**DevOps - Automation**

**Continuous Integration and Continuous Delivery (CI/CD)**

**A PROJECT REPORT**

*Submitted in partial fulfilment for the award of the degree*

*of*

**Master of Science**

***in***

**Information Technology**

*by*

**SHRUTI CHAKRANARAYAN**

**(14MIN2456)**

*Under the guidance of*

**Dr./Prof. PRANSHU SAKALLEY**

**VIT**

****



**School of Information Technology and Engineering**

June, 2018

****

**DECLARATION BY THE CANDIDATE**

I hereby declare that the thesis entitled **“SUGAR CRM – INVESTMENT BANKING”** submitted by me to Vellore Institute of Technology, Vellore, in partial fulfillment of the requirement for the award of the degree of **Master of Technology** in **Information Technology** is a record of bonafide project work carried out by me under the supervision of **Pranshu Sakalley**. I further declare that the work reported in this project has not been submitted and will not be submitted, either in part or in full, for the award of any other degree or diploma in this institute or any other institute or university.

**Place**:

**Date**: **Signature of the Candidate**

**School of Information Technology and Engineering**

****

**BONAFIDE CERTIFICATE**

This is to certify that the project work entitled “**SUGAR CRM INVESTMENT BANKING” by** **Shruti Chakranarayan (14MIN2456),** to Vellore Institute of Technology, Vellore, in partial fulfillment of the requirement for the award of the degree of **Master of Technology** in **Information Technology**, is a project bonafide work carried out by him/her under my supervision. The project fulfills the requirement as per the regulations of this Institute and in my opinion meets the necessary standards for submission. The contents of this report have not been submitted and will not be submitted either in part or in full, for the award of any other degree or diploma in this Institute or any other Institute or University.

**Prof/Dr.Pranshu**

**Sakalley**

**Internal Supervisor**

**VIT**

**Internal Examiner(s) External Examiner(s)**

**Contents**

[1. Introduction vi](#_gjdgxs)

[1.1 CONFLICTS MODULE vi](#_30j0zll)

[1.2 UHNW vi](#_1fob9te)

[2. Issues in existing system vi](#_3znysh7)

[3. Proposed Solution vi](#_2et92p0)

[3.1. OBJECTIVES: vii](#_tyjcwt)

[4. Software and hardware specifications vii](#_3dy6vkm)

[5 Architecture Design viii](#_1t3h5sf)

[5.1 ARCHITECTURE DIAGRAM viii](#_4d34og8)

[5.1.1 UHNW viii](#_2s8eyo1)

[5.1.2 Conflicts Workflow ix](#_17dp8vu)

[5.2Use Case Diagrams x](#_3rdcrjn)

[5.2.1When to Use: Use Cases Diagrams x](#_26in1rg)

[5.3 Class Diagrams xi](#_lnxbz9)

[5.3.1 When to Use: Class Diagrams xii](#_35nkun2)

[5.4 Sequence diagrams: xii](#_1ksv4uv)

[6. Implementation xiv](#_44sinio)

[6.1STRATEGIC APPROACH TO SOFTWARE TESTING xv](#_2jxsxqh)

[6.2 SYSTEM TESTING xv](#_z337ya)

[6.2.1 Implementation and Evaluation xvi](#_3j2qqm3)

[6.2.2 Operation Evaluation: xvi](#_1y810tw)

[6.2.3 User Manager Assessment: xvi](#_4i7ojhp)

[6.2.4 Development Performance xvi](#_2xcytpi)

[6.3 System Implementation components include: xvi](#_1ci93xb)

[7. Test Plans xvii](#_3whwml4)

[7.1Test Objectives xvii](#_2bn6wsx)

[7.2 Extent of Tests xvii](#_qsh70q)

[8. Test design and Strategies xviii](#_3as4poj)

[8.1 General Test Strategy xviii](#_1pxezwc)

[8.2Integration Test Strategy xix](#_49x2ik5)

[8.2.1Test Case alignment with test phases xix](#_2p2csry)

[8.3Suspension and resumption xix](#_147n2zr)

[8.4 Automated Unit Test Suite xx](#_3o7alnk)

[9 Module Description xx](#_23ckvvd)

[9.1 Test Cases xx](#_ihv636)

[10. Reports xxvi](#_32hioqz)

[11. Abbreviations xxvi](#_1hmsyys)

## 1. Introduction

## 1.1 CONFLICTS MODULE

The Conflict Review process is a mechanism required by the institution in order to validate (or reject) whether CS can transact on behalf of a client and/or on the deal. The Conflict Review process captures information when determining whether or not a specific deal or project can be pursued due to a conflict of interest. This conflict of interest could be another deal that is currently underway a conflict, or has the potential to damage CS reputation. The conflict review process can be initiated form Horizon Desktop via a system integration of by the creation of a record with Sugar system and a conflict record is created. If a banker calls the conflicts officer directly the CO will create the record in sugar.

## 1.2 UHNW

Ultra high net worth product group in IBCM deals with the process which involves some background verification of client. If the client wants to invest money in the bank we need to perform some checks whether he is doing some gaming, gambling business or is he politically exposed person and few other factors. Based upon the approval provided by the ultra high net worth group, it is decided that client can do business with Credit Suisse or not.

## 2. Issues in existing system

A combination of tools and process are used by Credit Suisse in order to facilitate support and document the conflict review process.

The conflict review process requires coordination and collaboration among different people and departments, performed by the conflicts officer (CO). The current system (Spider) charged with supporting these processes is a 15 year old system that lacks the flexibility and usability required to support the current and future needs of the business. Spider forces the CO to capture the majority of the investigation, conversations and additional client information within a single notes field in spider.

## 

## 3. Proposed Solution

The vision is to make the sugar platform a key component within the horizon family of initiative. All designed and developed to support the business effort of credit Suisse.

The conflict review process is the first part of making the vision a reality. Many of the in system actions and tasks required for conflict review performed by a CO will be enhanced , such as management of tasks, method of assignment and how information is entered, stored, searched and retrieved. The information structure and specific activities will be modernized around the sugar platform. The platform is designed to support not only the as –ease processes, but also flexibility for yet to be determined processes. Within phase1, the new system will preserve the existing capability as a baseline.

## 3.1. OBJECTIVES:

The business objectives for gathering requirements and implementing on the sugar platform are to:

* Migrate the conflicts review process from Spider to Sugar
* Migrate historical conflict review information from spider to sugar
* Provide audit trail to support compliance requirements
* Deliver TAD level view of conflicts approval and status.
* Minimize the duplication of effort in data entry and communication
* Support current as ease business process for Conflict review.

## 4. Software and hardware specifications

**Hardware:**

Processor : Intel Pentium III or Above

Ram : 256 MB or more

Cache : 512 KB

Hard disk : 16 GB hard disk recommended for primary partition.

**Software:**

Operating system : Windows XP or later

Front End Software : JAVA, HTML

Back End Software : SQL Server 2005

* Postman or Newman
* Java
* Selenium
* Oracle DB
* Jmeter
* Junit

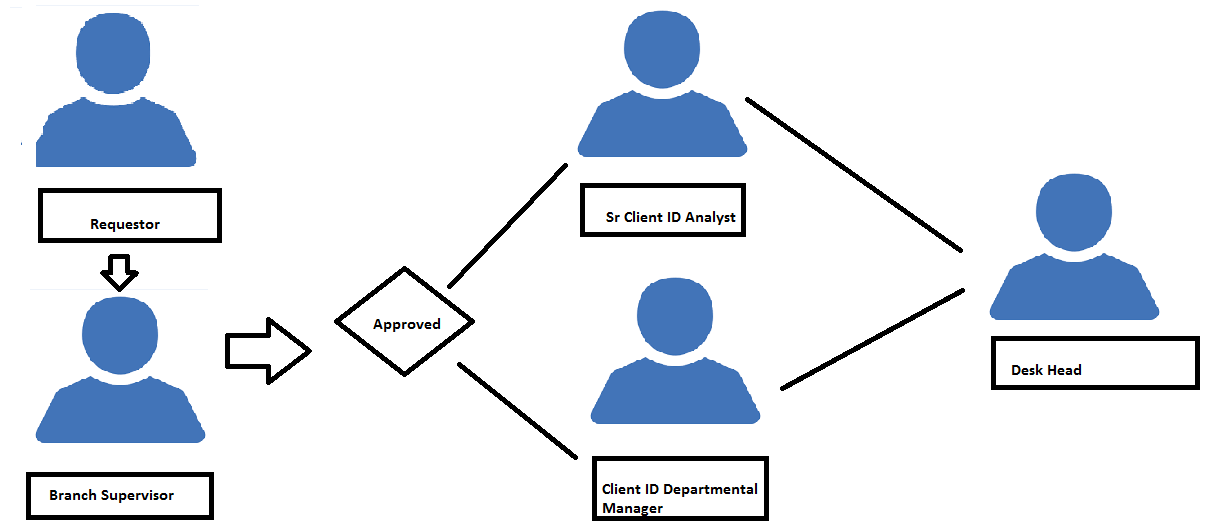
## 

## 5 Architecture Design

## 5.1 ARCHITECTURE DIAGRAM

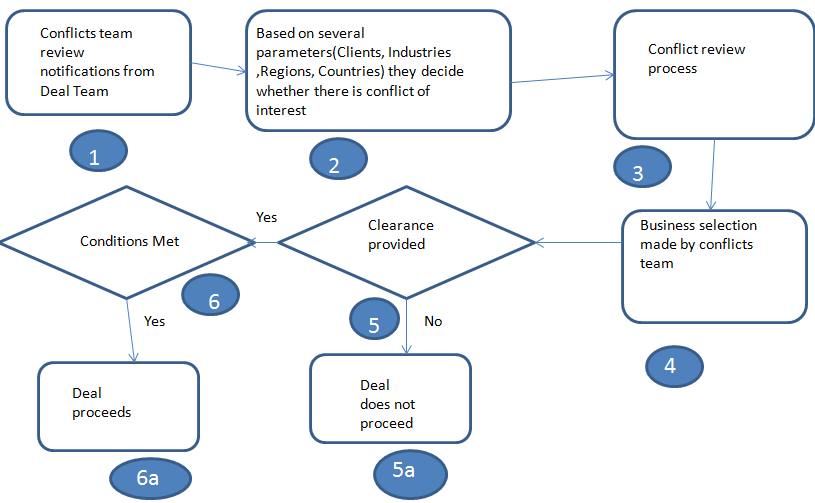
## 5.1.1 UHNW

UHNW has five user roles namely Requestor, Branch Supervisor, Sr Client ID Analyst, Client ID departmental Manager and Desk head



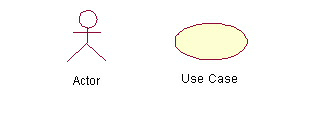
## 

## 5.1.2 Conflicts Workflow

****

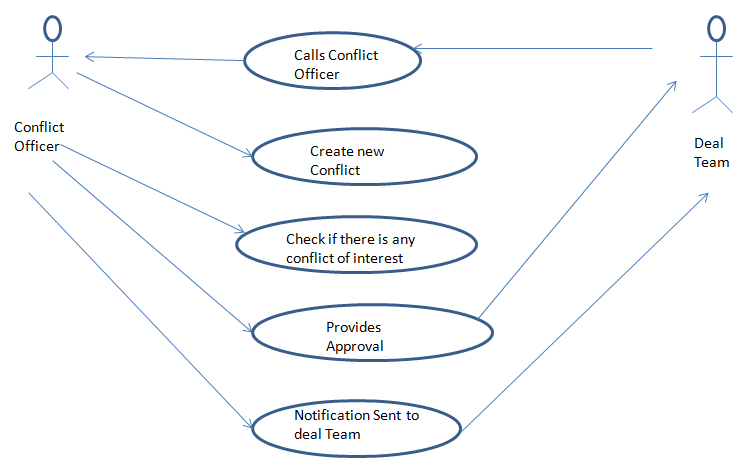
## 5.2Use Case Diagrams

A use case is a set of scenarios that describing an interaction between a user and a system.  A use case diagram displays the relationship among actors and use cases.  The two main components of a use case diagram are use cases and actors.



## 5.2.1When to Use: Use Cases Diagrams

Use cases are used in almost every project.  They are helpful in exposing requirements and planning the project. During the initial stage of a project most use cases should be defined, but as the project continues more might become visible.

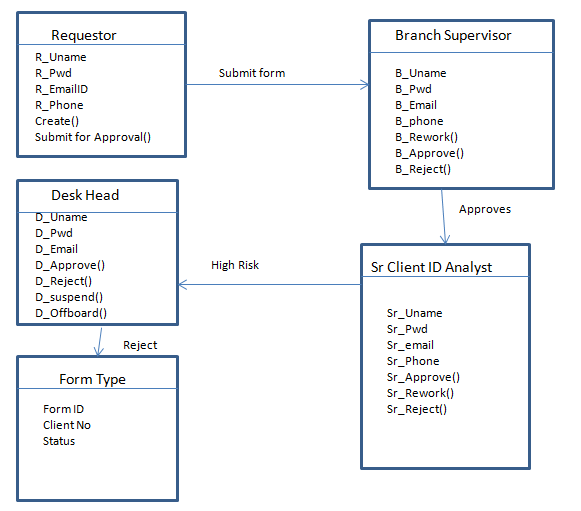


## 5.3 Class Diagrams

Class diagrams are widely used to describe the types of objects in a system and their relationships.  Class diagrams model class structure and contents using design elements such as classes, packages and objects.  Class diagrams describe three different perspectives when designing a system, conceptual, specification, and implementation. These perspectives become evident as the diagram is created and help solidify the design.  This example is only meant as an introduction to the UML and class diagrams.  If you would like to learn more see the Resources page for more detailed resources on UML.

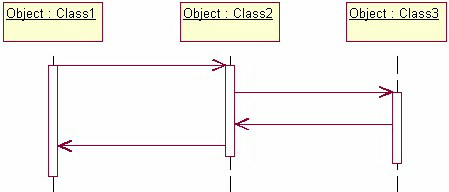
## 5.3.1 When to Use: Class Diagrams

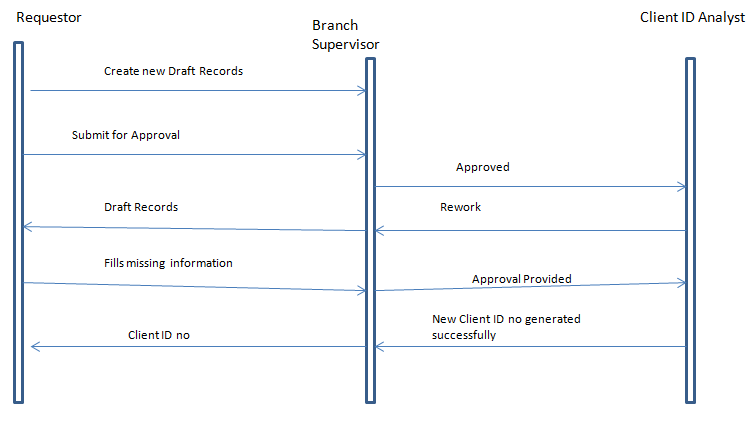
Class diagrams are used in nearly all Object Oriented software designs. Use them to describe the Classes of the system and their relationships to each other.



## 5.4 Sequence diagrams:

Sequence diagrams demonstrate the behavior of objects in a use case by describing the objects and the messages they pass.  The diagrams are read left to right and descending.  The example below shows an object of class 1 start the behavior by sending a message to an object of class 2.  Messages pass between the different objects until the object of class 1 receives the final message.





## 6. Implementation

Software testing is a critical element of software quality assurance and represents the ultimate review of specification, design and coding. In fact, testing is the one step in the software engineering process that could be viewed as destructive rather than constructive.

A strategy for software testing integrates software test case design methods into a well-planned series of steps that result in the successful construction of software. Testing is the set of activities that can be planned in advance and conducted systematically. The underlying motivation of program testing is to affirm software quality with methods that can economically and effectively apply to both strategic to both large and small-scale systems.

## 

## 6.1STRATEGIC APPROACH TO SOFTWARE TESTING

The software engineering process can be viewed as a spiral. Initially system engineering defines the role of software and leads to software requirement analysis where the information domain, functions, behaviour, performance, constraints and validation criteria for software are established. Moving inward along the spiral, we come to design and finally to coding. To develop computer software we spiral in along streamlines that decrease the level of abstraction on each turn.

A strategy for software testing may also be viewed in the context of the spiral. Unit testing begins at the vertex of the spiral and concentrates on each unit of the software as implemented in source code. Testing progress by moving outward along the spiral to integration testing, where the focus is on the design and the construction of the software architecture. Talking another turn on outward on the spiral we encounter validation testing where requirements established as part of software requirements analysis are validated against the software that has been constructed. Finally we arrive at system testing, where the software and other system elements are tested as a whole.

## 6.2 SYSTEM TESTING

During system testing the system is used experimentally used to ensure that the software does not fail, i.e., it will run according to its specification and in the way the users expect. Special test data are input for processing and the results examined. A limited number of users may be allowed to use the system to see whether they try to use it in unforeseen ways. It is preferable to discover any surprises before the organization implements the system.

## 6.2.1 Implementation and Evaluation

Implementations is the process of having systems personnel checkout and put new equipment into use, train users, installs the new application and construct any files of data needed to use it.

Evaluation of the system is performed to identify its strengths and weakness. The actual evaluation can occur along any of the following dimensions.

## 6.2.2 Operation Evaluation:

Assessment of the manner in which the system functions, including ease of use, response time, suitability of information formats, overall reliability and level of utilization.

Identification and measurement of benefits to the organization in such areas as financial concerns operational efficiency, and competitive impact. Includes impact on internal and external information flows.

## 6.2.3 User Manager Assessment:

Evaluation of the attitudes of senior and user managers within the organization, as well as end-users.

## 6.2.4 Development Performance

Evaluation of the development process in accordance with such yardsticks as overall development time and effort, conformance to budgets and standards, and other project management criteria.

System Implementation components

## 6.3 System Implementation components include:

**Personal Orientation:**

Introduce people to the new system and their relationship to the system

**Training:**

Give employees the tools and techniques to operate and use the system.

**Hardware Installation:**

Schedule for, prepare for, and then actually install new equipment.

**Procedure Writing:**

Develop procedure manual to follow in operating the new system.

**Testing:**

Ensure that the computer programs properly process the data.

**File Conversion:**

Load the information of the present files onto the new system files.

**Parallel Operation:**

Use the new system at the same time, as the old to make sure results are correct

## 7. Test Plans

The purpose of this document is to outline the test strategy and overall test approach for the AST Identity project. This includes test methodologies, traceability, and resources required, and estimated schedule.

## 7.1Test Objectives

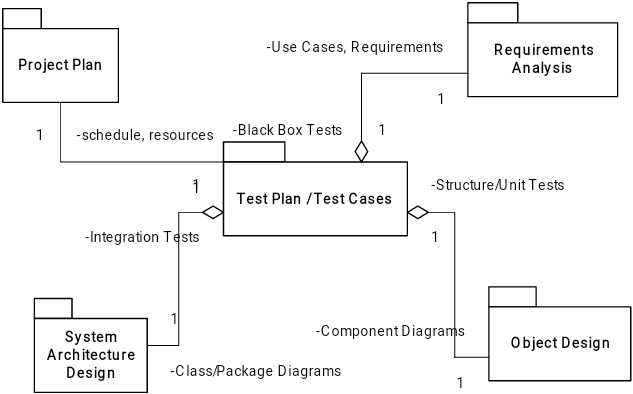
The main objective of testing is to uncover a host of errors, systematically and with minimum effort and time. Stating formally, we can say,

* Testing is a process of executing a program with the intent of finding an error.
* A successful test is one that uncovers an as yet undiscovered error.
* A good test case is one that has a high probability of finding error, if it exists.
* The tests are inadequate to detect possibly present errors.
* The software more or less confirms to the quality and reliable standards.

## 7.2 Extent of Tests

The tests referenced herein are written to validate use cases, requirements (both functional and non-functional), system architecture, and object design. The structured tests for object design will be run first as the components of the system are developed. The structured tests to validate the system architecture will be run next as the system is integrated in bottom-up fashion during integration test.

A visualization of the relationships to the other documents can be seen in the diagram below.



## 8. Test design and Strategies

## 8.1 General Test Strategy

Unit testing and component testing will be performed on the components as they are developed. Test will be executed using test code in the form of either custom test tools or as an automated suite of tests run against the components in their individual sandboxes.

Integrations tests will be performed by both the component testers as well as the system testers. However, as the integration begins to include GUI level functionality, the tests being run will utilize significantly more manual testing and less automated testing.

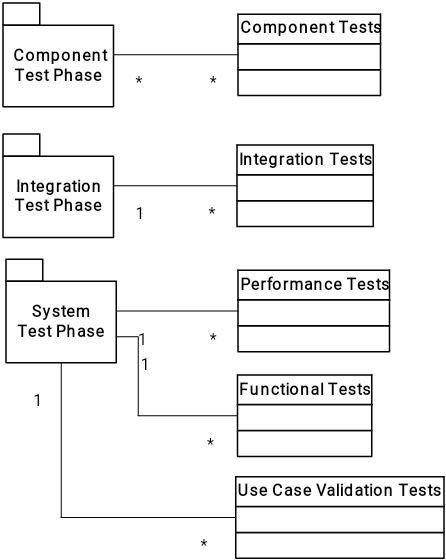
System test will require a new set of tools that can measure NFRS compliance, such as Jmeter (performance testing) or Quality Center (java source code analysis for security issues). Manual tests will start by validating functionality based on the requirements. Later stages of system test will include manual end-to-end tests to validate use cases.

## 8.2Integration Test Strategy

Because the components will be developed from the bottom-up and top-down, the test strategy will also align to the order of development of components. This will utilize a mostly bottom-up integration test approach, but will also involve the sandwich integration test approach.

Please review the system architecture overview and the subsystem architecture in architecture section

## 8.2.1Test Case alignment with test phases



# 8.3Suspension and resumption

This section specifies the criteria for suspending the testing on the test items associated with the plan. It also specifies the test activities that must be repeated when testing is resumed.

## 8.4 Automated Unit Test Suite

As components are being developed, unit tests will be developed to test the interfaces of the components and low-level unit tests will be developed to test the methods of the underlying classes in the components.

When the unit-test suite reports failures, testing will not occur on that build until the failures have been analyzed and resolved. Testing will resume on a build that passes the automated unit test suite.

## 9 Module Description

## 9.1 Test Cases

Scenario: To verify that Conflict officer is able to create new conflict record in Sugar.

|  |  |  |
| --- | --- | --- |
| Steps | Steps Description | Output |
| Step 1 | Login to Sugar QA | Login is successful |
| Step 2 | Go to Conflicts Review ->Create Conflicts Review | New Conflicts form should be opened |
| Step 3 | When CO fills all the Details in Section 1: CS Division Transaction Type M&A Transaction Type Company Client Role Primary Project Secondary Project Lead Banker Secondary Banker Regions Countries Industries SpecialCategories Cross Border | All the details should be entered successfully |
| Step 4 | When CO fills all the details in Transaction Details section Status Transaction Notes Transaction Size Estimated Gross Fee Pitch Date Mandate Date Announced Date Completed Date Lost Date Dead Date NDA CA Status Engagement Letter Status Fairness Opinion Status Hostile/Friendly CS Role  Process | All the details should be entered successfully |
| Step 5 | When CO fills Clearance Notes and Code Name | Entry date should be populated |
| Step 6 | When user fills all the details in Clearance Instructions: Clearance Status Clearance Conditions Clearance Conditions Date  Clearance Instructions Banker Submission Notes | All the details should be entered successfully |
| Step 7 | When CO click on Save button | Then Conflict should be created successfully |

Scenario :To verify that CO if able to filter clients with Client name

|  |  |  |
| --- | --- | --- |
| Step 1 | Login to Sugar QA | Login is successful |
| Step 2 | Go to Conflicts Review ->Client search New | Client Search Screen should be opened |
| Step 3 | When user selects Starts with radio button and select name from the filter and press enter key | All the clients should be loaded whose name starts with Name entered |
| Step 4 | When user selects contains radio button and select name from the filter and press enter key | All the clients should be loaded whose name contains Name entered |

Scenario : To verify that CO if able to filter clients with Client Code

|  |  |  |
| --- | --- | --- |
| Step 1 | Login to Sugar QA | Login is successful |
| Step 2 | Go to Conflicts Review ->Client search New | Client Search Screen should be opened |
| Step 3 | When user selects Starts with radio button and select Client Code from the filter and press enter key | All the clients should be loaded whose Client Code starts with Client Code entered |
| Step 4 | When user selects contains radio button and select Client Code from the filter and press enter key | All the clients should be loaded whose Client Code contains Client Code entered |

Scenario : To verify that CO if able to filter clients with RIC Ticker

|  |  |  |
| --- | --- | --- |
| Step 1 | Login to Sugar QA | Login is successful |
| Step 2 | Go to Conflicts Review ->Client search New | Client Search Screen should be opened |
| Step 3 | When user selects Starts with radio button and select RIC Ticker from the filter and press enter key | All the clients should be loaded whose RIC Ticker starts with RIC Ticker entered |
| Step 4 | When user selects contains radio button and select RIC Ticker from the filter and press enter key | All the clients should be loaded whose RIC Ticker contains RIC Ticker entered |

Scenario: TO verify that error message should be displayed while linking if project is already linked to a conflict

|  |  |  |
| --- | --- | --- |
| Step 1 | Login to Sugar QA | Login is successful |
| Step 2 | Go to Conflicts Review ->Project Reconciliation Dashboard | PRD screen should be opened |
| Step 3 | Select and project and status and select Conflicts linked as Yes | Projects details should be opened |
| Step 4 | In Screen two click on Link button to link a conflict to a project | Verify the error message 'Project cannot be linked as it is already linked to some other conflict Record' |

Scenario : TO verify that error message should be displayed while creating if project is already linked to a conflict

|  |  |  |
| --- | --- | --- |
| Step 1 | Login to Sugar QA | Login is successful |
| Step 2 | Go to Conflicts Review ->Project Reconciliation Dashboard | PRD screen should be opened |
| Step 3 | Select and project and status and select Conflicts linked as Yes | Projects details should be opened |
| Step 4 | In Screen two click on create button to create new conflict | Verify the error message 'Project cannot be created as it is already linked to some other conflict Record' |

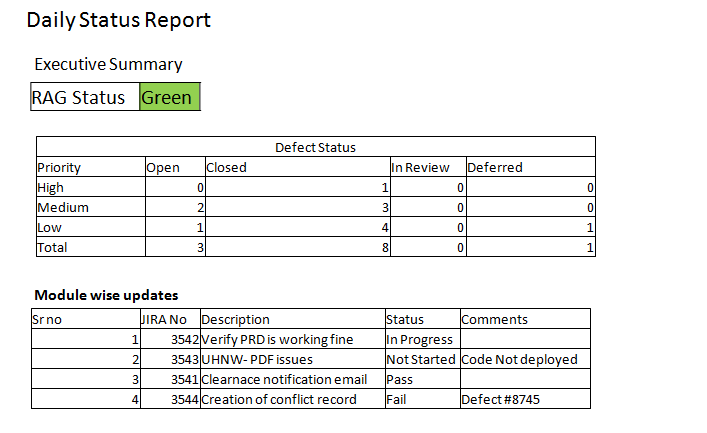
Scenario : To verify that UHNW requestor is able to create new draft records

|  |  |  |
| --- | --- | --- |
| Step 1 | Login to Sugar QA | Login is successful |
| Step 2 | Go to UHNW -> Create UHNW | New UHNW form should be opened |
| Step 3 | When Reqestor fills all the details in CS Contact Section CS Referral Contact1 Contact2 How was the Client obtained? Sales/Team Coverage | All the fields should be filled successfully |
| Step 4 | When Reqestor fills all the details in Customer Identification Information Entity or Individual? Spider ID Project Code Client Mnemonic First Name Last Name Middle Name Suffix Country of Birth DOB/Incorporation Date Government ID Type Government ID Identification Tax ID ype Tax ID Identification Tax ID no Address 1 Address2 City State Zip  Cuntry of Legal Domicile Email Address | All the fields should be filled successfully |
| Step 5 | When Reqestor fills all the details in Financial Information Section Employement Status Industry Years Employed Address1 address2 Address3 Source of wealth  Investable Assets Net Worth Valuation Comments | All the fields should be filled successfully |
| Step 6 | When UHNW requestor fills Risk as High or Low | Risk should be entered |
| Step 7 | When UHNW requestor fills all the Mandatory questions in EDD section | Alls the questions shouls be selected |
| Step 8 | When UHNW Requestor fills all the details in Financial Crime Risk Appetite Section | Details should be filled |
| Step 9 | Select the PEP staus: Yes No In Process | PEP status should be entered |
| Step 10 | After filling all the details clcik on save button | For should be create successfully and client ID no should be generated |

Scenario: To verify that Sr Client ID Analyst should be able to Approve, Reject or send the form for rework

|  |  |  |
| --- | --- | --- |
| Step1 | Login to Sugar QA as Sr Client ID Analyst | Login is successful |
| Step2 | Go to UHNW -> View UHNW | View Screen should be opened |
| Step3 | Select record with status 'Submitted to Client ID and click on edit | Form should be opened in edit mode |
| Step4 | In Review Approval Section Select : Approved Rework Reject | If Approved Form is sent to Cleint ID team. If Rewok then it is sent to Requestor team. If rejected then status is shown as rejected |

## 10. Reports



## 11. Abbreviations

CO – Conflict Officer

UHNW- Ultra High Net Worth

PRD- Project Reconciliation Dashboard