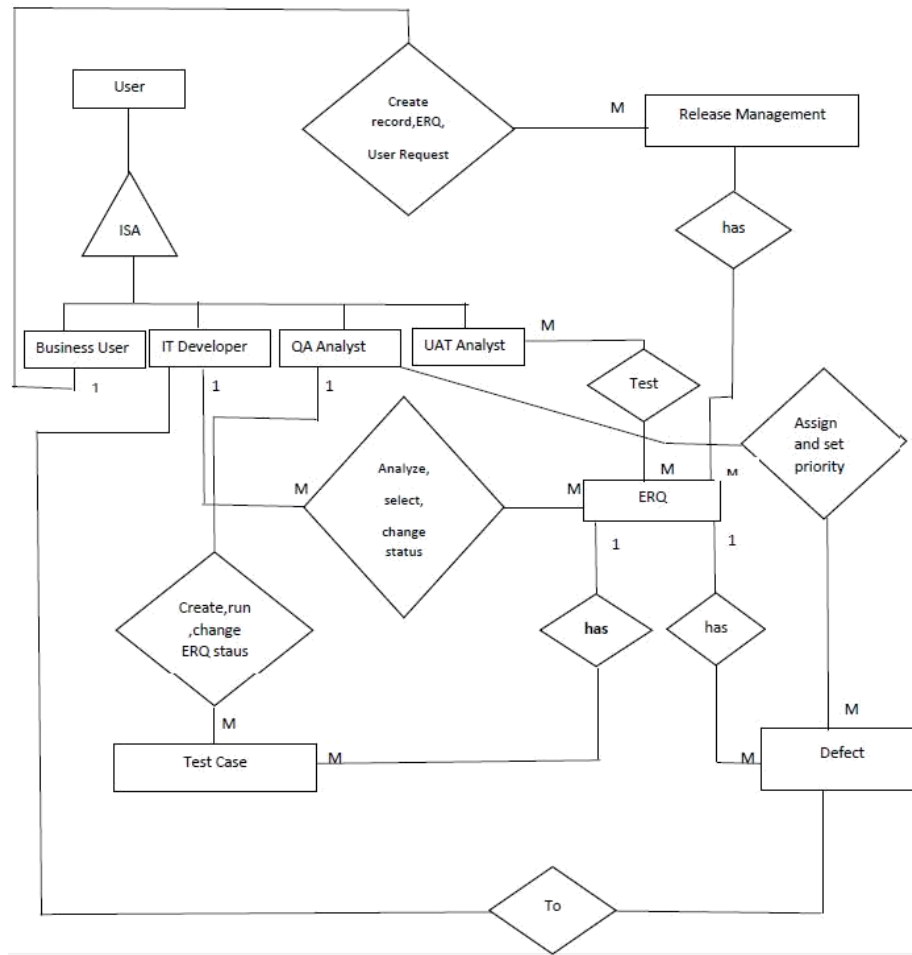
3. Sequence diagram : QA analyst



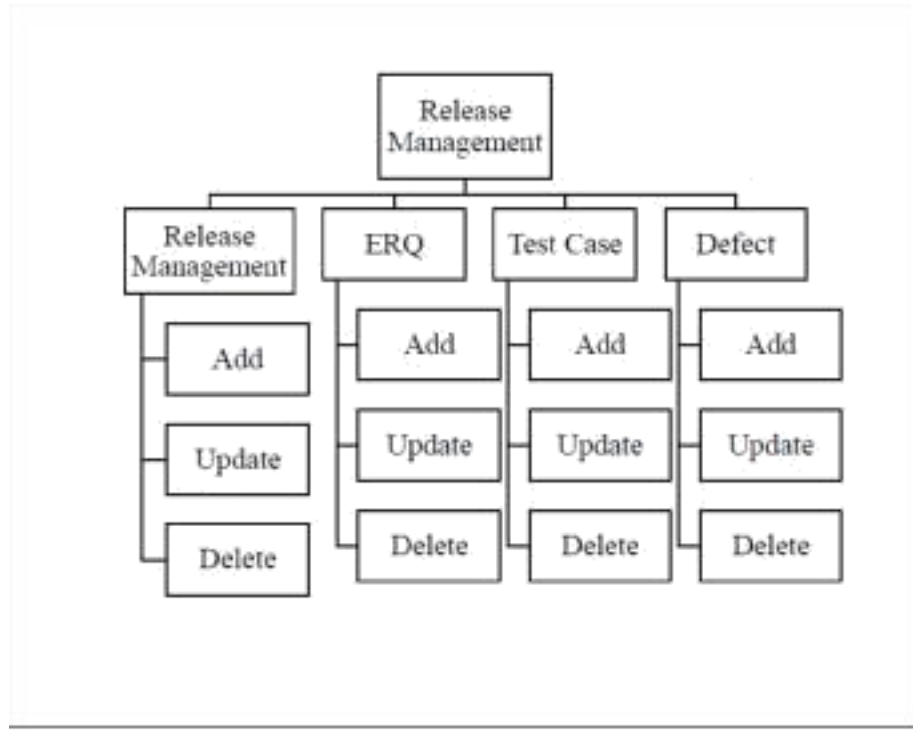
4.Sequence diagram : UAT analyst



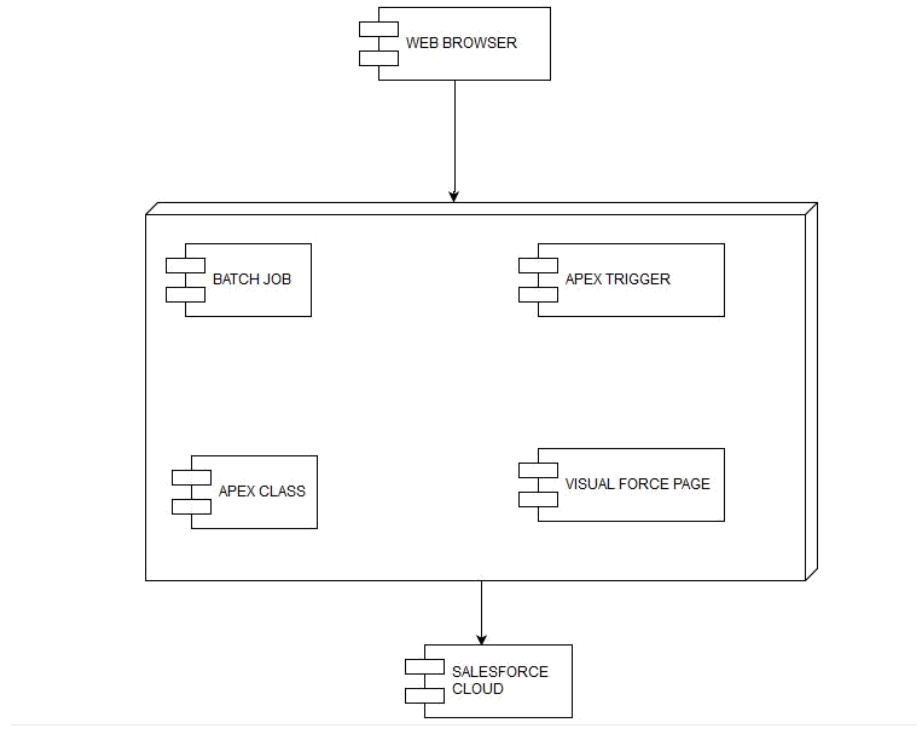
3.6 Entity Relationship Diagram



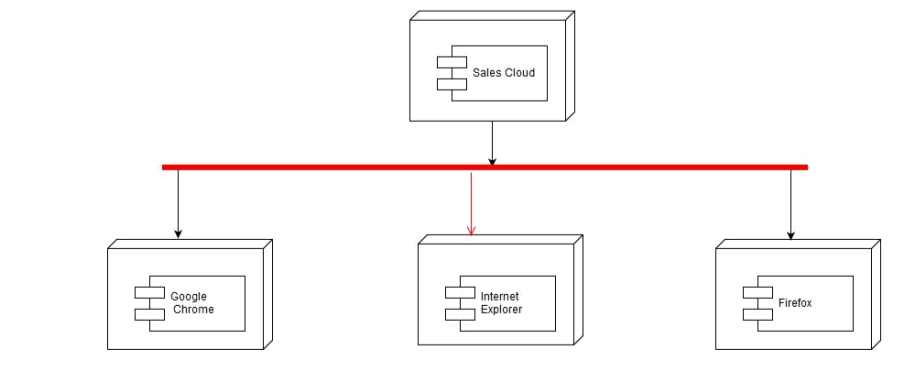
3.7 Module Hierarchy Diagram



3.8 Component Diagram



3.9 Deployment Diagram



3.10 Module Specifications

Release Management :

In this module, business users create enhancement request based on their requirements.

ERQs:

We have monthly releases that consist of ERQs. Those ERQs are the requirements given by the business. If there is any requirement which is impacting multiple areas then that ERQ needs to be approved by ops Council. Ops council is group where all business users and managers decide whether the ERQ should be worked upon or not.

IT developers analyze those requirements and set priority and severity. After selection of ERQ, development starts.

When development is completed, ERQs are ready for the testing.

Test Cases:

QA analyst create test cases. Run the test cases on ERQs.

If any defect is found, it assigned to the user by sending email notification.

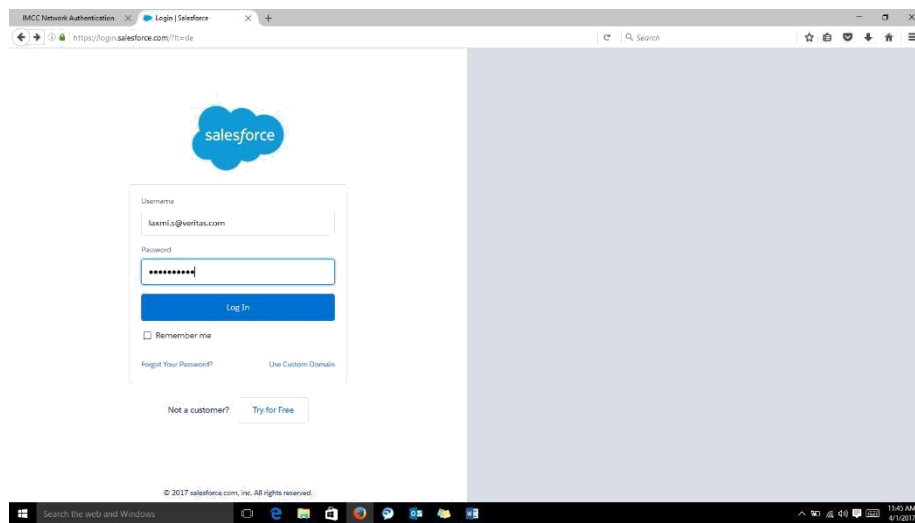
IT developer fix the defect. QA analyst verify that defect is fixed or not. After verification, ERQs are sent to UAT analyst. If any defect is identified during UAT, it is assigned to the user by email notification.

Defects:

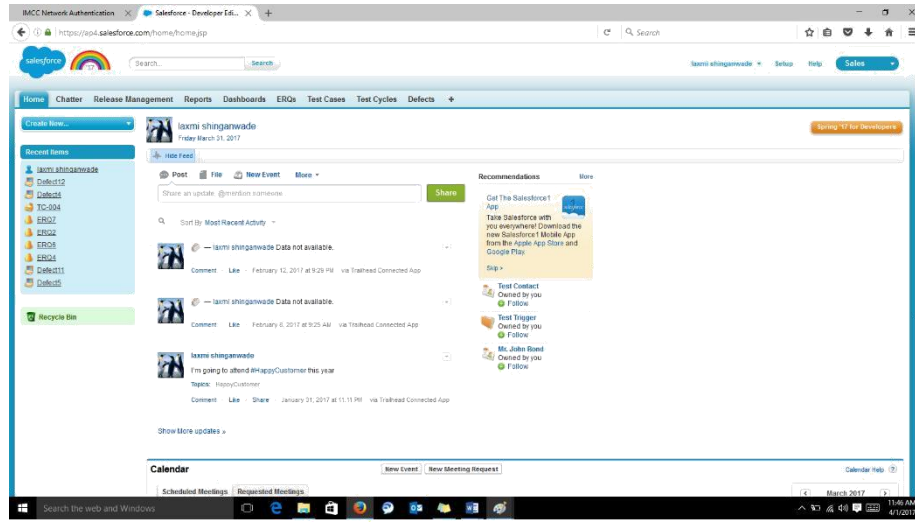
When new defect is created or status is changed, email notification is sent to the assigned user.

3.11 User interface Design

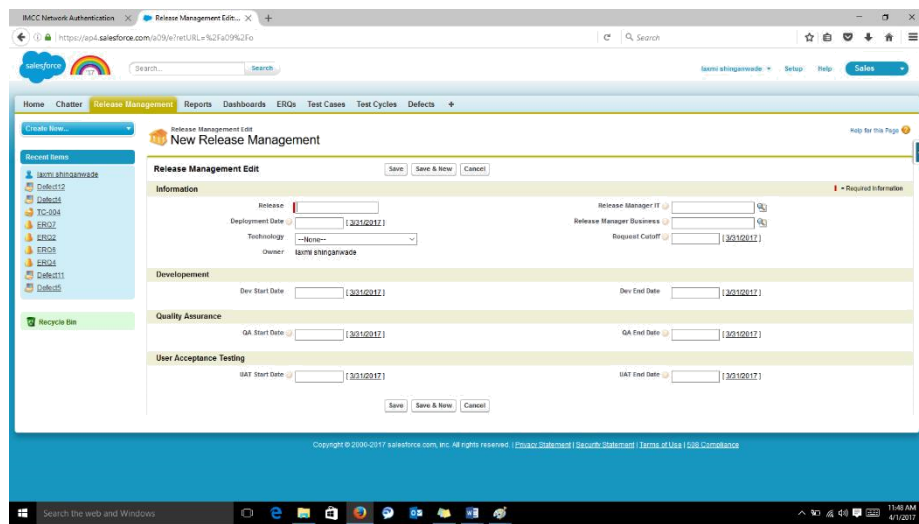
Log In:



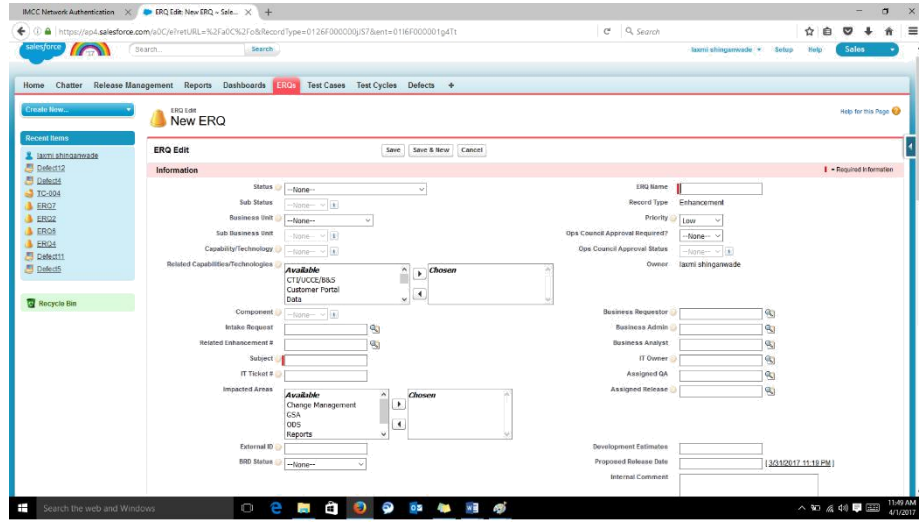
Home Page



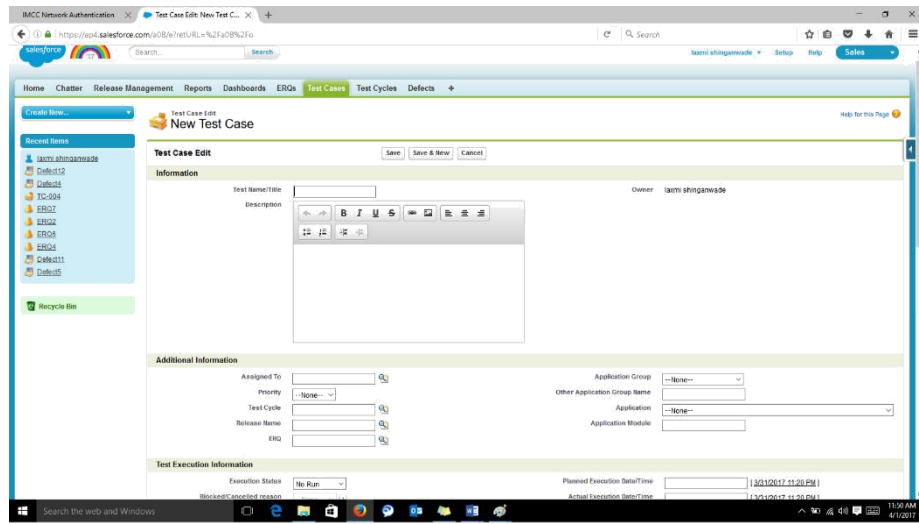
New Release Management



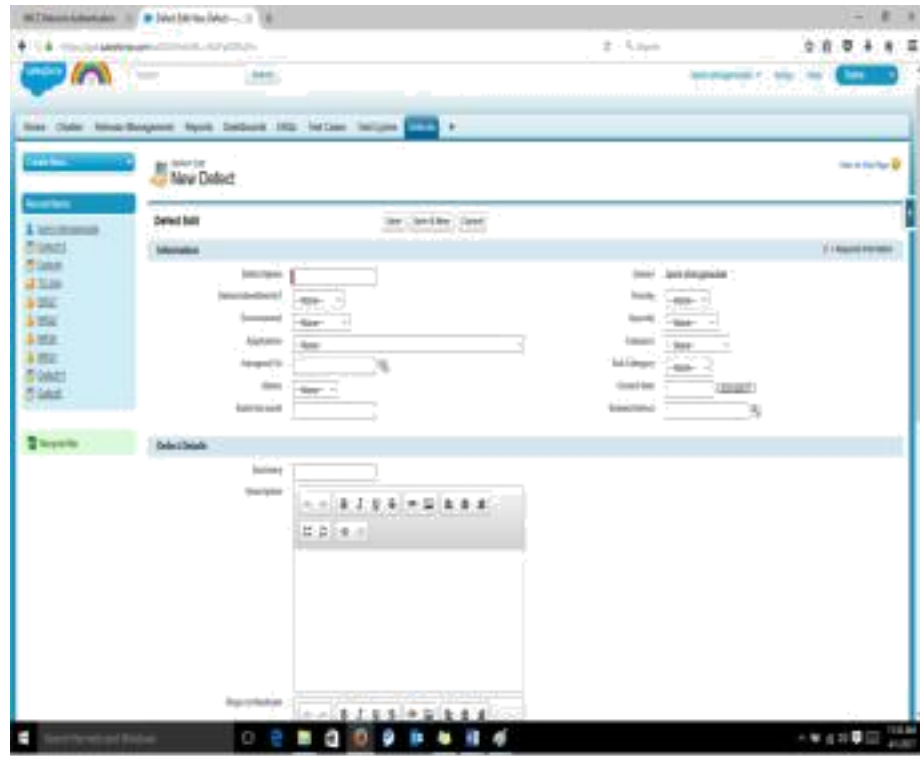
New ERQ



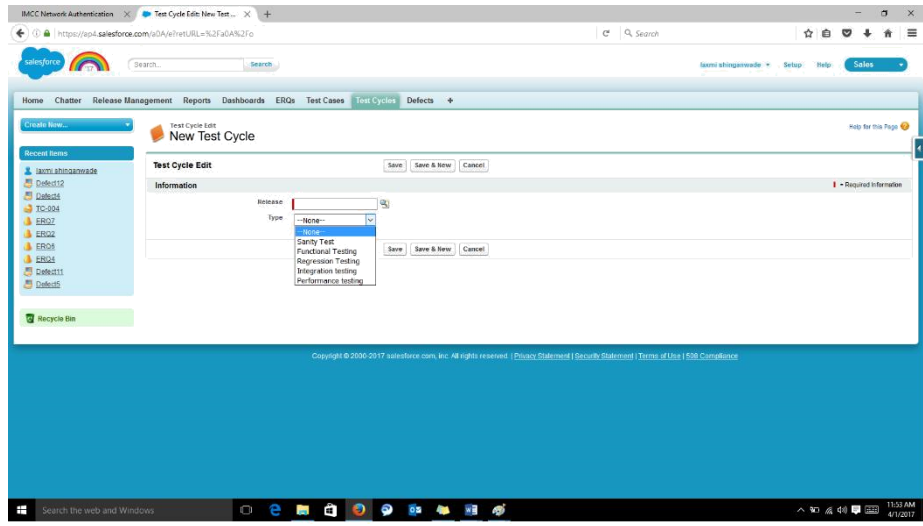
New Test Case



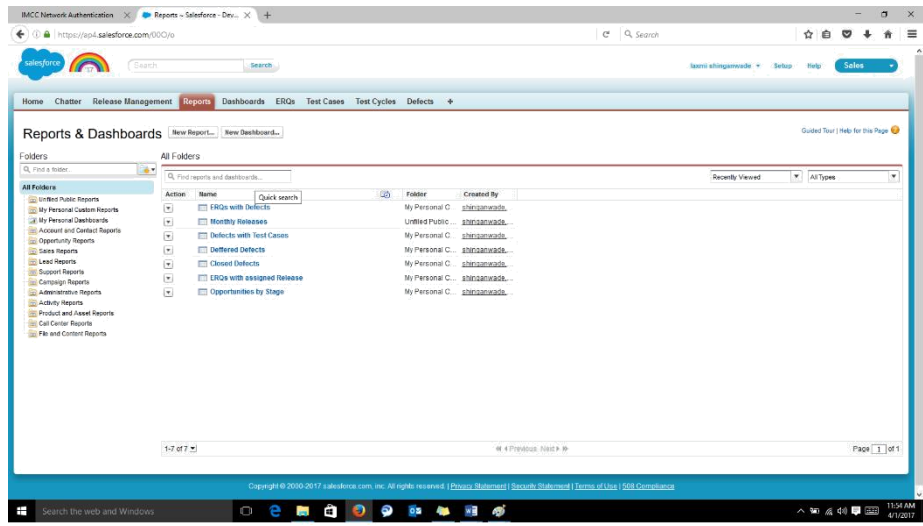
New Defect



New Test Cycle



Reports



3.12 Data Dictionary

COLUMN NAME	DATA TYPE	CONSTRAINTS
Acceptance Criteria	Long Text Area(15000)	
Actual Execution Date/Time	Date/Time	
Actual Result	Rich Text Area(32768)	
Application	Picklist	
Application	Picklist	
Application Group	Picklist	
Application Module		
Assigned QA	Lookup (User)	
Assigned Release	Lookup (Release Management)	
Assigned To	Lookup (User)	
Assigned To	Lookup (User)	
Blocked or Cancel Reason	Picklist	
Build Version	Text (10)	
Business Requestor	Lookup (User)	
Business Admin	Lookup (User)	
Business Analyst	Lookup (User)	
Business Reason	Long Text Area (15000)	
Business Unit	Picklist	
Capability/ Technology	Picklist	
Category	Picklist	
Change Owner	Picklist	
Change Ticket	Text (50)	
Closed Date	Date/Time	
Closed Date	Date	
Defect Identified In?	Picklist	
Defect Id	Autonumber	Primary Key
Deployment Date	Date	
Description	Rich Text Area (32768)	

Description	Rich Text Area (32768)	
Dev End date	Date	
Dev Start Date	Date	
Development Estimates	Number(16,0)	
Environment	Picklist	
ERQ	Lookup (ERQ)	
ERQ	Lookup (ERQ)	
ERQ Name	Text (80)	
ERQId	Autonumber	Primary Key
Execution Status	Picklist	
Expected Result	Rich Text Area (32768)	
External Id	Text(255)	
Intake Request	Lookup (Intake Request)	
IT Owner	Lookup (User)	
IT Ticket	Text(30) (External Id)	
OPS Council Approval Required?	Picklist	
OPS Council Approval Status	Picklist	
Other Application Group Name	Text (255)	
Owner	Lookup (User,Queue)	
Owner	Lookup (User,Queue)	
Planned Execution Date/Time	Date/Time	
Priority	Picklist	
Priority	Picklist	
Priority	Picklist	
Proposed Release date	Date/Time	
QA End date	Date	
QA Start Date	Date	
Record type	Record type	
Related Defect	Lookup (Defect)	
Related enhancement	Lookup (ERQ)	
Release	Lookup (Release management)	

Release Manager Business	Lookup (User)	
Release IT Manager	Lookup (User)	
Release name	Lookup (Release Management)	
Resolution Comments	Long Text Area (131072)	
RMID	Autonumber	Primary Key
Root Cause	Long Text Area (131072)	
Severity	Picklist	
Status	Picklist	
Status	Picklist	
Steps To Replicate	Rich Text Area (32768)	
Sub Business Unit	Picklist	
Sub Category	Picklist	
Sub Status	Picklist	
Summary	Text (255)	
Technology	Picklist	
Test Case	Lookup(Test cases)	
Test Cycle	Lookup (Test cycle)	
Test Name / Cycle	Text (255)	
Test Steps	Text(255)	
TestCaseId	Autonumber	Primary Key
UAT End Date	Date	
UAT Start Date	Date	

3.13 Table Specifications

1. Release Management

COLUMN NAME	DATA TYPE	CONSTRAINTS
ID	Autonumber	Primary Key
Change Owner	Picklist	
Deployment Date	Date	
Dev End Date	Date	
Dev Start Date	Date	
QA End Date	Date	
QA Start date	Date	
Release Manager business	Lookup (USER)	
Release Manager It	Lookup (USER)	
Technology	Picklist	
UAT End Date	Date	
UAT Start date	Date	

2. ERQs

COLUMN NAME	DATA TYPE	CONSTRAINTS
ID	Autonumber	Primary Key
Status	Picklist	
Sub Status	Picklist	
Business Unit	Picklist	
Sub Business unit	Picklist	
Capability/ Technology	Picklist	
Intake Request	Lookup (INTAKE REQUEST)	
Related Enhancement	Lookup (ERQ)	
IT Ticket	Text(30) (EXTERNAL ID)	
External Id	Text (255)	
ERQ name	Text (80)	

Record Type	Record Type	
Priority	Picklist	
OPS Council Approval Required?	Picklist	
OPS Council Approval Status	Picklist	
Business Requestor	Lookup (USER)	
Business Admin	Lookup (USER)	
Business Analyst	Lookup (USER)	
IT Owner	Lookup (USER)	
Assigned QA	Lookup (USER)	
Assigned Release	Lookup (RELEASE MANAGEMENT)	
Development Estimates	Number(16,0)	
Proposed Release Date	Date/Time	
Business Reason	Long Text Area (15000)	
Acceptance Criteria	Long Text Area (15000)	
Closed Date	Date/Time	

3. Test Cases

COLUMN NAME	DATA TYPE	CONSTRAINTS
ID	Autonumber	Primary Key
Test Name / Cycle	Text (255)	
Description	Rich Text Area (32768)	
Owner	Lookup (USER,QUEUE)	
Assigned To	Lookup (USER)	
Priority	Picklist	
Test Cycle	Lookup (TEST CYCLE)	
Release Name	Lookup (RELEASE MANAGEMENT)	
ERQ	Lookup (ERQ)	
Application Group	Picklist	
Other Application Group Name	Text (255)	
Application	Picklist	
Application Module		

Execution Status	Picklist	
Blocked or Cancel Reason	Picklist	
Planned Execution Date /Time	Date/Time	
Actual Execution Date /Time	Date/Time	
Test Steps	Text (255)	
Expected Result	Rich Text Area (32768)	
Actual Result	Rich Text Area (32768)	

4. Defect

COLUMN NAME	DATA TYPE	CONSTRAINTS
ID	Autonumber	Primary Key
Defect Identified in?	Picklist	
Environment	Picklist	
Application	Picklist	
Assigned To	Lookup (USER)	
Status	Picklist	
Build version	Text (10)	
Owner	Lookup (USER,QUEUE)	
Priority	Picklist	
Severity	Picklist	
Category	Picklist	
Sub Category	Picklist	
Closed date	Date	
Related Defect	Lookup (DEFECT)	
Summary	Text (255)	
Description	Rich Text Area (32768)	
Steps to Replicate	Rich Text Area(32768)	
Root Cause	Long Text Area (131072)	
Resolution Comment	Long Text Area (131072)	
ERQ	Lookup (ERQ)	
Release	Lookup (RELEASE MANAGEMENT)	
Change Ticket	Text (50)	
Test Case	Lookup (TEST CASES)	

3.14 Test Procedures and Implementation

TESTING

Testing plays a vital role in the success of the system. System testing makes a logical assumption that if all parts of the system are correct, the goal will be successfully achieved. Once program code has been developed, testing begins. The testing process focuses on the logical internals of the software, ensuring that all statements have been tested, and on the functional externals, that is conducted tests to uncover errors and ensure that defined input will produce actual results that agree with required results.

TYPES OF TESTING:

- Unit testing
- Integration testing
- Validation testing
- System testing
- Negative testing

1) **Unit testing:**

Unit testing focuses verification effort on the smallest unit of software design the module. Unit testing exercise specific paths in the module's control structure to ensure complete coverage and maximum error detection.

i) Black Box Testing: In black box testing, test cases are designed from an examination of the input/output values and no knowledge of design or code is requires.

ii) White Box Testing: There are several white box testing strategies. Each testing is based on some heuristic. White box testing strategy is based on heuristic. White box testing is stronger than other strategy, if all types of errors detected by the first strategy are also detected by the second testing strategy & the second strategy additionally detects some more types of errors.

2) **Integration Testing:**

Integration testing addresses the issues associated with the dual problems of verification & program construction. After the

software has been integrated a set of high-order tests are conducted/

The following are the types of Integration Testing:

Top-Down Integration:

This method is an incremental approach to the construction of program structure. Modules are integrated by moving downward through the control hierarchy, beginning with the main program module.

Bottom-Up Integration:

This method begins the construction & testing with the modules at the lowest level in the program structure. Since the modules are integrated from the bottom up, processing required for modules subordinate to a given level is always available & the need from stubs is eliminated.

3) Validation Testing:

At the end of Integration Testing, software is completely assembled as a package, interfacing errors have been uncovered and correction testing begins.

Software Testing & Validation is achieved through serried of black box tests that demonstrate conformity with the requirements. A test plan outlines the classes of tests to be conducted and a test procedure defines specific test cases that will be used to demonstrate conformity with requirements.

Both, the plan and the procedure are designed to ensure that all functional requirements are achieved, documentation is correct and other requirements are met.

4) System testing:

System testing is series of different tests whose primary purpose is to fully exercise the computer based system. Although each test has a different purpose, all the work should verify that all system elements have been properly integrated and perform allocated functions.

There are essentially three kinds of system testing:

- i) Alpha Testing: Alpha testing refers to the system testing carried out by the test team within the development organization.
- ii) Beta Testing: Beta testing is the system testing performed by a select group of friendly Users.
- iii) Acceptance Testing: Acceptance testing is the system testing performed by the Users to determine whether to accept or reject the delivery of system.

5) Negative Testing:

Negative testing ensures that your application can gracefully handle invalid input or unexpected user behaviour. For example, if a user tries to type a letter in a numeric field, the correct behaviour in this case would be to display the “Incorrect data type, please enter a number” message.

Test item-Login

Cardinality: negative

Test case name	Test case description	Input	Expected output	Actual output	Status	Priority
Validate login	User name or password Fields are blank	Username = Password =	Focus on Username and password field and Give message Field should not be empty	Please enter user name	pass	High
Validate Login	Password Entered is wrong	Username = Sharvari Password = Password	Display message Incorrect password	Incorrect User-name or password	pass	High

Cardinality: positive

Test case name	Test case description	Input	Expected output	Actual output	Status	Priority
Validate login	Enter correct user name And Password	Username=sharvari.bhavsar@centelon.com Password=Sharvari123	Successfully login and redirect To home page	Redirect to Home page Of Appropriate user	pass	High

Test item-User Signup

Cardinality: negative

Test case name	Test case description	Input	Expected output	Actual output	Status	Priority
Validate Email-id filed	Enter wrong Email id	Emailed = a@bcom	Message will be display Invalid email id	Invalid email id	pass	High
Validate Emailed field	Filed kept Blank	Emailed =	Message will be Display Emailed	Message will be	pass	High

			Required	Displayed Required		
--	--	--	----------	-----------------------	--	--

Cardinality : positive

Validate login	Enter all Field correctly	Username=a Emailed= a@gmail.com	Successfully registered And redirect To login page	Successfully registered And redirect To login page	pass	High
----------------	---------------------------	--	--	--	------	------

Test item: Change password

Test case name	Test case description	Input	Expected output	Actual output	Status	Priority
Validate Change password field	Enter password	password = Sharvaril 23	Message will be display Enter confirm password	Message will be display Enter confirm password	pass	High

Validate Confirm password field	Field kept Blank	Confirm password = 	Message will be Display Confirm password Required	Message will be Display Confirm Password Required	pass	High
Validate Confirm password field	Confirm Password entered wrong	Confirm password = Sharv	Passwords must be same Alert	Passwords must be Same alert	Pass	High

Test item: Edit profile

Cardinality: positive

Test case name	Test case descriptio n	Input	Expecte d Output	Actu- al output	Statu s	Priorit y

Validate Userna me	Enter Usernam e	Click on usernam e	Messag e Will be Display that this field cannot change	Messag e Will be Display that this field cannot change	pass	High
--------------------------	-----------------------	-----------------------------	---	--	------	------

CHAPTER 4

USER MANUAL

4.1 User Manual

The user manual is meant to be used by all users using the system, with prior training session from the Development unit, plus he/she has to be skilful enough in operating the intended system. This manual is used for benefit of intended user to make it more clearly along with functioning of system, the processes and precautions that must be followed during working with the system. It also tells the user how to use various functionality of the site being provided according to the type of user, whenever user logs in to the system.

There are many users of Release management such as business user, IT developer, QA Analyst, UAT analyst.

The users can go to www.salesforce.com, and login using the own credential.

The IT developer can add release management record for the requirements specified by the business users.

The users can view releases, ERQs, test cases, defect information by clicking on the respective tab visible on the

menu bar and can update or delete record by clicking on edit or delete button and can even add new record by clicking on new button.

The user can login to the system by login page, if user doesn't have login id he can create one via sign up option. The users can click on the tab chatter and write message in it and click enter which will post the message on the chatter wall.

4.2 Operations Manual/Menu Explanation

Home Page Menu:

The home page of the website contains menu

Home

Release Management

ERQs

Test Cases

Defect

Reports

Dashboard

ERQs Consists of four Record types:

1. Enhancement: By selecting enhancement type you can view the list of enhancement records and by clicking on it you can view the ERQ details.

2. Project Requirement:

By selecting project requirement type you can view the list of project record and by clicking on it you can view the project details.

By clicking on new button you can create a new record for project.

3. Sales and marketing: By selecting sales and marketing type you can view the list of records and by clicking on it you can view the more details.

4. VLink Case: By selecting VLink case type you can view the list of records and by clicking on it you can view the more details.

4.3 Program Specification/ Flow Chart

Following are the Program specifications used in the development process of the system.

Release Management:

Module Name	Release Management	
Program Name	Manage Release	
Purpose	To Create RM record	
Event	Select Deployment date	
Input	Constraints	Description
Select “Deployment Date”	The required field should not be null	Deployment date must not be the past date
Output	We can not select past date for deployment date	

ERQs:

Module Name	ERQs
Program Name	Requirements
Purpose	Create records for requirements based on priority and severity.
Event	After Creation of RM record

Input	ERQ should be approved by Ops Council. Ops Council Status should be “Approved”
Output	ERQ should be assigned to developer.

Test cases:

Module Name	Test cases
Program Name	TestCase
Purpose	Testing Requirements
Event	After completion of Development process of ERQs.
Input	Select Release and ERQ. Enter planned Execution date.
Output	Identify defect and assign it to developer

Defects:

Module Name	Defect
Program Name	Defect
Purpose	To identify and fix defect
Event	After QA or UAT
Input	Assign it to developer
Output	ERQ will be finalize for production

Drawbacks of the system

This system requires 24*7 internet connection without which system will not work.

Even if the internet speed is low this system will not work efficiently.

It needs good internet speed.

Without internet user cannot access the data.

Even if user exports the data from salesforce to refer it offline or save in the workstation, it will be only saved in the form of CSV file and no other format.

Proposed Enhancement

Sometimes, large companies have multiple different release management processes even for the same org. For one division, they may be using one set of processes and for another division, they could be having a totally different set of processes. It is necessary to streamline and standardize the processes across the various sets of users since they are using the same foundation, platform and resources to enable their business.

Release management has been implemented on salesforce platform..

We have monthly releases that consist of ERQs. Those ERQs are the requirements given by the business. If there is any requirement which is impacting multiple areas then that ERQ needs to be approved by ops Council. Ops council is group where all business users and managers decide whether the ERQ should be worked upon or not.

Release management is capable of managing the Release/Requirements/Test cases /Defects /Reports and Dashboards.

Conclusion

The project titled as “**Release Management**” is a white labeled application. This software is developed with scalability in mind. Additional modules can be easily added whenever required. The system is developed with modular approach. All modules of the system have been tested with valid and invalid data and everything works accurately. The project has been completed successfully with maximum satisfaction of organization. The system is designed as like it was decided in design phase. Project gives good idea on developing full-fledged application satisfying the user requirements. The system is very flexible .This system has user-friendly screens that enable users to use it without any inconvenience. The application has been tested with live data and has provided successful result. Hence the software has proved to work efficiently

BIBLIOGRAPHY

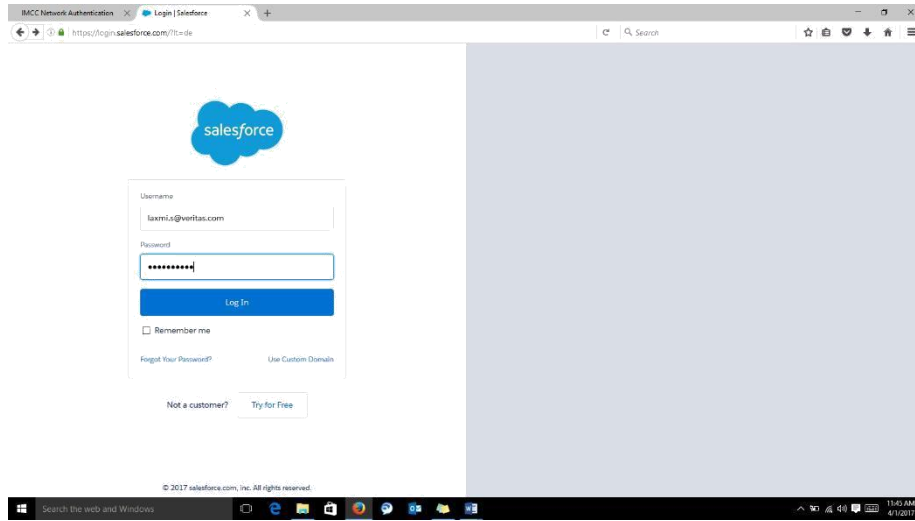
Bibliography

Websites:

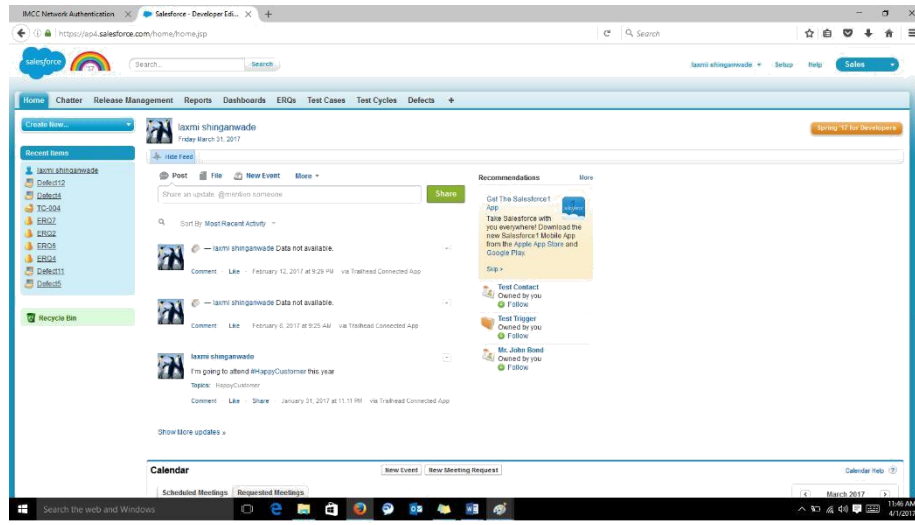
- www.google.com
- ¹⁰ <https://trailhead.salesforce.com/>
- www.salesforce.com
- www.wikipedia.com

ANNEXURE 1(User Interface Screens)

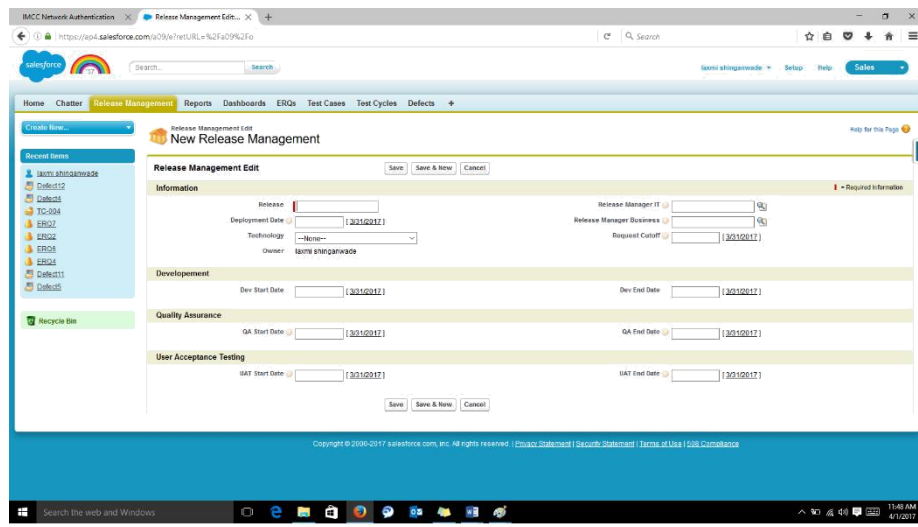
Log In:



♣ Home Page



New Release Management



♣ New ERQ

The screenshot shows the 'New ERQ' form in Salesforce. The form is titled 'ERQ Edit' and includes several sections:

- Information:** Fields for Status, Sub Status, Business Unit, Sub Business Unit, Capability/Technology, Business Responder, Business Analyst, and Assigned Release.
- ERQ Name:** A text field for the name of the ERQ.
- Priority:** A dropdown menu set to 'Low'.
- Record Type:** A dropdown menu set to 'Enhancement'.
- Opex Council Approval Required?:** A dropdown menu set to 'None'.
- Opex Council Approval Status:** A dropdown menu set to 'None'.
- Owner:** A text field with the value 'Iamri shingamade'.
- Business Responder:** A text field.
- Business Analyst:** A text field.
- IT Owner:** A text field.
- Assigned QA:** A text field.
- Assigned Release:** A text field.
- Development Estimates:** A text field.
- Proposed Release Date:** A date field with the value '1/24/2017 11:19 PM'.
- Internal Comment:** A text area.

⌘ New Test Case

The screenshot shows the 'New Test Case' form in Salesforce. The form is titled 'Test Case Edit' and includes several sections:

- Information:** Fields for Test Name/Title, Description (with a rich text editor), and Owner (Iamri shingamade).
- Additional Information:** Fields for Assigned To, Priority, Test Cycle, Release Name, ERQ, Application Group, Other Application Group Name, Application, and Application Module.
- Test Execution Information:** Fields for Execution Status (set to 'No Run'), Planned Execution DateTime (1/24/2017 11:20 PM), and Actual Execution DateTime (1/24/2017 11:20 PM).

New Defect

MCC Network Authentication X Defect Edit: New Defect - X +

https://ap4.salesforce.com/a0D/e/retURL=%2Ffa0%2Fo

salesforce Search... Search

laxmi shingamwade Setup Help Sales

Home Chatter Release Management Reports Dashboards ERGs Test Cases Test Cycles Defects +

Create New...

Defect Edit

New Defect Help for this Page

Defect Edit Save Save & New Cancel

Information - Required Information

Defect Name

Defect Identified In? --None--

Environment --None--

Application --None--

Assigned To

Status --None--

Build Version#

Owner laxmi shingamwade

Priority --None--

Severity --None--

Category --None--

Sub Category --None--

Closed Date

Related Defect

Recycle Bin

Defect Details

Summary

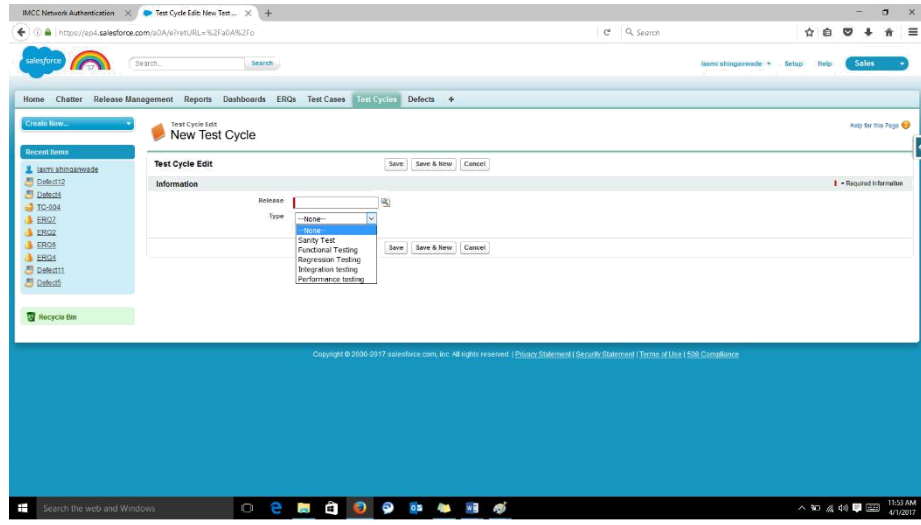
Description

Steps to Replicate

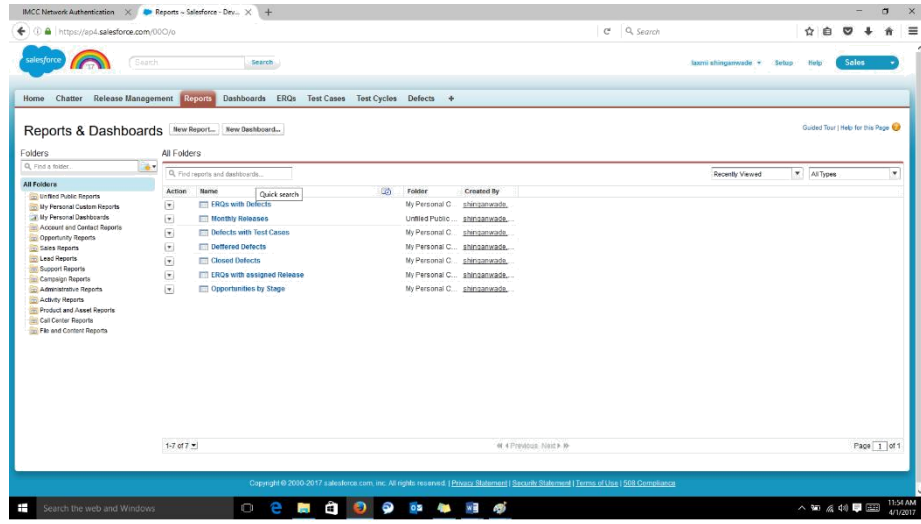
Search the web and Windows

11:52 AM 4/1/2017

New Test Cycle



Reports



ANNEXURE 2(Output Reports With Data)

ERQs with Assigned Release

Report Generation Status: Complete

Report Options:

Summarize Information by: Assigned Release: Release (My ERQs) | Date Field: Closed Date | Range: Custom

Time Frame: From: | To: |

Run Report | Hide Details | Customize | Save | Save As | Delete | Printable View | Export Details | Subscribe

Grouped By: Assigned Release: Release | Sorted By: Assigned Release: Release

ERQ	ERQ Name	Assigned Release: Release	Records
ERQ1	ERQ1	Release: Release	2 records
ERQ2	ERQ2	Release: Release	2 records
ERQ3	ERQ3	Release: Release	2 records
ERQ4	ERQ4	Release: Release	1 record
ERQ5	ERQ5	Release: Release	0 records
ERQ6	ERQ6		
ERQ7	ERQ7		
ERQ8	ERQ8		
ERQ9	ERQ9		
ERQ10	ERQ10		

Closed Defects

Report Generation Status: Complete

Report Options:

Summarize Information by: My defects | Date Field: Closed Date | Range: Custom

Time Frame: From: | To: |

Run Report | Hide Details | Customize | Save | Save As | Delete | Printable View | Export Details | Subscribe

Filtered By: Edit | Status equals Closed Clear

Defect	Defect Name	Test Case	Category	Severity	Status
Defect5	TC-002	Design	3-Medium	Closed	
Defect10	TC-004	Requirement	5-Cosmetic	Closed	
Defect11	TC-007	Requirement	4-Low	Closed	

Grand Totals (3 records)

Confidential Information - Do Not Distribute

Copyright © 2000-2017 salesforce.com, Inc. All rights reserved. | Privacy Statement | Security Statement | Terms of Use | SSO Compliance

Deferred Defects

The screenshot shows the Salesforce 'Deferred Defects' report page. The report generation status is 'Complete'. The report options include 'Summarize information by' set to 'None', 'Show' set to 'All defects', and 'Time Frame' set to 'Closed Date' with a 'Range' of 'Custom'. The 'Run Report' button is visible. Below the filters, the report is filtered by 'Egil' and shows a table of defects.

Defect	Defect Name	Test Case	Category	Severity	Status
Defect1	TC-001	Environment	3-Medium	Deferred	
Defect2	TC-004	Code	3-Medium	Deferred	
Defect3	TC-005	Configuration	1-Critical	Deferred	

Grand Totals (3 records)

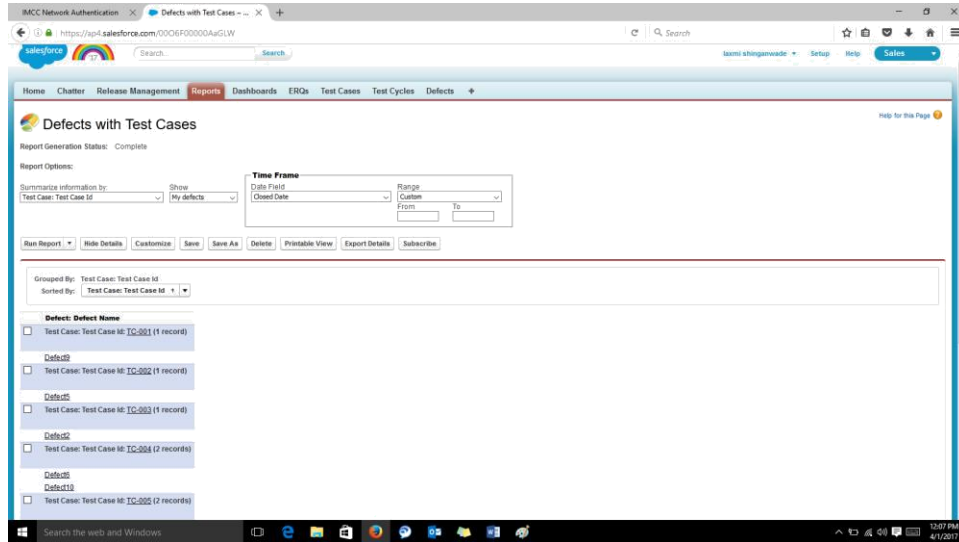
Monthly Releases

The screenshot shows the Salesforce 'Monthly Releases' report page. The report generation status is 'Complete'. The report options include 'Summarize information by' set to 'None', 'Show' set to 'My release management', and 'Time Frame' set to 'Deployment Date' with a 'Range' of 'Custom'. The 'Run Report' button is visible. Below the filters, the report shows a list of releases.

Release Management	Release
Release2	Release2
Release3	Release3
Release1	Release1

Grand Totals (3 records)

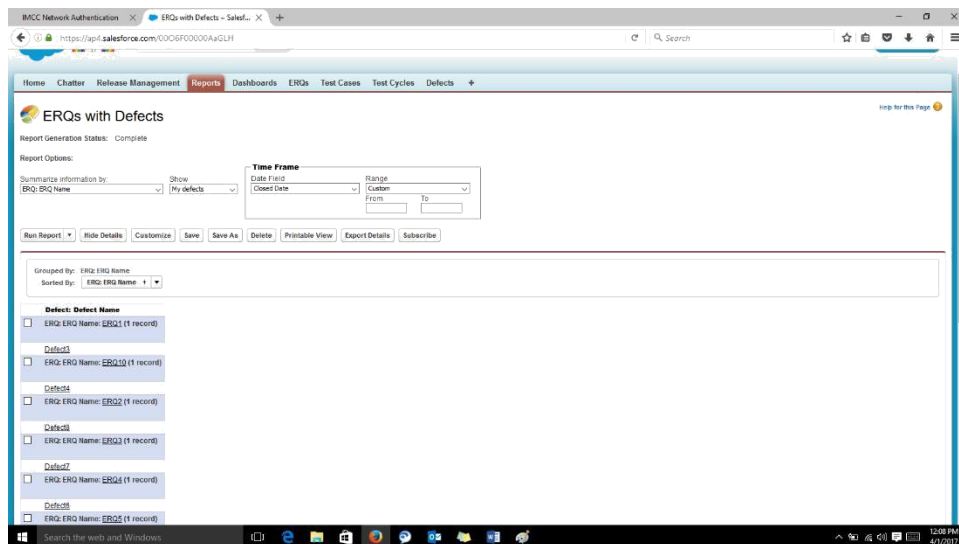
Defects With Test Cases



The screenshot shows the Salesforce Reports interface for "Defects with Test Cases". The report generation status is "Complete". The report options include "Summarize information by: Test Case: Test Case Id" and "Time Frame: Date Field: Closed Date, Range: Custom". The report is grouped by "Test Case: Test Case Id" and sorted by "Test Case: Test Case Id". The data is displayed as a list of defects, each associated with a test case ID and a count of records.

Defect Name	Test Case: Test Case Id	Count
Defect1	TC:001	1 record
Defect2	TC:002	1 record
Defect3	TC:003	1 record
Defect4	TC:004	2 records
Defect5	TC:005	2 records

ERQs With Defects



The screenshot shows the Salesforce Reports interface for "ERQs with Defects". The report generation status is "Complete". The report options include "Summarize information by: ERQ: ERQ Name" and "Time Frame: Date Field: Closed Date, Range: Custom". The report is grouped by "ERQ: ERQ Name" and sorted by "ERQ: ERQ Name". The data is displayed as a list of defects, each associated with an ERQ name and a count of records.

Defect Name	ERQ: ERQ Name	Count
Defect1	ERQ: ERQ1	1 record
Defect2	ERQ: ERQ2	1 record
Defect3	ERQ: ERQ3	1 record
Defect4	ERQ: ERQ4	1 record
Defect5	ERQ: ERQ5	1 record

Error Report

Release Management Edit

Error: Invalid Data.
Review all error messages below to correct your data.

Information | = Required Information

Release	SFDC Enhancement	Release Manager IT	laxmi shinganwade
Deployment Date	3/21/2017 [3/22/2017] <small>Error: Deployment Date must not be past date.</small>	Release Manager Business	shrutika joshi
Technology	Salesforce.com	Request Cutoff	3/22/2017 [3/22/2017]
Owner	laxmi shinganwade		

Development

Dev Start Date	3/22/2018 [3/22/2017] <small>Error: Dev Start date must be before Deployment Date and Dev End Date</small>	Dev End Date	3/20/2018 [3/22/2017] <small>Error: Dev End Date must be before Deployment Date</small>
----------------	---	--------------	--

Quality Assurance

QA Start Date	2/22/2018 [3/22/2017] <small>Error: QA Start Date must be after Dev End Date and before Deployment Date</small>	QA End Date	2/20/2018 [3/22/2017] <small>Error: QA End Date must be after QA Start Date and before Deployment date</small>
---------------	--	-------------	---

User Acceptance Testing

UAT Start Date	1/22/2018 [3/22/2017] <small>Error: UAT Start Date must be after QA end date and before Deployment Date</small>	UAT End Date	1/20/2018 [3/22/2017] <small>Error: UAT End Date must be after UAT Start Date and before Deployment Date</small>
----------------	--	--------------	---

New Defect Creation Notification

 Reply  Reply All  Forward  IM



Fri 3/31/2017 11:48 PM

Salesforce Support <laxmi.shinganwade@veritas.c

Notification on creation of new Defect#

To  Laxmi Shinganwade

Hi,

This is to Notify that Defect Defect12 has been created. Below are the additional details regarding the Defect

Defect #:- Defect12
Change Ticket#:-
Release:-
Application:- AppWorx
Status:- New
Priority:- Medium
Severity:- 3-Medium
Category:- Design
Assigned To:- 0056F000006bmgpQAA
Detected By:- 0056F000006bmgpQAA
Summary:-
Description:-
Steps to Replicate:-

Regards,
Veritas IT Quality Services

Defects status change notification

 Reply  Reply All  Forward  IM



Fri 3/31/2017 11:51 PM

Salesforce Support <laxmi.shinganwade@veritas.c

Status Change Notification for Defect#

To  Laxmi Shinganwade



Hi,

This is to Notify that Defect Defect12 status has been changed to Fixed. Below are the additional details regarding the Defect

Defect #:- Defect12

Change Ticket#:-

Release:-

Application:- AppWorx

Status:- Fixed

Priority:- Medium

Severity:- 3-Medium

Category:- Design

Assigned To:- 0056F000006bmgpQAA

Detected By:- 0056F000006bmgpQAA

Summary:-

Description:-

Steps to Replicate:-

Regards,

Veritas IT Quality Services

ANNEXURE 3(Source Code)

DefectHandler.apxc

```
public class DefectHandler {  
    public static string TriggerDispatch(  
        list<Defect__c>newList,  
        list<Defect__c>oldList,  
        map<Id, Defect__c>newMap,  
        map<Id, Defect__c>oldMap,  
        booleanisBefore,  
        booleanisAfter,  
        booleanisInsert,  
        booleanisUpdate,  
        booleanisDelete,  
        booleanisExecuting){  
  
        for(Defect__c defect : newList) {  
  
            String[] toAddresses = new String[] {'example@example.com'};  
  
            List<Messaging.SingleEmailMessage> mails =  
            new List<Messaging.SingleEmailMessage>();  
  
            Messaging.SingleEmailMessage mail = new  
            Messaging.SingleEmailMessage();  
  
            mail.setTargetObjectId(defect.Assigned_To__c);
```

```
mail.setWhatId(defect.Id);

mail.setSenderDisplayName('Salesforce Support');

mail.setUseSignature(false);

String[] sendingToBccAdd = new
String[] { 'laxmi.shinganwade@veritas.com' };

mail.setBccAddresses(sendingToBccAdd);

mail.setSaveAsActivity(false);

if(Trigger.isInsert){

if(defect.Status__c == 'new') {

EmailTemplate et=[Select id from EmailTemplate
where DeveloperName=: 'New_Defect_Notification'];

mail.setTemplateId(et.id);

Messaging.SendEmailResult [] r = Messaging.sendEmail(new
Messaging.SingleEmailMessage[] { mail});

}

}

if(Trigger.isUpdate) {

if(defect.Status__c == 'Open') {
```

```
EmailTemplate et=[Select id from EmailTemplate where
DeveloperName=:Defect_Status_Change_Notifications'];

mail.setTemplateId(et.id);

Messaging.SendEmailResult [] r = Messaging.sendEmail(new
Messaging.SingleEmailMessage[] {mail});

    }

else if(defect.Status__c == 'Assigned') {

EmailTemplate et=[Select id from EmailTemplate where
DeveloperName=:Defect_Status_Change_Notifications'];

mail.setTemplateId(et.id);

Messaging.SendEmailResult [] r = Messaging.sendEmail(new
Messaging.SingleEmailMessage[] {mail});

    }

else if(defect.Status__c == 'Fixed') {

EmailTemplate et=[Select id from EmailTemplate where
DeveloperName=:Defect_Status_Change_Notifications'];

mail.setTemplateId(et.id);

Messaging.SendEmailResult [] r = Messaging.sendEmail(new
Messaging.SingleEmailMessage[] {mail});

    }

else if(defect.Status__c == 'Rejected') {
```

```
EmailTemplate et=[Select id from EmailTemplate where
DeveloperName=: 'Defect_Status_Change_Notifications'];

mail.setTemplateId(et.id);

Messaging.SendEmailResult [] r = Messaging.sendEmail(new
Messaging.SingleEmailMessage[] {mail});

    }

else if(defect.Status__c == 'Reopen') {

EmailTemplate et=[Select id from EmailTemplate where
DeveloperName=: 'Defect_Status_Change_Notifications'];

mail.setTemplateId(et.id);

Messaging.SendEmailResult [] r = Messaging.sendEmail(new
Messaging.SingleEmailMessage[] {mail});

    }

else if(defect.Status__c == 'Deferred') {

EmailTemplate et=[Select id from EmailTemplate where
DeveloperName=: 'Defect_Status_Change_Notifications'];

mail.setTemplateId(et.id);

Messaging.SendEmailResult [] r = Messaging.sendEmail(new
Messaging.SingleEmailMessage[] {mail});

    }

else if(defect.Status__c == 'Closed') {
```

```
EmailTemplate et=[Select id from EmailTemplate where
DeveloperName=:Defect_Status_Change_Notifications'];

mail.setTemplateId(et.id);

Messaging.SendEmailResult [] r = Messaging.sendEmail(new
Messaging.SingleEmailMessage[] {mail});

    }

    }

    mails.add(mail);

    }

return null;

    }

}
```

DeffectTriggers.apxt

triggerDeffectTriggers on Defect__c (after insert, after update)

```
{ DefectHandler.TriggerDispatch( trigger.new,
```

```
trigger.old,
```

```
trigger.newMap,
```

```
trigger.oldMap,
```

```
trigger.isBefore,
```

```
trigger.isAfter,
```

```
trigger.isInsert,
```

```
trigger.isUpdate,
```

```
trigger.isDelete,
```

```
trigger.isExecuting);
```

If an error happened, show it to the

```
user. //f (errMsg != null) {
```

```
    if (trigger.isInsert || trigger.isUpdate || trigger.isUndelete)
```

```
    { trigger.new[0].addError(errMsg); } else {
```

```
    trigger.old[0].addError(errMsg); }
```