“A supporting conceptual plugin for manage bugs and change requirements in agile software development”

P U K Perera
179474P

Supervisor: Mr. Charmen Wijesiriwardhana

Dissertation submitted to the Faculty of Information Technology, University of Moratuwa, Sri Lanka for the fulfillment of the requirements of Degree of Master of Science in Information Technology.

June 2020

Declaration

I declare that this thesis is my work and has not been submitted in any form for another degree or diploma at any university or other institution of tertiary education. Information derived from the published or unpublished work of others has been acknowledged in the text and a list of references is given.

Name of Student: Miss P U K Perera		Student ID – 179474P

Signature of Student: ………………………

Date: …………………………………

Supervised by

Name of Supervisor: Mr. Charmen Wijesiriwardana

Signature of Supervisor: …………………

Date:………………

Acknowledgments

First of all, I would like to thank my supervisor Mr. Charmen Wijesiriwardhana for his valuable contribution to this thesis through insightful discussions, guidance, and support throughout the process. And his expert advice and feedback on my thesis.

In addition to all of the other lectures that give their guidance and advice to improve the quality of this thesis.

I must acknowledge the contributions of the large supporting cast from the organizations involved in this research; the people who were interviewed, allowed access to their work artifacts, speculated about the challenges they were facing as well as share their views regarding the role of collaborative technologies. Without their support and co-operation, there would have been nothing to report.This thesis would not have been possible without the support of my parents. Your continuous encouragement, support

Finally hope this thesis has benefited from the help and support from several people who deserve my respect and acknowledgment. I am grateful to everyone who has been directly or indirectly involved in this process.

Table of Contents

List of Tables

List of Figures

Abstract

This thesis reports a comprehensive investigation of the modern software industry, it is widely considered the concept of software as a service. In this concept, they always use the agile methodologies, and they get involved with customer representatives and valued their ideas for the final product. Even after completing their projects, the companies are agreed to do the changes according to user personas and the dependencies of the system environments by getting some agreements. At this point requirements of the system start evolving to address new business needs, usability study results, changing assumptions. As a result, the project scope may extend for quite several reasons. The modern world is changing very fast, and today fixed project requirements are a thing of the past. So this identified the challenges encountered in managing requirements change and investigates supports of the project management tools to simplify the identification of **Bugs and Change Requirements**

CHAPTER ONE
INTRODUCTION

With the current complexity of the industry software, teams need to be aware of the client's feedback. Those customers will report the changes and the existing issues both in the same manner because they don’t have a proper idea of the development process. They will think it is just a small change, but considering the development process, it will make a huge impact on the final product. At the same time, it will make a huge difference with the final cost. In some cases, these reported points will not give with the finalized SRS. On the other hand, the customer has already reported it but from the development team, it will miss or misunderstood what they requested. So we have to consider the deviations against the already reported requirements as Bugs and Non-reported requirement changes as Change Requirements.

When they filter out these 2 scenarios, it will make many conflicts between the clients and the team. In some situations for the customer side, they will miss their records what they have finalized with the team. In some situations, the team may also be missing the things that customers have reported. When they have started the project, they will be agreed with the SRS and then they will report the issues through the emails. Sometimes they will change some requirements within the meetings, discussions, or group calls. So there may be a big possibility of missing some points. And also this will happen due to some resource changes within both sides.

So within this kind of situation change requirement management play a huge role within the software industry. When analyzing the current situation most widely used software management tool is identified as Jira, it is easily able to manage project task handling. Time estimations. But when managing some project related activities such as write and execute test cases it needs to add some sperate plugins. Same as they provided some extra plugins to their tool. But any of these plugins are unable to Differentiate the **Change Requirements** and **Bugs** automatically. Even they already had tool Confluence it is unable to track the requirements on component-wise and do a comparison. So it is unable to produce a proper report on Change requirement management. So I have identified this as a main limitation of Jira.

And at the same time, I have identified some collection of issues related to Software Change Requirement Management in the software industry. Then have categorized then into the main 7 scenarios

Issue 1

Issues regarding reporting the client feedbacks in the concept of Agile Software Development and the Software as a service

a. The client will use different formats to report their feedbacks (group calls, discussions, emails, etc)
b. When reporting through the emails, handling emails will be a huge mess.
c. Maybe having some language-related issues
d. The client will have a lack of knowledge with the system

Issue 2

Issues regarding handling the old existing Requirements Vs New Reported Issues, And compare and contrast them

a. Usually, the SRS is written in document format
b. When analyzing the issues the team has to read all point very carefully
c. This will be a time wastage

Issue 3

Issues regarding defining parameters and standards to filter out Bugs and issues.

a. Some conflictions were happening when filtered out change requirements and bugs, due to the uncertainty of the pre-requirements

b. Most projects have no proper accepted standard for this within clients and team

c. Defining CR as a bug will be a loss to the company at the same time defining bug as a CR will be a loss to the client.

Issue 4

Issues regarding getting clarifications within a global team that spread around different geo-locations.

a. In most situations, current software teams have distributed teams

b. Different team members are in different locations, Maybe they have outsourced resources or virtual teams.

c. Even they used project management tools, they have some limitations with the CR management sector.

Issue 5

Issues regarding finding out impact analysis.

a. If there are new resources without having the proper idea about the pre-requirements and Changes they are unable to define actual impact analysis

Issue 6

Varying the time estimations due to the wrong impact analysis.

a. The team have to work hard and achieve tuff schedules

b. Overstressed teams

Issue 7

Having some arguments and Conflictions within the team

a. They will have some doubts about the responsible person
b. Have no proper idea on, from whom it missed and all the blams comes to the QA team

So Finally, I'm expecting to propose a ***Web-based plugin for Change Requirement Management*** on Jira project management tool to support in mange, the customer reported issue handling, decision making for change requirements, impact analysis and make target estimations and make the development process more efficient.

1.1 Research Background

In the modern software industry, it is widely considered the concept of software as a service. In this concept, they always use the agile methodologies, and they get involved with customer representatives and valued their ideas for the final product. Even after completing their projects, the companies are agreed to do the changes according to user personas and the dependencies of the system environments by getting some agreements. At this point requirements of the system start evolving to address new business needs, usability study results, changing assumptions. As a result, the project scope may extend for quite a several reasons. The modern world is changing very fast, and today fixed project requirements are a thing of the past.

The dynamic and competitive nature of the modern business makes very frequent changes due to the evolution of business needs. So it makes ways to gives rise to new and changing software requirements.

Nowadays all the requirements are aligned based on customer needs. But it occurs many issues when we communicate with the customers.

Because they are coming from different backgrounds such as medical, finance, Educational, etc. so they are unable to express their idea to the development team. So, it is more important to guided all people to speak the global language, So to achieve this the better solution is to use a global web-based tool. When analyzing the current industry most used tools can find some of the common points with the **JIRA**. So seems it will more be profited to software companies and easily market a plugin to an existing solution other than introducing a new solution. Since the **JIRA** the web-based solution already follows this method to Some other processes here also find out it will be the best solution to solve this issue. Here are some options that **JIRA** has provided to users. And This solutions will be added among the Project Management Plugins

Jira Test Management Plugins, Allows users to create a test case, mange test cycles, execute them, edit and update test steps easily. Get the reviews, get the approvals, Copy, Clone, Move, Manage QA sprints, get the summary reports.

Jira Project Management Plugins add the tasks on separate requirements, do the allocations, manage sprint Wise kanbun bord. And this easily manageable to identify the task that needs to do, in progress, completed, in QA and Done very easily and also Jira Time Tracking Plugin handles the time estimations and gets the charts and reports to presenting manner easily

So according to the above mention plugins, the researcher has followed the same manner to implement the solution to the identified problem on automating the method of filter out the bug and CR

1.2 Aims And Objectives

Aims

To manage the customer reported issue handling, decision making for change requirements, impact analysis, and make target estimations. Through all these this will make the development process more efficient. And to introduce a conceptual new methodology on existing project management tool JIRA. And resolved the main limitation that the management people and the team has to face when dealing with the Bug and managing Change requirements

Objectives

1. Formatted the customer issue reported process according to industrial standards and make usable, easy understand and simple

2. Build a proper understanding of the project's requirements and changes for both clients and the team by reducing the knowledge gap, language issues, misunderstanding.

3. Make a database for customer pre-requirements, the customer reported issues and change requirements, and centralized all the requirements for one manageable point by reducing the duplications

4. Automatically filter out the bugs and change requirement and the summary

5. Automate the impact analysis method with a clear and simplify the idea based on each component

6. Support for decision making about how much the changes will be impacted for the whole system and, how much this should re-check and regression check

7. Make efficient the estimation handling in change requirement management and the issue regarding with the project scope and the resource allocations.

8. Centralized the change requirement management process with a globally distributed team in different geo-locations.

9. This will efficient teamwork and give a better service for the customer with a proper understanding of their change requirements

10. Finally, the researcher needs to introduce this concept to the global software industry by implementing a proper solution with the support of the properly experienced developers.

1.3 Thesis Structure

Chapter 01 – Introduction - This chapter includes a basic introduction to the thesis report structure. It emphasis what will be described in the whole report

Chapter 02 - Background and Literature Review - This chapter has analyzed the existing literature in the area of requirements change management. It has highlighted the challenges and also some limitations need to be resolved

Chapter 03 - Research Design and Methodology - This chapter has described the basics of research. Why the researcher selects this problem. What is the background, How data are gathered, What are the conditions based on this study

Chapter 04 - Analysis and Findings - In this chapter is explained the contextual analysis of the case studied. And considerations of practiced models to resolve change management issues in the selected methods, logical design, logical functionalities. And this described the user wise functionalities and how will be the solution supports to each user. And what should be the instructions to the client when using this solution. And how the client should get the basic idea about the system. Finally, it describes the Software development life cycle that the researcher has used and what methods used in the prototype

Chapter 05 - Implementation - This chapter has discussed the full implementation part of the solution, the development architecture that used for development. How will it be implemented.

Chapter 06 - Conclusions - This final chapter of the thesis has reiterated the motivation for this study into the challenges of managing requirements change management, the assumptions made, cost-benefit analysis and finally how will it get the advantage for the software industry.

1.4 Chapter Summary

This chapter includes a basic introduction to the thesis report structure. It emphasizes what will be described in the whole report. And also the Aims and objectives that hope to achieve by the success of the research

CHAPTER TWO

BACKGROUND AND LITERATURE REVIEW

With the experience of engaging the Software Industry, many issues are popped up when working with the ***Change Requirements***. When managing the changes with an already developed system or a module have been more careful because most of the time team has to work with the clients that don't have a proper idea on the software development process. Most probably they are thinking making a change is a very easy thing, because there is already a working product that completed. But for a software development team, it is a risk because when they are doing a change they should have proper knowledge of the impact analysis. Otherwise, it will make a disadvantage for a company. And they will be lost time, resources and cost. So making the software service more beneficial they should have good practices and good processes. In this point, project management tools make them more efficient and profit

2.1 Literature review

This section is based on the background of the researchers that referred to a critical review of the selected research topic. When analyzing the requirements change management process basic research once a researcher found the basics steps of requirement change management. Initial Change request, prepare change proposal note, evaluate the impact of change, accept or reject and implement the change **[1]**

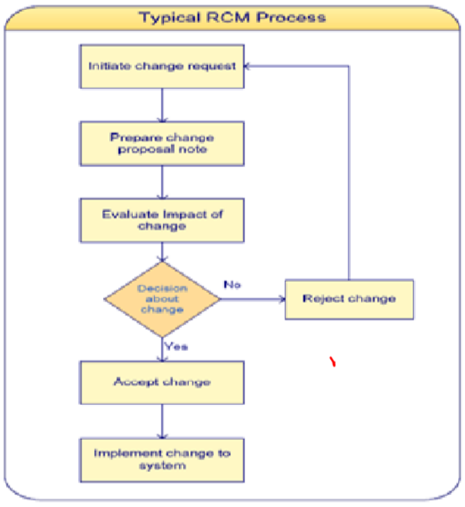


Figure 2.1: Typical RCM process

By based on this basic model there is a different kind of studies has conducted. Among those some are highlighted, there may be several reasons for requirements change. [13]suggests that changes in requirements arise mainly because people change their minds regularly. It happens when stakeholders realize that a) they have missed a requirement needed in the system, b) identified a bug and that has now become a requirement c) the actual needs were not understood initially d) the political landscape has changed giving rise to new priorities that motivate new requirements e) the place has changed d) the new legislation had to be adhered to either by adding new requirements or making changes to the existing system [13] Those studies have pointed out that most of these reasons are distributed among the different parties such as Email played a vital role in collaboration and communication not only among local team members on the client-side but also with the remote team members for sharing requirements and changes to obtain their feedback. They were used as a mechanism to communicate and collaborate 'quickly' within the team as well as with the vendor [2]

On such an environment, some of the researchers are highlighted the Values: a)Individuals and interactions over processes and tools b)Working software over comprehensive documentation c) Customer collaboration over contract negotiation d) Responding to change over following a plan –Principles: a) The highest priority is to satisfy the customer through the early and continuous delivery of valuable software b) Welcome changing requirements, even late the development. Agile processes harness change for the customer's competitive advantage c) Deliver working software frequently, from a couple of weeks to a couple of months, with a preference for the shorter timescale d) Business people and developers must work together daily throughout the project e) Build projects around motivated individuals. Give them the environment and support they need and trust them to get the job done f)The most efficient and effective method by which to convey information to and within a development team is a face-to-face conversation [10]

But there are some situations that these principles are unable to reach, Those situations can identify as the challenges that faced on requirement change management, multitasking architectures, which analyze the impacts of changes more complex. Sharing software parts in several product environments complicates analyzing the impacts of the change. The impact analysis of Enterprise software products must also be efficient, since unwanted impacts may create problems which could result in expensive or irremediable error situations afterward. The module can be used in several different HW and SW environments, and the effects of the modification have to be analyzed in all of them High-reliability demands. When incomplete, not properly tested SW components are integrated, it is extremely difficult to find out the actual root causes for errors. Code optimization [4]

Even modern processes have used some tools among those also there are many identified challenges and limitations. It recognizes that most of the existing processes are not supporting current software change requirement management. For example, most of the tools used manually to specify the impact of the requested change while other processes same like software requirement management do systematically. There is no help for indirect correlation in all tools excluding Top Team Analyst, and in RequisitePro and Caliber only links which are in the same path are used, so a manual check for the effected requirement is still needed[3]. The summary of the issues is a) Currently, the Impact analysis checklist process is not perfectly supported current software projects b)Most of the Solutions are specified with too many details by high-level analysts to perform accurate decisions. c) Usually, Analysis is performed by the wrong persons d) In the tools, there is no option for determining the type of change. e) In the current practices, Change request decisions are based on interest f) Different change requests have different levels of complexity, and there is no strategy for appropriate decision. g) Responsibilities and project balances are difficult to handle for analyses that span several systems.[3]

When increasing the complexity of software development, the researchers have identified the standardization of the processors, So suggested some models, use knowledge for knowledge management Framework for the assessment of the anticipated expert judgment [5] In GSD due to communication issues RCM is difficult to manage. The proposed framework help to improve the understanding of roles, work of art and behavior involved in the GSD particularly from the perception of the change management system.[5] Below table can be given as an example that a pre researchers conclusion on the benefits of using an RCM model

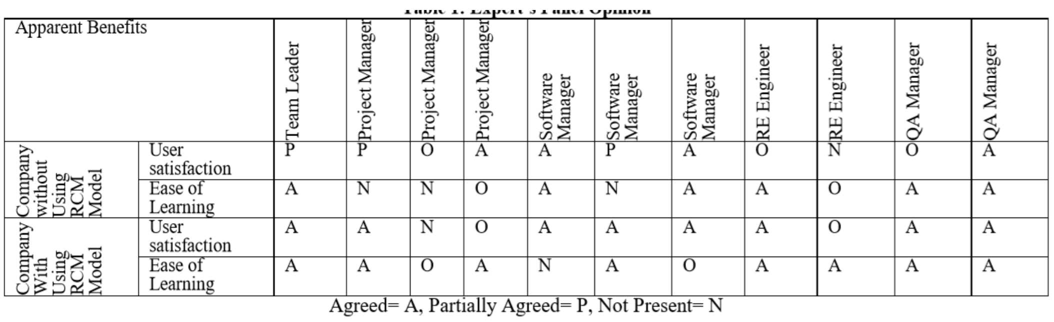


Table 1: Expert's Panel Opinion

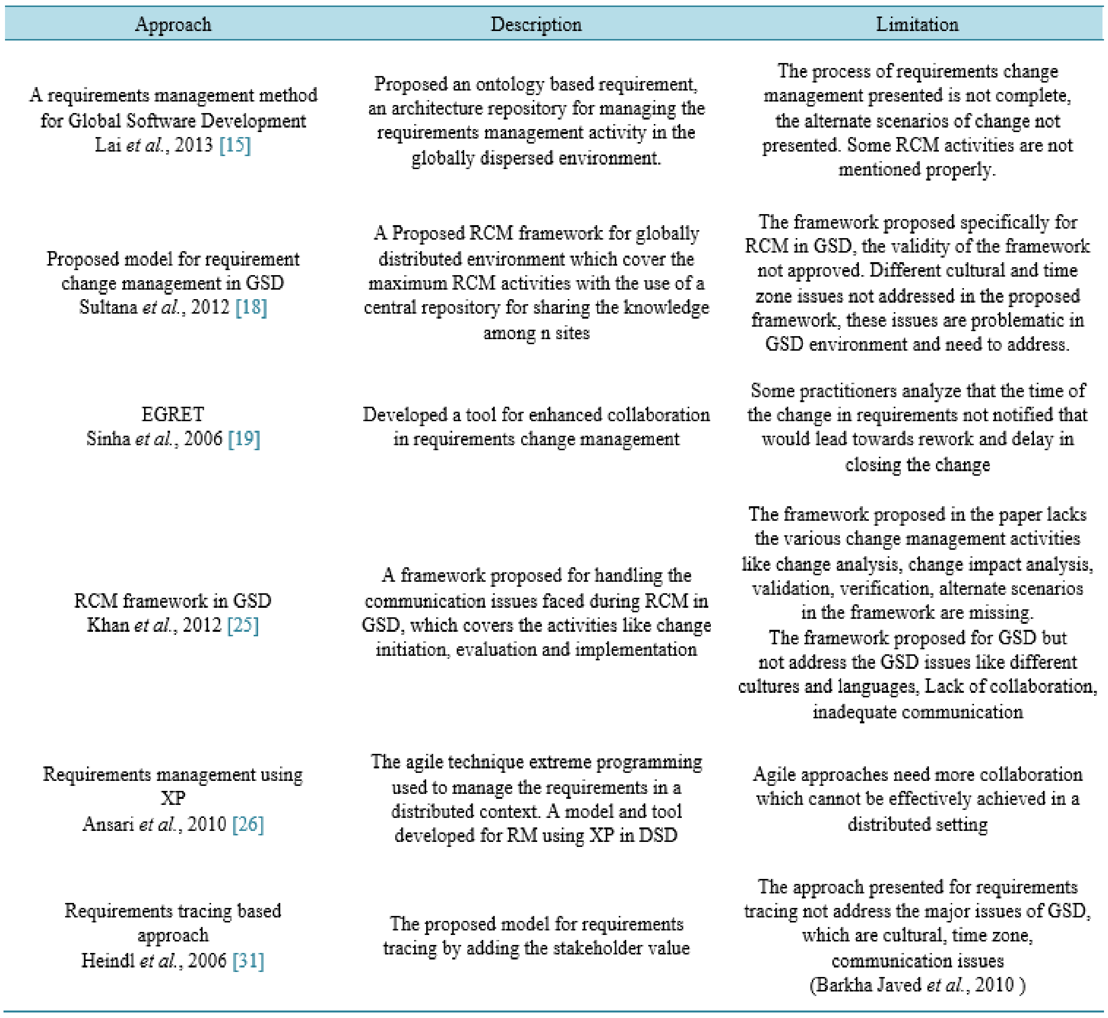
Apparent Benefits		Team Leader	Project Manager	Project Manager	Project Manager	Software Manager	Software Manager	Software Manager	RE Engineer	RE Engineer	QA Manager	QA Manager
Company without Using RCM Model	User satisfaction	P	P	O	A	A	P	A	O	N	O	A
	Ease of Learning	A	N	N	O	A	N	A	A	O	A	A
Company With Using RCM Model	User satisfaction	A	A	N	O	A	A	A	A	O	A	A
	Ease of Learning	A	A	O	A	N	A	O	A	A	A	A

Agreed= A, Partially Agreed= P, Not Present= N

Table 2 .1 : Benefits of using an RCM model

Same as this from some studies it is highlighted the success factors of the RCM model to each party that contributed to the software development process. a body of both researchers and practitioners that can help them to successfully execute the RCM activities. The identified SuccessFactors represent certain key areas that practitioners need to focus on to improve the implementation of the RCM process in the GSD environment. The ultimate objective of this study is to propose a model that could help the software development firms towards the successful [8]

The researcher of the article “An Improved Framework for Requirement” (2014) has mentioned an evaluation of Proposed Framework The RCM_GSD framework has been evaluated by two different methods. 1) Evaluation through industrial stakeholder feedback based on case study results 2) Evaluation through comparison with the existing literature-based frameworks.[6]



Approach	Description	Limitation
A requirements management method for Global Software Development Lai *et al.*, 2013 [15]	Proposed an ontology based requirement, an architecture repository for managing the requirements management activity in the globally dispersed environment.	The process of requirements change management presented is not complete, the alternate scenarios of change not presented. Some RCM activities are not mentioned properly.
Proposed model for requirement change management in GSD Sultana *et al.*, 2012 [18]	A Proposed RCM framework for globally distributed environment which cover the maximum RCM activities with the use of a central repository for sharing the knowledge among n sites	The framework proposed specifically for RCM in GSD, the validity of the framework not approved. Different cultural and time zone issues not addressed in the proposed framework, these issues are problematic in GSD environment and need to address.
EGRET Sinha *et al.*, 2006 [19]	Developed a tool for enhanced collaboration in requirements change management	Some practitioners analyze that the time of the change in requirements not notified that would lead towards rework and delay in closing the change
RCM framework in GSD Khan *et al.*, 2012 [25]	A framework proposed for handling the communication issues faced during RCM in GSD, which covers the activities like change initiation, evaluation and implementation	The framework proposed in the paper lacks the various change management activities like change analysis, change impact analysis, validation, verification, alternate scenarios in the framework are missing. The framework proposed for GSD but not address the GSD issues like different cultures and languages, Lack of collaboration, inadequate communication
Requirements management using XP Ansari *et al.*, 2010 [26]	The agile technique extreme programming used to manage the requirements in a distributed context. A model and tool developed for RM using XP in DSD	Agile approaches need more collaboration which cannot be effectively achieved in a distributed setting
Requirements tracing based approach Heindl *et al.*, 2006 [31]	The proposed model for requirements tracing by adding the stakeholder value	The approach presented for requirements tracing not address the major issues of GSD, which are cultural, time zone, communication issues (Barkha Javed *et al.*, 2010)

Table 2. 2: An evaluation of the Proposed RCM_GSD framework

When evaluating all the above conditions, situations and challenges of a researcher have introduced an approach for distributed agile development to manage requirements and their changes during the development processes. This approach works to fill the gap between the industry and research in distributed agile development by combining industrial practice and academic techniques. The suggested approach is based on a feature model using a feature tree. The feature model is the original introduced model of the commonality and variability in the domain engineering phase of software product line development A supporting tool is developed to help to manage software requirements changes. Its main objective is to mitigate some of the challenges facing distributed agile development. The supporting tool is tested and evaluated in real environments. [7]

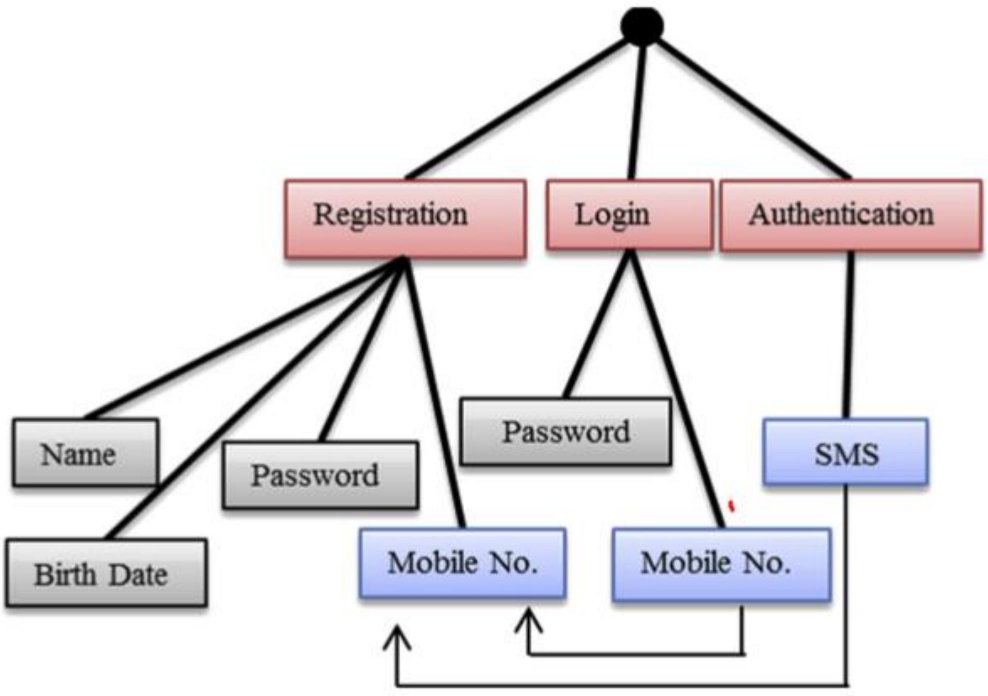


Figure 2. 2 : Feature tree model

The process of constructing the feature tree consists of two main steps. The first step aims to construct the initial form of the feature tree. The initial form of features tree will contain

all candidate requirements that shows an example of the initial feature tree. The second step is managing the requirements changes. After each iteration the development team will decide on approving the new changes or preventing it using the approval and prevention notations presented, The change management process can be summarized in three activities. a) Modification of requirements in the form of adding, deleting, and updating. b) Controlling of changes in the form of approving or prevent. c) Keep the tacking of the dependency relationship between requirements in the form of implication and exclusion. To introduce a complete distributed agile approach with requirement change management, the authors combined the suggested feature tree with The New Agile Process This is based on exploring different approaches which follow the agile development models and the agile manifesto [7]

So in this research, this study has analyzed above mention concepts and studies and proposed a new solution.

2.2 Limitations of earlier studies

limitations	References number
It just mentions a typical RCM process and it should manually create a change proposal note for impact analysis	[1]
Did not specifically identified the limitations of specified tools	[14]
This has specified the Jira and confluence as the best-interacted tool, but it does not give an exact solution for it.	[2]
Identified the reasons and impacts for the requirement changes but no suggestions to solve the impact of the changes	[13] [4]
conducted a review of software change impact analysis models, but not highlighted a common behavior	[3]
Suggested model and success factors depend on the knowledge management framework, but no logical pattern behavior	[5] [8]
Even there is tool support it won't get the proper impact analysis because some points are executed manually	[3]
Even the researcher has introduced a featured tree concept of the agile method the levels of tree is not enough to present the depth of the requirement	[7]

Table 2. 3 : limitations of earlier studies

2.3 Chapter Summary

This chapter has analyzed the existing literature in the area of requirements change management in global software development (GSD). It has highlighted the persistent challenges and also some limitations need to be resolved in this area. So this chapter pointed out the further understand and explore how requirements change management is performed with the agile concepts. The chapter has reviewed the literature on the available collaborative technologies, tools, and frameworks

CHAPTER THREE

RESEARCH DESIGN AND METHODOLOGY

In this chapter, it is presented how the research flows Then the selection of research methodology and the context of the research is provided. Next, the selected cases, research participants, and data collection methods are discussed in detail. Furthermore, details about the analysis techniques applied and the tool used in the research.

3.1 Research Question Selection

Step 01 - Analysis the researcher's professional background

As a professional Quality assurance engineer, the researcher has many experiences that the industrial people are facing in the software development process. When the project is ongoing the informal changes were often discussed only among the client and the management. Since it makes a problem of communicating change information to the testing team Most of the points that the people have thought the quality of the final product is dependent on the QA Engineer. But the final product quality is a result of the whole team. So the incompatibilities of the communication regarding informal changes will occur several other challenges form changes. Until the change requirements and managing, traceability information did not track while going for the verification. It occurs the confusion resulted between developers and testers. When the implemented changes in requirements were reported as bugs. This got a huge impact on a testing team during the testing cycle. They faced challenges in identifying the right people who have responded. So it is important that starting the project with a proper standard and continues that standard until the end of the product relies upon. For this, in the current situation, they have used some project management tools. But these tools are unable to track the depth of the required changes. So most are handled by the manual proses. So the researchers thought to identified the limitations with the current tools and introduced a solution

Step 02 – pointed out the problems

Issue 1 - Issues regarding reporting the client feedbacks in the concept of Agile Software Development and the Software as a service

Issue 2 -Issues regarding handling the old existing Requirements Vs New Reported Issues, And compare and contrast them

Issue 3 - Issues regarding defining parameters and standards to filter out Bugs and issues.

Issue 4 -Issues regarding getting clarifications within a global team that spread around different geo-locations.

Issue 5 - Issues regarding finding out impact analysis.

Issue 6 - Varying the time estimations due to the wrong impact analysis.

Issue 7 -Having some arguments and Conflictions within the team

Step 03 – Derived the research questions

- Research Question 1: *What is the most used project management tool used in the current software development process?*

- Research Question 2: *What are the limitations that you identified on the most used tool regarding the change requirement management?*

- Research Question 3: *What will be the best solution that handles that limitation as to the most profitable and efficient?*

3.2 Deriving the solutions

Step 04 – identified the solutions to derived problems by analyzing the proficiency level data gatherings

Data collection methods

- Depend on the previous own experiences of the researchers

- By intervening the some of the contacted professionals in various local and international software companies

Solutions from analysis

a. Research Question 1: What is the most used project management tool used in the current software development process?

- Solution 01; JIRA

b. Research Question 2: What are the limitations that you identified on the most used tool regarding the change requirement management?

- Solution 01; it has no introduced plugin to manage the requirements change management

- Solution 02; No simplified option to clients to report their issues through the tool, they can just report it as a ticket, And also no manageable database to client reported issues to filter out by comparing the existing requirement. So this can be solved out by automating this process

c. Research Question 3: What will be the best solution that handles that limitation as to the most profitable and efficient?

- Solution 01; introduced a new plugin to the JIRA tool to requirements change management

Step 05 – identified the basic structure of the JIRA plugins

Jira is following a component base hierarchical structure

i. Level 01 has identified as the project categories that tracked every single point that needs to identify. This has behaved like the root component of the whole product. No component identified without the project categories.
ii. Level 02 has identified the project sub-sections and the version milestones with the related projects
iii. Level 03 has managed the issue and the issue types on behalf of each project component and project
iv. Level 04 is the final level that manged with the sub-tasks that related to each main tasks related to a project

These components can map each other as they needed. It is maintaining the different levels of components such as component, project, sprint, sprints, issues, etc. All these components are mapped to a center point *PROJECT*

And then you will be able to get some summery report to this center point depend on different filters and there are specific users that you can create with different user permissions and roles and the admin can handle the access to each user role. And if you need more features such as adding test cases, the user will be able to buy some extra JIRA plugins by paying only to that feature. When a user buys a separate plugin it will be installed based on already added project sprints and Users. Jira has followed a very simple UI structure

(Existing UI in the current JIRA tool is provided in appendix A)

Step 06 – Analysis the way of the current requirement tracking process

When analyzing the current requirement tracking processes it is distributed collaboration over different manners. The JIRA gives the feature of CONFLUENCE This should be used as a separate plugin But Confluence can just upload the documents Or add the requirements descriptions as web pages. Since that handling the confluence in not that much easy. It is not efficiently readable. For distributed team members in both cases to clarify, report, and discussed change-related information make ambiguities. and enable the knowledge gap in requirements to change management. This will more be impacted by global software development challenges and enables some practical issues, for project success. This considered the main factors for the success or failure in a project to be the client and team relationship, leadership, and management of people. Some issues in distributed stakeholder collaboration hold the key to the impact of managing requirements

In the modern collaborative development environments the Confluence & JIRA support for code management, requirements management, and team collaboration. But these all are failed to automate the concept of filter out the bug and Change requirements and simplify the impact analysis process.

In Some cases, the clients did not adopt a project on Confflueneced process standards that created knowledge gaps between the client and team, especially for business analysts and developers. So some of the main clients found the interface complexity and understanding of the Confluence and JIRA to be very challenging. Since they did not use them for information sharing. So this created problems regards to change related knowledge sharing. So the researcher noticed it is important getting improved for the issue-reporting usability will influence the users to report the issues throughout a system and increased productivity.

And also the researcher has identified the problems while working with issue tracking software, They are unable to manage the timely occurrence issue information and identifying relevant information for a particular issue while manging in task allocation accurate reporting and getting approvals and difficulties in regaining historical record of a team’s while the process of development is managing with the time

In another point of view, Emails can consider another way of communicating changes The emails messages were also getting more difficult. This will be a point of getting more misunderstandings since getting some language issues and the contextual views are not available. People from different organizations and cultures will have different views and attitudes. So the emails can be identified as a method that gets very law support to resolve change requirement management issues

Web conferencing tools also be able to report the changes but having the trobles with trackings and centralizing the requirements. The connection problems, poor audio quality are some difficulties At some scenarios Spreadsheets can be used to manged the changes but will be having inconsistency of requirement information

(Existing UI in the current CONFLUENCE tool is provided in appendix B).

Step 07 – Limitations and the issues occurs with the current process

With the current process even you uploaded the requirement document, emails there is no method to easily search back the needed requirements So unable to get a requirement summery when there are changes that happened It should run a manual process when the team needs to get the impact analysis. Since the team members are moved it is very difficult to identify the impact analysis When the changes request from the client-side it occurs some language issues. Usually, there is some knowledge gap between the technical people and the business people. So it is important that having a common language that will be understood to any of them. And also it will occur some conflicts between team and clients And the clients are directly unable to report the issue through the project management tool

Change management practices have faced may limitations during the current situation. The new requirements as well as changes in the existing ones continued the changes until the end of the project. Since the change management process in some cases invoked high impact changes as well as smaller and low impact changes this will be taken an important role in the outcome of the project. So the limitations of the process will result in an overwhelming number of unmanageable issues that highly impacted the cost and schedule of the whole project. The initial project plan will constantly be updated which resulted in delays and dragged the planned release dates since additional project cost and additional time required to finish the project. Therefore the flow of requirements changes management should strongly manageable for the success of the project.

The globally distributed projects are facing big trouble when deciding which requirements will and will not move into the next software releases since they are unable to get a proper impact analysis requirements. Some of these changes may be positively advantaged But some may get disadvantage when it appeared as a bug

Late changes in requirements caused a heavy impact on cost and schedule. At the same time, it impacted the stability of the existing system and make frustration among the clients and team and the company

The scope changes will cost for resource management, however, the development team has to fight for work exhaustive extra hours and kept the project on course for on-time delivery. Some changes will be effective as immediate fixes. So it is important having an easy time-saving method of handling all the required changes and identified how much will it impact the whole project.

Because wast of time and resources will be a loss for the important project for a software development company

Step 08 – Then identified the user groups involved and their functionalities

This research work should target on major 2 groups involving the Software Agile based development process

<u>External Clients</u>

The responsible client representative from the client-side. That person will be given a Knowledge transfer from the team at the 1st stage

<u>The internal Software development team</u>

Project Manager, the main responsible person for all the project task

Business Analyst the responsible person for requirement handling and communicate to customers

Developer responsible person to develop the product and give an estimation to impact of the changes

Quality Assurance Engineer the responsible person to validate and verify the requirements has implemented as expected

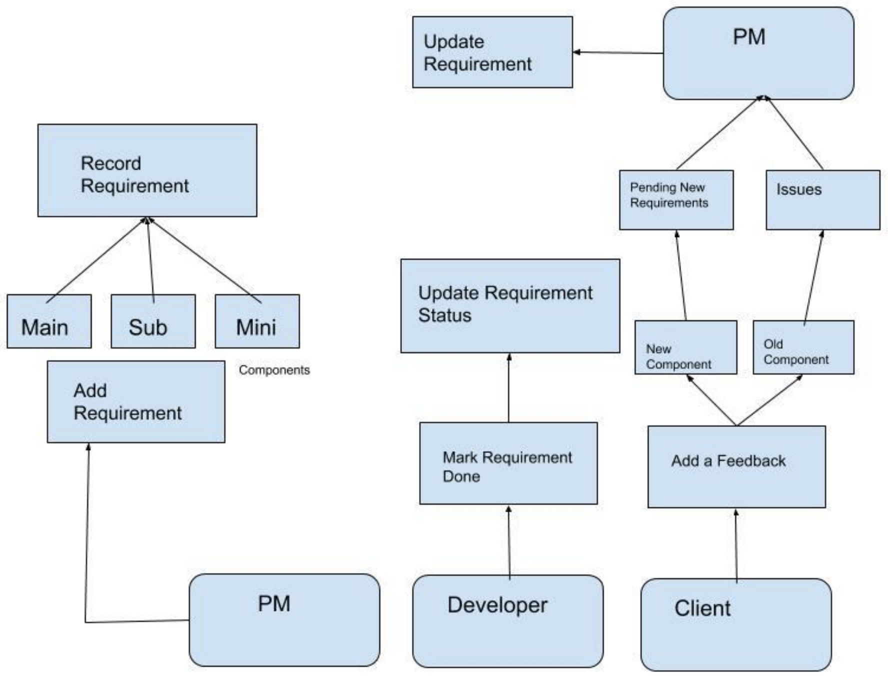


Figure 3 . 1 : User-based functionality

i. First, the BA created some component-based requirement DB
ii. At the end of each sprint, they realized working product to the customer
iii. And through a system UI client reported the issues
iv. And then mapped pre-requirements with the user reported issue
v. Filtered out the Changes and Bugs through the system
vi. Then BA assigned it to team
vii. The team gives the estimations
viii. At the end DB updated.

Step 09 – Suggested solutions through the all analysis

1 – Preconditions to system usage

1. This should be working as a plugin to JIRA.
2. There should be JIRA accounts that already created with the users
3. Same JIRA user groups are expected to link
4. When you sign the project agreement, you should provide training and a user manual to client-side by describing how to manage then issue reporting
5. But the client does not need knowledge about change requirements and bugs. Clients just reported the issue they found
6. The team should manage the tool as a usual project management tool
7. They do not need new knowledge
8. This too is behaved as a usual web-based tool

2 - System processors context

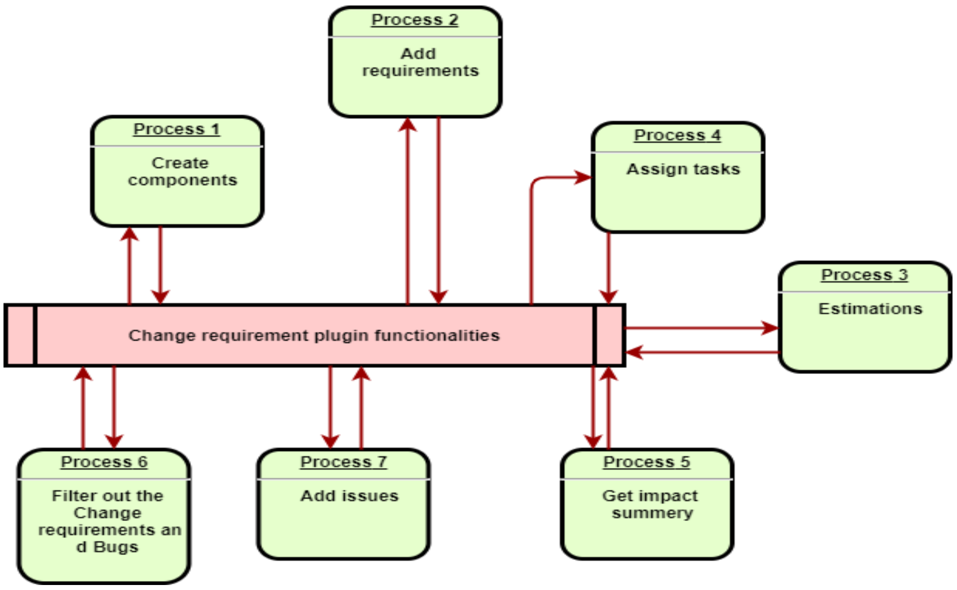


Figure 3. 2 : functionality Context

The solution concept is mainly contained three modules, for Filter out the bug and Change requirement, Find out the impact analysis, and task assignment and status management. Within these three modules, the system contains seven processes,

Process 01 is to create components, this is functioning while the business analyst derived the requirements individually, While the Business Analyst received all project requirements he has manually handled these requirements and documents them, according to this solution the tasks are filtered while it's identified the relevant component. These identified components should be added to the system. Then Process 02 is to add the requirement in here the requirements are added by mapping to the created components, At the same as the process 07 will get the inputs for issues with the mapped components Then the process 06 filters out the bug and change requirement will be executed and find out what is the bug and what is the change request. These processes are related to module 01. Then again the process 05 will be executed for module 02 and get the impact summary for each reported issue. Finally, process 04 will assign the tasks and process 03 will get the inputs for the estimations on module 03

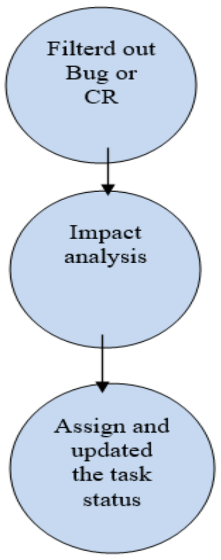


Figure 3. 3 : Modules of the Solution

3 - Problem vs identified Solution

The one of a problem with the currently existing solutions is the complexity and the difficulty in understanding for the clients, So in the proposed solution, designed more simplified user interfaces that anybody can easily understand what they have to do and how it should be operated. So this reduces the number of click events that users should follow to one task. This is designed to be more efficient and usable

And in the change requirement management process the most important point is managed the pre-requirements. So currently this struggle with handling hard copies, soft copies, emails, chats, conference recordings, spreadsheets, etc. All these communication mediums are unable to get on a centralized point. And in some situations, these may be misplaced with the resource movements. If the responsible people are moved out some of the knowledge transfers will not happen properly. So with the proposed system, all the issues are reported with the system UI, and all the pre-requirements are stored on a database table with the details of the responsible people. The soft copies or hard copies of the documents may be more difficult to access back and it takes more time to read a document. And the email attachments will be lost due to the technical issues. There will be some accessible issue occurs with chats and the conferences. So all these issues are solved with the proper standard method with the solution. It introduces easily access UI and the report summaries. And automatically centralized all the project requirement in a database, it gives easy access methods to view each requirement and the issues

And the most important point is the method of defining the bug and change requirements. Currently, this has happened manually. So this in scenarios the solution is used as a component structure to each requirement and the issues and flowed a hierarchical structure to mapped each requirement and the issue. Each requirement and the issues are separated by a title that attached to the component hierarchy. By comparison of a component mapping logic, this will automatically define what is a bug and what will be the change request

Then the next thing is to find out the impact analysis, without having proper knowledge of the pre-requirements and the changes the new resources will be failed to track the proper requirement analysis. Then this will occurs new issues and even the system will be able to prompt a lot of incorrect results and the outputs. And also there may be lots of possibilities in the system crashes. So this will be impacted by the development team's image. Not only that this will be more costly and make the losses to the development company. By this component hierarchy, it will take the opportunity to identify the impact on each component level. That means this will get the impact from each small point, So the solution will pave the way to track each change

Finally, it will be some misunderstandings with the responsible people when the issues occur. So the solution will be able to keep the tracked for reported issue up to fixing and closed the issue, So this is easily identified by the responsible people. With all the above solutions finally, this system solution will make an automated standard method to filter out Bug and change requirements to replace the manual process

4– Core component hierarchical structure

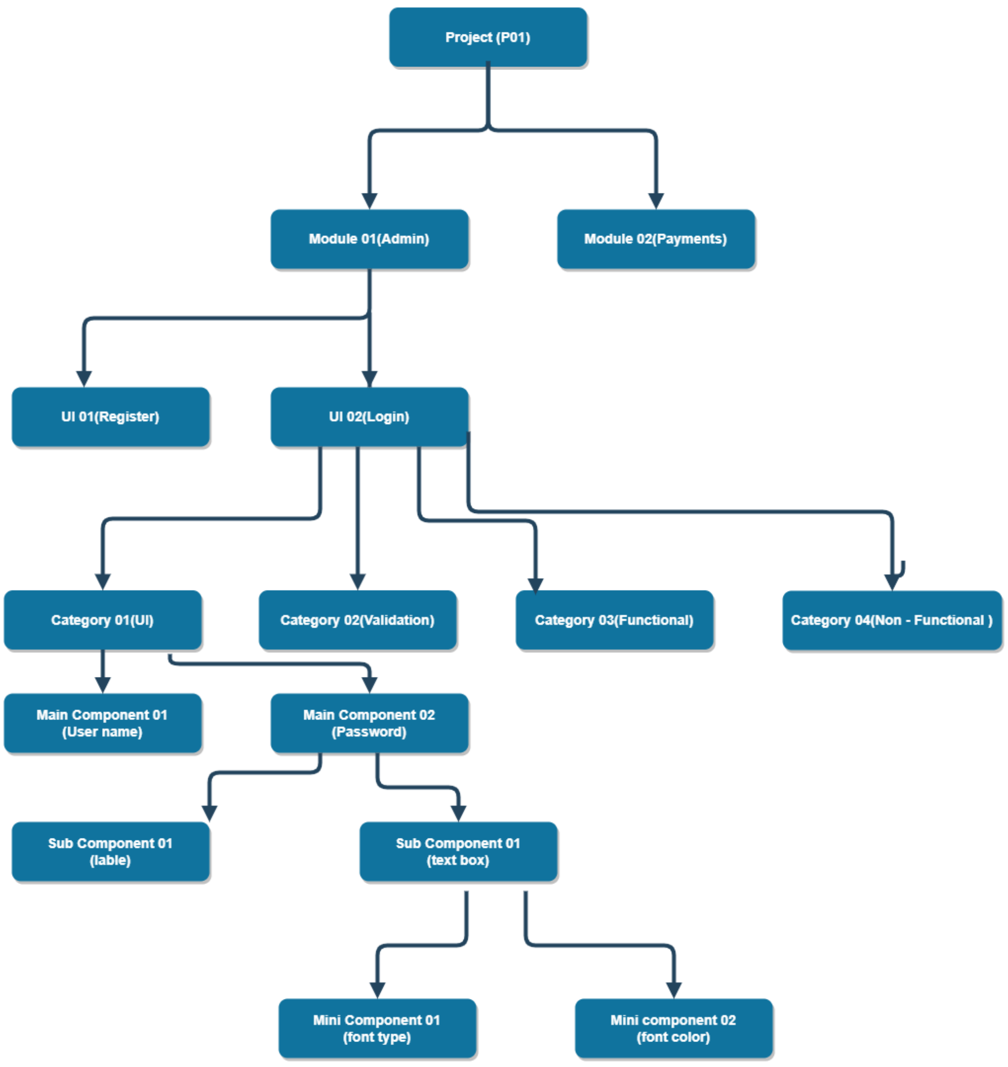

Project (P01)
Module 01(Admin)
Module 02(Payments)
UI 01(Register)
UI 02(Login)
Category 01(UI)
Category 02(Validation)
Category 03(Functional)
Category 04(Non - Functional)
Main Component 01 (User name)
Main Component 02 (Password)
Sub Component 01 (lable)
Sub Component 01 (text box)
Mini Component 01 (font type)
Mini component 02 (font color)

Figure 3 . 4 : Core Component hierarchical structure

i. level 01 - Root Component - Project eg: Student management

ii. level 02 - Module - eg: Student Registration

iii. level 03 - UI (Interface)- eg: login page

iv. level 04 - Requirement Category - eg; UI, Validations, Functional, Non-Functional (these 4 types are not changed, I only used these 4 categories when getting feedback from client-side)

v. level 05 - Main component -eg; User name

vi. level 06 - Sub Component- eg; Lable,Text Box

vii. level 07 - Mini Component - eg; Font color, Font face

3.3 Chapter Summary

This chapter has described the basics of research. Why the researcher selects this problem. What is the background, How data are gathered, What are the conditions based on this study and the basic context expected to use?

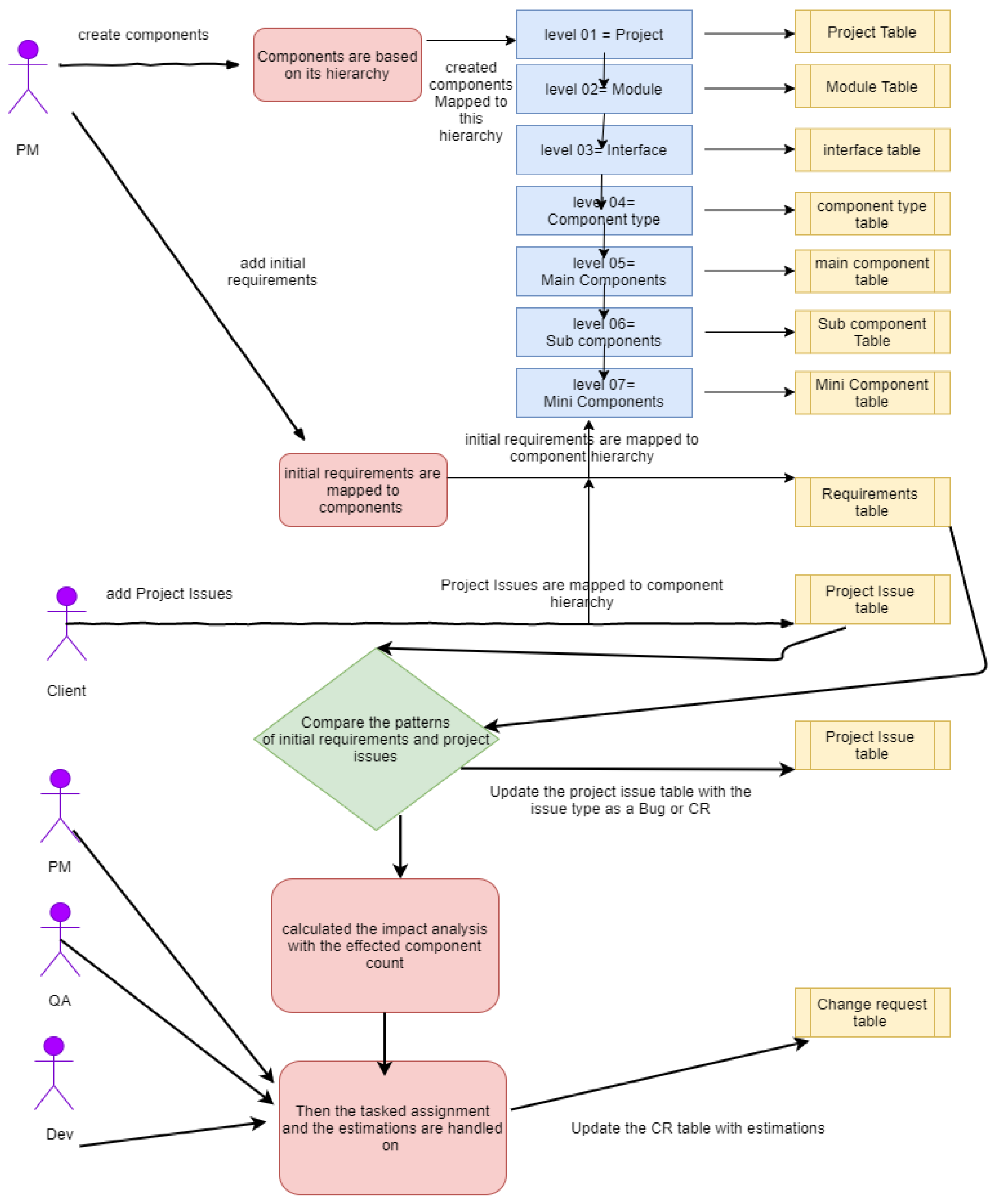
In here researcher highlighted the features of the pre analyzing tool and its structure and how the current solution will depend on that basic structure

CHAPTER FOUR

ANALYSIS AND FINDINGS

The purpose of this chapter is to present the results and findings based on the design and methodologies if chapter 3

4.1 Logical design of the solution



create components
PM
Components are based on its hierarchy
created components Mapped to this hierarchy
level 01 = Project
level 02= Module
level 03= Interface
level 04= Component type
level 05= Main Components
level 06= Sub components
level 07= Mini Components
Project Table
Module Table
interface table
component type table
main component table
Sub component Table
Mini Component table
add initial requirements
initial requirements are mapped to component hierarchy
initial requirements are mapped to components
Requirements table
add Project Issues
Client
Project Issues are mapped to component hierarchy
Project Issue table
Compare the patterns of initial requirements and project issues
Project Issue table
Update the project issue table with the issue type as a Bug or CR
PM
QA
Dev
calculated the impact analysis with the effected component count
Then the tasked assignment and the estimations are handled on
Change request table
Update the CR table with estimations

Figure 4.1 : Logical design of the solution

4.2 Logical Functionalities

This is mainly processed with specifically identified component structure. When the process starts with the development team side, the Project manager and the Business analyst will be identified as the component structure after analyzing the initial requirements.

The first component is filtered out on the project, As an example, it will be a Student management project to a university. So the level 01 component identified as Student Management. Then should filter out the module wise. Within the example, the module will be Registration, Examination, Attendance, etc. For this hierarchy, we should move forward across the hierarchy. In any case, there should not be any of the isolated components that not mapped to each other according to the defined hierarchy. As an example, there won't any modules without a project. Then moving to level 3 it should be the interface (UI) level. For a software solution, there is a collection of interfaces that mapped to different modules.in this example, for module registration, there will be some interfaces like enter parents' details, enter contact details, enter general details, etc. When considering the interfaces we are discussing the different requirements on the selected interfaces such as functional, Non – functional, UI, validations. So as the 4th level it's defined 4 master data components as a)Functional b)Non-Functional c)Validations d)UI. The entry requirement that reported will belong to any of this category. For the functional requirements, it belongs to the functionalities like getting a total of marks. And for non-functionality, it can describe the loading time, For under UI it describes the colors, fonts, alignments, etc. For validations, it will be such things as email validation. Then the 5th level is defined as the main component When it described under the UI component type we will take the mail component as user name, under that the 6th level subcomponent becomes user name label. Then the last level 7th level becomes a mini component label color. Like this, each requirement that reported should be mapped under the component structure.

4.3 The method of the solution logic

When the requirements are identified properly the components should be created to the requirements. That each component is mapped to a component hierarchy according to requirement need to add. In the same manner, the user reported issues also mapped to the component hierarchy and the logic is executed and it will automatically define the what is Change request and what is Bug.

For example, let us think about some requirements for a student management system, The clients give one of a pre-requirement as below example.

Requirement 01 - In the student management project, the module registration, register page, UI, student name, label, font type, should be Arial

Then this requirement will be derived for the component levels as below

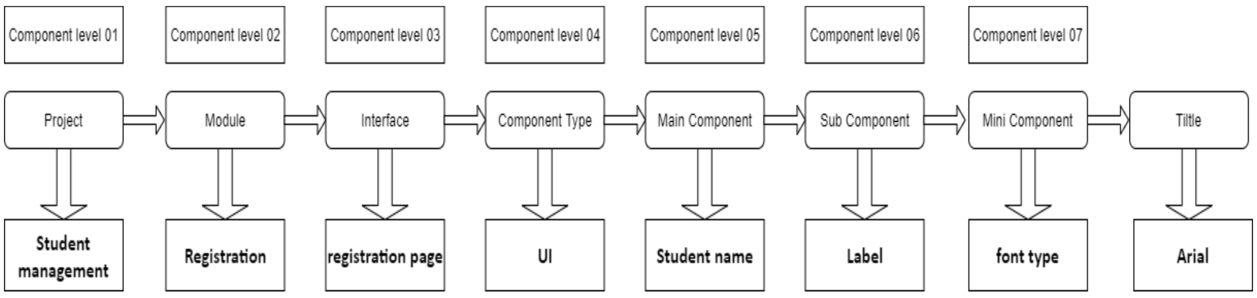


Figure 4 - 2 : How to derived requirement into component

At the end of the 1st sprint, the client will receive a working pice of product for the pre-requirement. Then the client represents to will reported below 2 issues through the system.

Issue 01

In the student management project, the module registration, register page, UI, student name, label, font type, should be Arial.

Issue 02

In student management project, the module registration, register page, UI, student name, label color should be yellow

Then that issue also derived for the component as below,

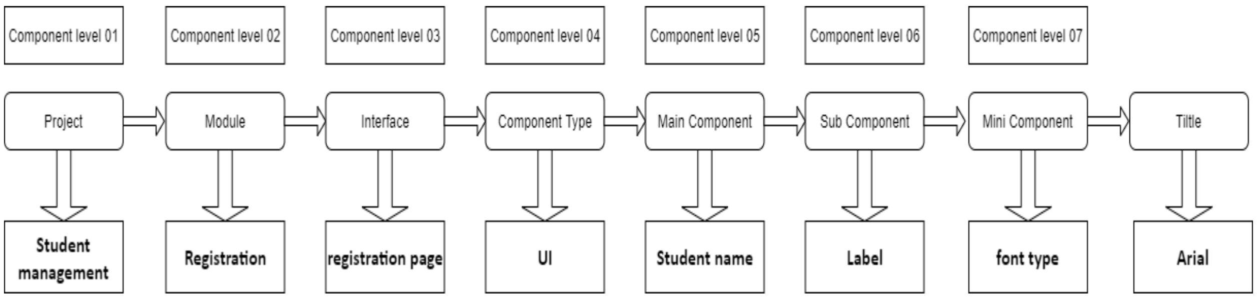


Figure 4 – 3 : Derived issue 01 for component levels

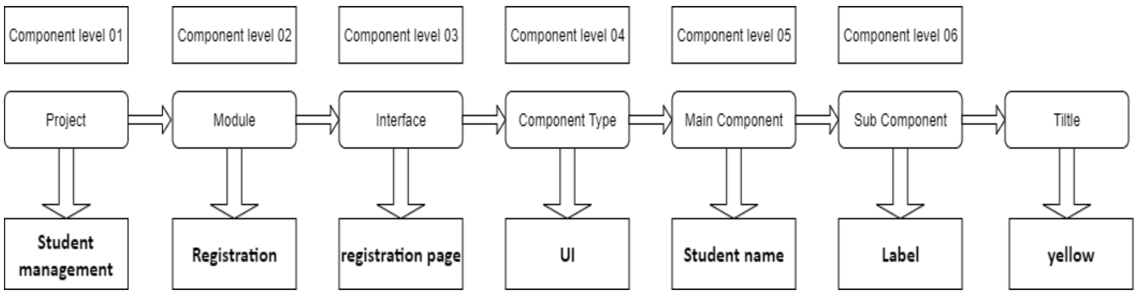


Figure 4 – 4 : Derived issue 02 for component levels

According to the above tow diagrams on the reported issue, the system will execute the logical function for define bug and change requests. As these two diagrams displayed the issue 01 and issue 02 has different component patterns with a different title, each reported issue will have it's own component pattern + title, By comparing these issues with the pre-requirements. If the same component pattern mapped with the same title. Then it will be defined as a Bug since it's already there previously. If there is no title mapped with the component structure for the issue then it will be a change request

Then the solution will be able to get the summary report on clearly identified Bugs and change request and the impacts for the issue will define from this, later this will assign for the team and manage the allocations, All the logics well executed according to the below Pseudo Codes

Pseudocode – Validate User

```
Initialized the  process validate user
          User  logging
          Validate the user & get the user role
IF user = Business Analyst
  Allowed user to Create components, Add requirements, Issue allocations, View
     the impact summery
IF user = Project Manager
  Allowed user to Issue allocations, View the impact summary, Change the bug to
     a Change Request or Change Request to a Bug
IF user = Developer
Allowed user to Issue allocations, View the impact summary, Add estimations
IF user = QA Engineer
Allowed user to Issue allocations, View the impact summary, Add estimations
IF user = Client
  Allowed user to create issues, View summary
End validate user process
```

Pseudocode – Create Component

```
Initialized the  process Create Components
Create a Project
  Get the component name and description for root-level = PROJECT
          Validate whether the component is existing on the project table
          If not exist create a new project
End Create Project
Create a Module
  Get the component name and description for root-level = PROJECT
          Validate whether the component is existing on the project table
```

```
                    Select the Project
            Get the component name and description for level 02 = MODULE
            Validate whether the component is existing on the Module table
            If not exist create a new module
            End Create Module
   Create a Interface
   Get the component name and description for root-level = PROJECT
            Validate whether the component is existing on the project table
                    Select the Project
            Get the component name and description for level 02 = MODULE
            Validate whether the component is existing on the Module table
                    Select the module
            Get the component name and description for level 03 = INTERFACE
            Validate whether the component is existing on the Interface table
            If not exist create a new interface
            End Create Interface
            Select Component Type = level 04
            From Componet type master data table
   Create a Main Component
   Get the component name and description for root-level = PROJECT
            Validate whether the component is existing on the project table
                    Select the Project
            Get the component name and description for level 02 = MODULE
            Validate whether the component is existing on the Module table
                    Select the module
            Get the component name and description for level 03 = INTERFACE
            Validate whether the component is existing on the Interfaced table
            Select the interface
                    Select Component Type = level 04
                    From Component type master data table
   Get the Main component name and description for level 05 = MAIN
       COMPONENT
            Validate whether the component is existing on the Interface table
            If not exist create a new main componet
            End Create Main Component
Create a Sub Component
   Get the component name and description for root-level = PROJECT
            Validate whether the component is existing on the project table
                    Select the Project
            Get the component name and description for level 02 = MODULE
```

```
    Validate whether the component is existing on the Module table
            Select the module
    Get the component name and description for level 03 = INTERFACE
    Validate whether the component is existing on the Interfaced table
    Select the interface
            Select Component Type = level 04
            From Component type master data table
    Get the Main component name and description for level 05 = MAIN
  COMPONENT
    Validate whether the component is existing on the main componet table
    Selecet main Component
    Get the Sub component name and description for level 06 = SUB
  COMPONENT
    Validate whether the component is existing on the  table
    If not exist create a new sub componet
    End Create Sub Component
```

Pseudocode – How to define Bug or Change Request

```
Initialized the  process Defined Bug or Change Request
Get the Pre requirments from Requirment table
Load into a list
Get the existing issues on the Issue table
Load to the list
Get the new added issue
While trying to save it to issue table
Execute
Get the title tag to the issue
    Validate whether it is = any of the titles from load list
    If NO
    Save to issue table by validating the components
    SET issue type = Change Request
    IF YES
        Get the component name and description for root-level =
        PROJECT
                Get the component name and description for level
  02 =
                MODULE
        Get the component name and description for level 03 =
        INTERFACE
```

```
                  Select Component Type = level 04
                        Get the Main component name and description for level
      05 =
                        MAIN COMPONENT
                  Get the Sub component name and description for level 06 =
                  SUB COMPONENT
                  Get the Subcomponent name and description for level 07 =
                  MINI COMPONENT
                  Check while the level of tile matched with a new title
   When found
      SET issue type = BUG
                  End Defined Bug or Change Request
```

Pseudocode – How to get Impact analysis

```
Initialized the  process Impact Analysis
Select the issue code
Load the number of components tagged to the issue code
Load them to a list
Get the count of the list
Print the list from the Root component to ascending order
End the  process Impact Analysis
```

Solution logic diagram

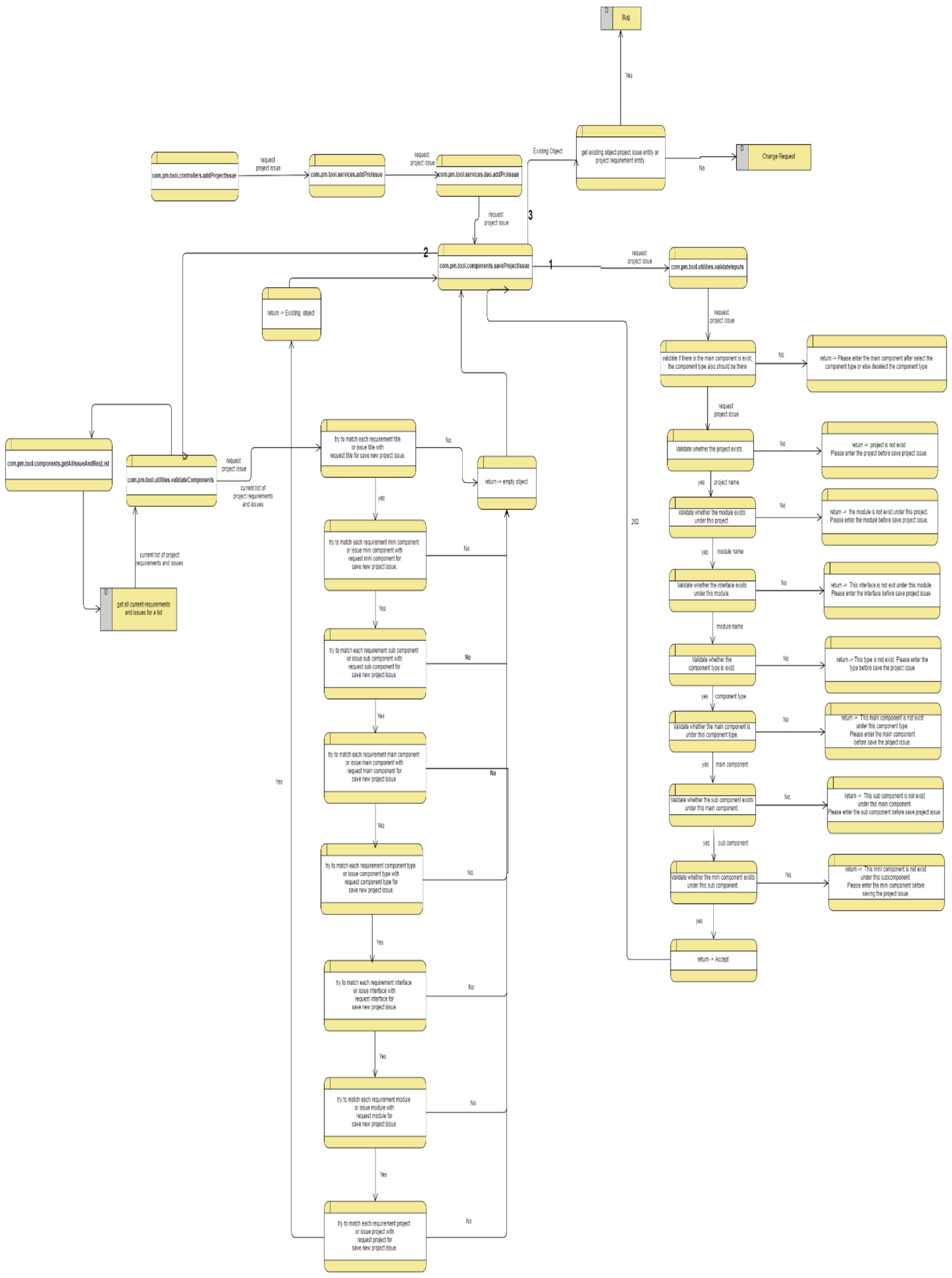

Bug
Yes
com.pm.tool.controllers.addProjectIssue
request project issue
com.pm.tool.services.addProIssue
request project issue
com.pm.tool.services.dao.addProIssue
Existing Object
get existing object project issue entity or project requirement entity
No
Change Request
request project issue
3
2
com.pm.tool.components.saveProjectIssue
1
request project issue
com.pm.tool.utilities.validateInputs
return -> Existing object
request project issue
validate if there is the main component is exist, the component type also should be there
No
return -> Please enter the main component after select the component type or else deselect the component type
request project issue
Validate whether the project exists.
No
return -> project is not exist. Please enter the project before save project issue.
yes
project name
Validate whether the module exists under this project.
No
return -> the module is not exist under this project. Please enter the module before save project issue.
202
yes
module name
Validate whether the interface exists under this module.
No
return -> This interface is not exit under this module. Please enter the interface before save project issue.
module name
Validate whether the component type is exist.
No
return -> This type is not exist. Please enter the type before save the project issue
yes
component type
Validate whether the main component is under this component type.
No
return -> This main component is not exist under this component type. Please enter the main component before save the project issue.
yes
main component
Validate whether the sub component exists under this main component.
No
return -> This sub component is not exist under this main component. Please enter the sub component before save project issue.
yes
sub component
Validate whether the mini component exists under this sub component.
No
return -> This mini component is not exist under this subcomponent. Please enter the mini component before saving the project issue.
yes
return -> Accept
com.pm.tool.components.getAllIssueAndReqList
com.pm.tool.utilities.validateComponents
request project issue
current list of project requirements and issues.
current list of project requirements and issues.
get all current requirements and issues for a list
try to match each requirement title or issue title with request title for save new project issue.
No
return -> empty object
yes
try to match each requirement mini component or issue mini component with request mini component for save new project issue.
No
Yes
try to match each requirement sub component or issue sub component with request sub component for save new project issue.
No
Yes
try to match each requirement main component or issue main component with request main component for save new project issue.
No
Yes
Yes
try to match each requirement component type or issue component type with request component type for save new project issue.
No
Yes
try to match each requirement interface or issue interface with request interface for save new project issue.
No
Yes
try to match each requirement module or issue module with request module for save new project issue.
No
Yes
try to match each requirement project or issue project with request project for save new project issue.
No

Figure 4 – 5 : Solution logic diagram

4.4 Analyzing the sample data for test the system

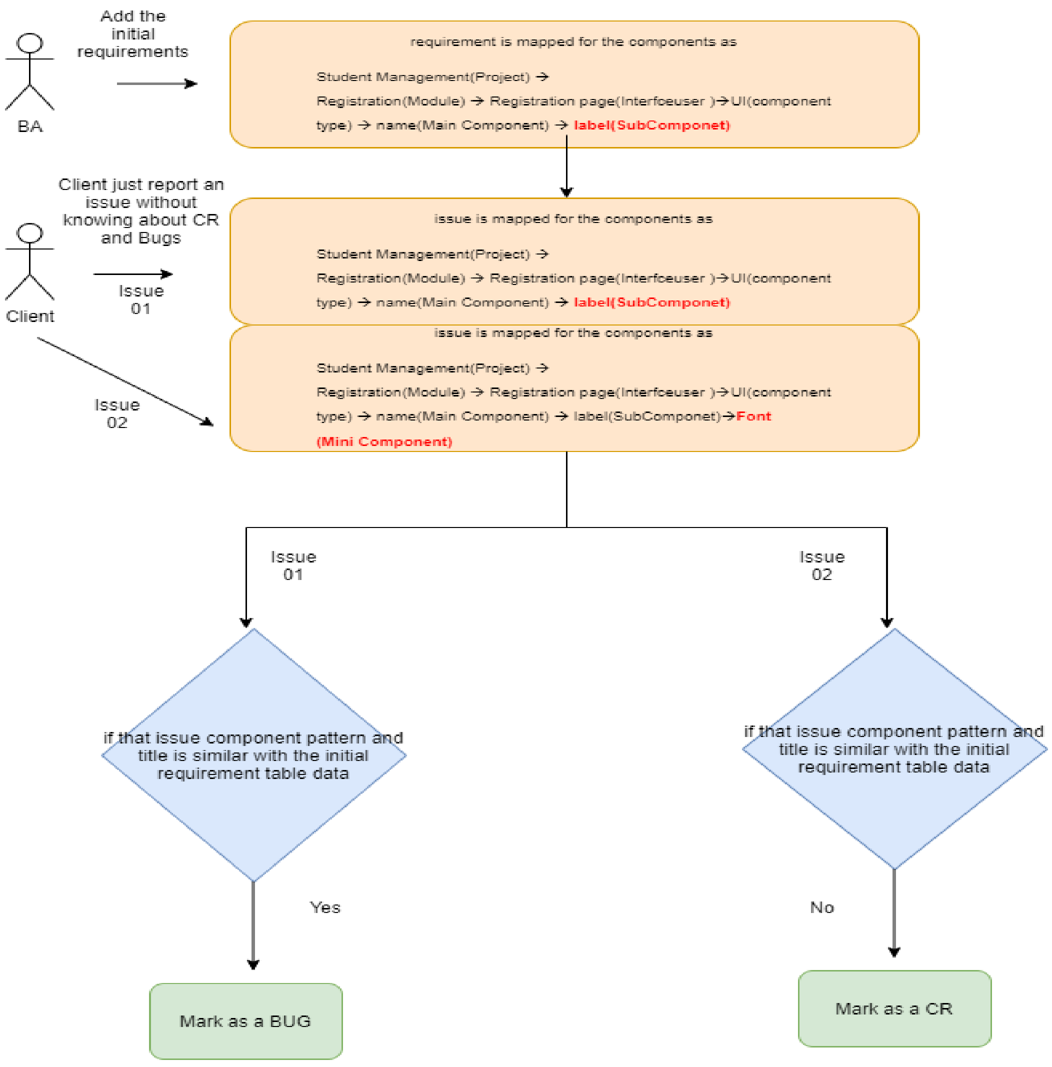
The method of filtering the Bug or CR is tested with some of the sample data that related to the actual project scenarios. The below table is included in the sample set of data.

project	Module	Interface	Component type	main components	subcomponent	Mini component
Student management	Registration	register page	UI	Student name	Lable	font type
Student mangemnet	Registration	regiter page	validation	Student name	Text type	Text
Student management	parent detail	parent detail page	UI	parent name	label	font type
Keels POS	create a bill	add product page	UI	product name	label	font type
Keels POS	create a bill	add product page	UI	product price	label	font type
Keels POS	generate bill	billing page	UI	generate button	button color	
Delivery mangemnet	Oder mangement page	order page	UI	Add button		
Delivery management	delivery approval	approval page	UI	select button		

Table 4-1: Sample set of test data

The above data are passing throughout the API calls and test the responses related to each API, There are some API to create the structures of the above components. The next API is added to the requirement and the other one is added to the issue. Then the system checked whether it is a Bug or CR. Scenario 01- User said I need the student management project registration module user name as a label.Student Management(Project) → Registration(Module) → Registration page (Interfceuser)→UI(component type) →

name(Main Component) → label(SubComponet) Here just mentions up to sub-component level.Then user reported an issue as, with the same pattern.Student Management(Project) → Registration(Module) → Registration page (Interfceuser)→UI(component type) → name(Main Component) → label(SubComponet).So this will become a **Bug**.Then again he said that he needs to change the **font** of the label.Student Management(Project) → Registration(Module) → Registration page (Interfceuser)→UI(component type) → name(Main Component) → label(SubComponet)→Font (Mini Component).Then the logic has will execute with the comparison 1st it checked the title and then refer through the hierarchy, This will be CR since it is moved to another component level



Add the initial requirements
BA
requirement is mapped for the components as
Student Management(Project) → Registration(Module) → Registration page(Interfceuser)→UI(component type) → name(Main Component) → label(SubComponet)
Client just report an issue without knowing about CR and Bugs
Client
Issue 01
issue is mapped for the components as
Student Management(Project) → Registration(Module) → Registration page(Interfceuser)→UI(component type) → name(Main Component) → label(SubComponet)
Issue 02
issue is mapped for the components as
Student Management(Project) → Registration(Module) → Registration page(Interfceuser)→UI(component type) → name(Main Component) → label(SubComponet)→Font (Mini Component)
Issue 01
Issue 02
if that issue component pattern and title is similar with the initial requirement table data
if that issue component pattern and title is similar with the initial requirement table data
Yes
No
Mark as a BUG
Mark as a CR

Figure 4 – 6: How to define Bug and CR

4.5 Prototype the Solution

When prototyping the model. The model requirements are defined in detail as expected. This involves interviewing several industrial people, analyzing the existing features of the current JIRA and confluence, and researchers' experience on the problem. With all these a preliminary, simple design is created for the new plugin. The prototype of the expected model is constructed from the preliminary design. This is usually a scaled-down model and represents an approximation of the characteristics of the final product. This model thoroughly evaluates and strengths and weaknesses, what needs to be added, and what should be removed. The prototype is modified, based on the limitations of the researcher's technical knowledge and the differences in the eligibility of the analyzed system and design. The preceding steps are iterated as many times as necessary until the best model is finding out for final product desired

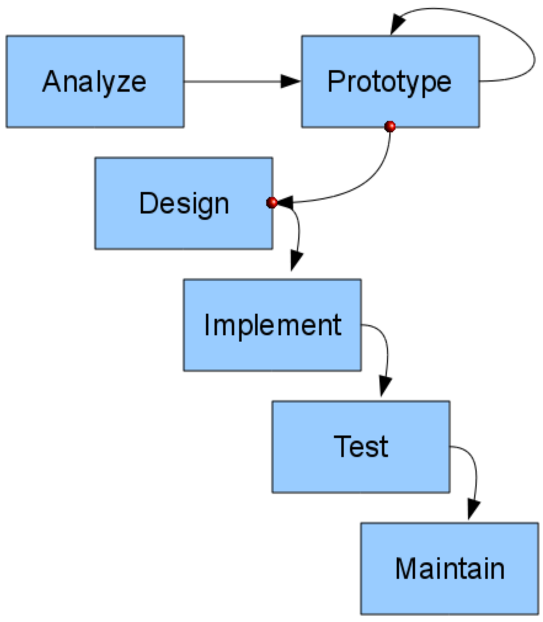


Figure 4-7 : Prototyping development mode

4.5.1 Used type of prototype

For this, the researcher has used the Rapid throwaway prototyping method. This method involves exploring ideas by quickly developing a prototype based on preliminary requirements that are then revised through on ging issues. The rapid throwaway refers to since each prototype is completely discarded and may not be a part of the final product. Same as in this project the researcher is starting the project from the prototype and completely discarded it when it developed. Rapid Prototyping applies an iterative approach to the design stage of a solution. The objective is to quickly improve the design using regularly updated, multiple short cycles. This saves time. It is used to build a physical prototype to demonstrate.

Tool sued for prototyping

https://www.fluidui.com/editor/live/

4.5.2 Considered software development concepts

When preparing the design for the solution the researcher has considered some software development concepts such as Laws of simplicity and the concepts of usability. A well-designed user interface and solution is comprehensible and controllable, helping users to complete their work successfully and efficiently, and to feel satisfied. Usability Evaluation focuses on how well users use a product to achieve their target. And also about effectiveness, efficiency, and the overall satisfaction of the user. So this research has considered about below points Because this is a system that involves both technical and no technical people. And this is used to report the feedback. So this should be understandable and clear and easy to learn

- Usefulness

  The system should provide information and functions provided to the user.

- Consistency

  Follow appropriate standards for the relevant domain.

- Real-world conventions:

  Use commonly understood concepts, the information in a natural and logical order.

- Simplicity

  Eliminate unnecessary elements.

- Visibility:

  most commonly used options should easily accessible

- Communication

  Should gives proper ides to the user so that he sees the results of his actions and what is going on with the system.

- Structure:

  Organized functions meaningfully. Put related things together.

- Efficiency

  Accommodate a user’s continuous advancement in knowledge and skill.

- Workload Reduction

  Make work easier, simpler, faster, or more fun. Automate unwanted workload.

4.6 Chapter Summary

In summary, this is explained the background and presented a contextual analysis of the case studied. And considerations of practiced models to resolve change management issues in the selected method. And explained the logical design, logical functionality with the conceptual view. And this described the user wise functionalities and how will be the solution supports to each user. And what should be the instructions to the client when using this solution. And how the client should get the basic idea about the system. Finally, it describes the Software development life cycle that the researcher has used and what methods used in the prototype

CHAPTER FIVE
IMPLEMENTATION

In this chapter the of the details about implementations and the tools and technologies used on the solutions

5.1 Implementation design architecture

This implementation is depended on the three-layered architecture Since this solution is suggested as a plugin to the existing solution the researcher has decided to maintain 3 tired architecture due to the three-layered architecture make the changes easy

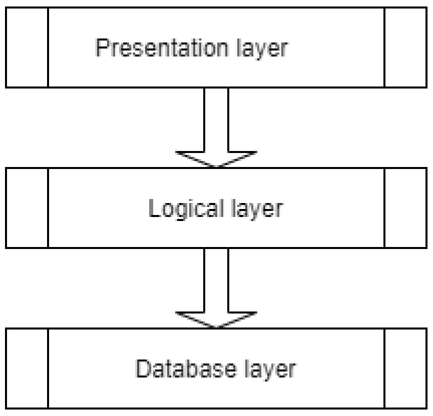


Figure 5- 1: Three-layered architecture

The Three-layer architectures look very beneficial for the development environment. So It is about modularizing the UI, business logic, and data storage layers. Such as

- Presentation layer
- Service layer
- Data Access layer
-

5.1.1 Presentation layer

The presentation tier represents the front-end layer of the user interface. This is graphical and accessible via a browser or a web-based application. So this section is mainly handled the user interaction to the system

This layer is often built on web technologies such as HTML5, JavaScript, CSS. Ajax Or used other popular web development frameworks and communicates with others layers through API calls.

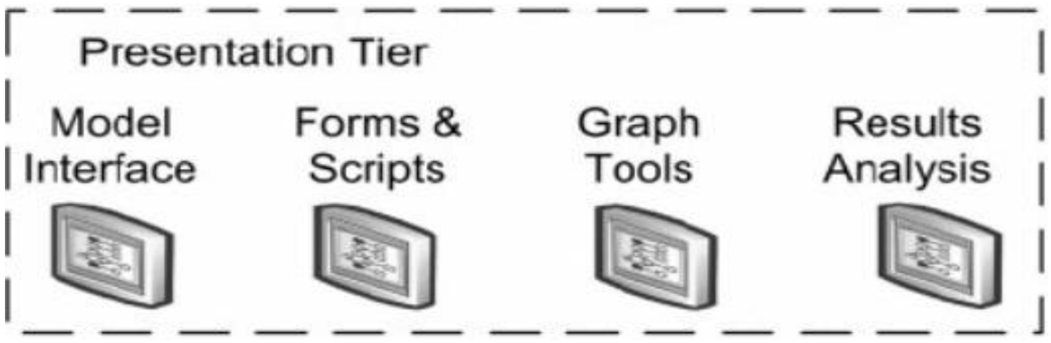


Figure 5 – 2: Presentation tier

5.1.2 Service layer

The service layer contains the function to control the application functionality by containing a set of rules for processing information and business logic. So this is separated the presentation layer and data layer by protecting confidentiality. Working on JSP, Java, Spring boot framework, and other programs. The logic tier would be run on a Web serve. It processes the inputs and interacts with the database.

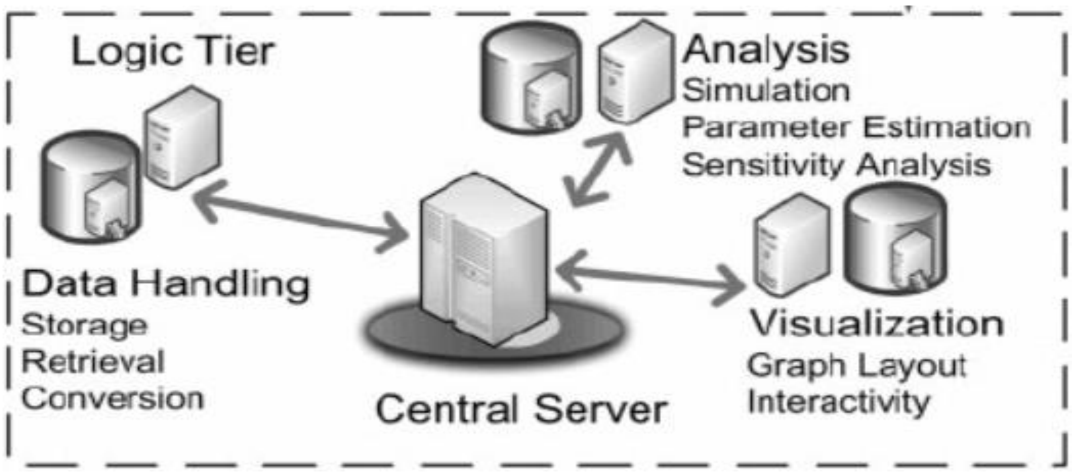


Figure 5- 3: Service tier

5.1.3 Database layer

The database layer saved the data which is inserted by the user. So all information that is entered will be saved in this layer. This provides access to the back end and data tier would be some sort of databases, such as MySQL and the framework technologies such as hibernate All of these are run on a separate database server.

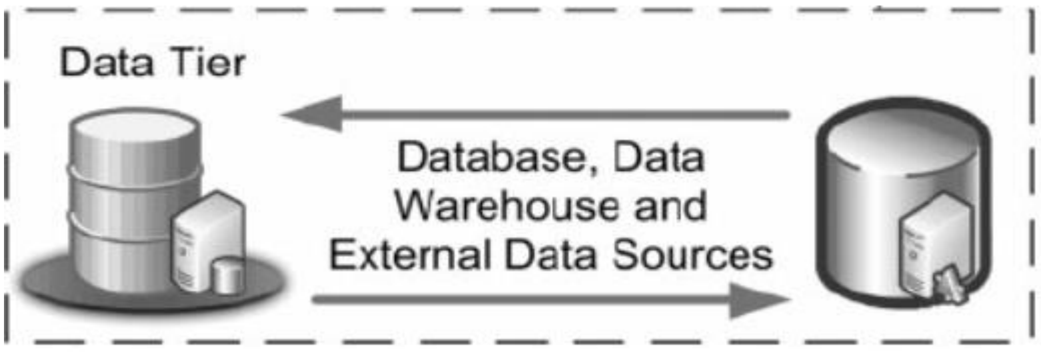


Figure 5 – 4 : Data-tier

5.2 Technologies used for the implementation

- **Spring Boot**

  Spring Boot is an open-source Java-based framework. It has comprehensive infrastructure support for developing a microservice. This framework, make the chance to configure all the things for yourself. But Spring Boot does not have a lot of configuration files. It chooses all dependencies, auto-configures all the features, and can start your application with one click. And it also simplifies the deployment. So it is auto-configuration, Standalone, Opinionated.

- **Hibernate**
  Hibernate is an ***object-relational mapping*** tool designed for java programming. It is become easier to develop a Java application and make it interact with the database. In this, entities or classes refer to the table in the database, an instance of classes refers to rows and attributes of instances of classes refers to a column of the table in the database. Here the performance is fast because the cache is internally used in the hibernate framework. It is lightweight and also provides the facility to create the tables of the database automatically. Hibernate Query Language is the object-oriented version of SQL. Simplifies Complex Join. Hibernate is it is database independent

- **AJAX**

  This is used for API calls. Ajax means asynchronous JavaScript and XML. It is a method of building interactive applications for the Web that process user requests immediately. It can combine several programming tools like java scripts, CSS, HTML, etc. Ajax uses a browser built-in XMLHttpRequest object and JavaScript and HTML.Ajax application acts as an intermediary between a browser and the server from which it is requesting information

- **Java Script**

  JavaScript uses to make web pages alive and interactive. But for this no need special preparation or compilation to run. This is used as a client-side scripting language. So it means that JavaScript code is written into an HTML page.

- **HTML 5**

  HTML5 is a programming language stands for HyperText Markup Language. It can find on a particular web page by knowing where the elements are and where to put the images and the texts. The basic structure of any page is basically defined in the HTML5. It supports Scalable Vector Graphics and consistent error handling.

- **Bootstrap 04**

  Bootstrap is a free, open-source framework .it contains HTML, CSS, and JS and also containing templates for text, forms, buttons, navigation, and other relevant interface components. It is compatible for latest versions of Google Chrome, Mozilla Firefox, Internet Explorer, Opera, and Safari browsers. So this can consider as the most popular front-end framework. It includes some set of predefined classes for building complex responsive layouts. Since the solution is given as a plugin to an existing product it is important handling consistency to design and code between projects

- **CSS**

  CSS stands for Cascading Style Sheets. It brings style to your web pages. CSS can re-position and it is better device compatibility. CSS definitions are saved in external CSS files. Because of that, it can change the entire website by changing just one file

- **Maven**

  This is the build automation tool for Java projects. It is an open-source tool that can set up local repository dependencies needed to build your project. It contains the below folder structure

5.3 Implementation of the presentation layer

Implementation of the presentation layer is the 1st step of researchers in the implementation process. In the design phase researcher has designed the prototypes for expected user interfaces. That prototype is used as a throwaway prototype. The whole system design is based on the prototype software development model. So it is starting from the Ui design and then moved forward to the nest steps such as database and logic implementations

For the implementation, the researcher has used JavaScript, HTML 5, Bootstrappedand CSS. The presentation layer is implemented by targeting the main 5 user groups Business analyst, QA, BA, Client, and Developer

(Actual UI screens are provided in appendix C).

5.4 Implementation of the service/ logical layer

This is the layer containing all the functionalities,

- Create components
- Add requirements
- Add issues
- Filter out the Issue vs Bugs

(Due to the lack of time, Researcher will be able to develop these functionalities).

With tools and technologies;

- For development – Java-based spring boot framework
- IDE – Eclipse
- For API testing – Postman
- Sever – Apache tomcat (Xaamp)
- Build automation tool- Maven

Here the researcher has specifically selected out the IDE that related to the maven build tool. Because of the development process, it has continued with the Maven build tool. Then continued the logical implementation with the Java and Spring boot. The Java main class which is started the project development 1st step is defined as below, Main Class = PmToolapplication.java. This is the code development initial step

```
package com.pm.tool;


import org.springframework.boot.SpringApplication;
import
org.springframework.boot.autoconfigure.SpringBootApplication;


@SpringBootApplication
public class PmtoolApplication {


        public static void main(String[] args) {
                SpringApplication.run(PmtoolApplication.class,
args);
        }}
```

When describing the controller folder structure, it is defining the class and its return types like the below examples. The class is flagged as a @RestController, by saying it is ready for use by Spring MVC to handle web requests. @RequestMapping maps / to the index() method. When invoked from a browser, the method returns. That is because @RestController combines @Controller and @ResponseBody, two annotations that result in web requests returning data.

```
@PostMapping("/post/addProject")
    public ResponseEntity<CommonResponse>
addNewCustomerController(@RequestBody ProjectsReqBean
projectsReqBean) {
        return
ResponseEntity.ok(pm_tool_service.addProject(projectsReqBea
n));
  }
```

Then in the services folder, it has included the PmToolsServices.java file by defining the response codes and define interfaces and parameters. The response codes are displayed below.

```
     static enum STATUS{
NOT_FOUND(404),INTERNAL_SERVER_ERROR(500),ACCEPTED(202),
       CREATED(201),  FOUND(302);
```

Then within the **Dao** folder, it includes the Impl implementation file for the defined interfaces

```
public class PM_Tool_Service_Impl implements
Pm_Tool_Service {

    @Autowired
    private SaveComponent saveCompo;

    @Autowired
    private RetriveComponents retCompo;

    @Override
    public CommonResponse addProject(ProjectsReqBean
projectsReqBean) {
        CommonResponse response = null;
        response = saveCompo.saveProject(projectsReqBean);
        return response;
  }
```

The components folder class files are defining to save and retrieve purposes. When analyzing the way of retrieving a bug or CR it is defined as the component Retrieve components. Java as below.

```
}
```

```
        String issueType;
        if (titleList.isEmpty()){
            issueType="Change Request";
        }else {
            issueType="Bug";
        }
  return issueType;
```

In SaveComponet.java it has defined the logic to save the data. This is applicable when saving the data to the database

```
public CommonResponse saveModule(ModuleReqBean
moduleReqBean){

        ProjectsEntity project =
pm_tool_service.getProjectByName(moduleReqBean.getProjectNa
me());

        ComponentStatusEntity compoStatus = new
ComponentStatusEntity();
        ComponentStatusEntity newStatus = new
ComponentStatusEntity();
        try {
            compoStatus =
pm_tool_service.getCStatBYStat(true);
            if (compoStatus==null){
                newStatus.setStatus(true);
            }
        } catch (Exception e){
            compoStatus.setStatus(true);
  }
```

Finally, on the repository folder, it holds the interfaces that extend to the database queries that related to the source code to get the data and give the outputs. All the interfaces are implements on separated .java file to each function like the below example.

```
public interface InterfaceRepo extends
JpaRepository<InterfacesEntity, Integer> {

    @Query(value = "select * from interfaces p WHERE
p.interface_name= :interface_name", nativeQuery = true)
    InterfacesEntity getInterfaceByName
(@Param("interface_name") String interface_name);
```

```
    @Query(value = "select * from interfaces p WHERE
p.module_id= :module_id", nativeQuery = true)
    List<InterfacesEntity> getIntByModName
(@Param("module_id") Integer module_id);
}
```

(Detail code on main logics are provided in appendix D).

5.5 Implementation of the database layer

As the 3rd layer of the implementation, it is identified as the DataBase layer. This depends on the entity folder content. The entity folder includes the entity.java class file.

These files are derived as the entities that annotated with the @Document annotation. The field in the entity should be annotated with the @Field annotation. This clearly shows the intent and design of the entity. There another annotation as @Id annotation. It needs to be always in place for name the property id.

This is implemented as the below example,

```
@Getter
@Setter
@NoArgsConstructor
@Entity
@Table(schema = "CHANGE_REQUEST_SCHEMA", name =
"CHANGE_REQUEST")
@SequenceGenerator(name = "SEQ_CR_ID",initialValue =
1,allocationSize = 1)
public class ChangeRequestEntity implements Serializable {
    @Id
    @GeneratedValue(strategy = GenerationType.SEQUENCE,
generator = "SEQ_CR_ID")
    @JsonProperty
    @Column(name = "cr_id", unique = true, nullable =
false)
private Integer crId;
```

Here when you are generating the tables it is maintaining a separate table as sequence tables for escape the error occurs for auto-generated ID sequence. Each auto-generated ID table creates its separate sequence to manage this issue.it is defining with this annotation @SequenceGenerator(name = "SEQ_CR_ID",initialValue = 1,allocationSize = 1). And it is used another annotation to maintain the sequence notations like @GeneratedValue(strategy = GenerationType.SEQUENCE, generator = "SEQ_CR_ID") , For this, it is used the technology called Hibernate. It is an object-relational mapping tool designed for java programming,

The entity represents a single instance of the object that saved into the database as data columns. It manged the attributes that represent columns in tables. The model represents a real-world object that is related to the problem or domain space. This creates classes to represent objects. These classes, manged the models, with some properties and methods. The entity classes are defined as plain Java classes with some annotations, Below classes are added to this solution as .java classes.

Here it has implemented one class for each table. The main logic od this solution is depended on the tables (classes) **Requirement** and the **ProjectIssues,** The **Requirement table** consists of the initial requirement. And the user reported issues to go to the **ProjectIssues table** When the issues are adding to the table it is compared with the **Requirement table.** And if it already exists on the requirement table it is taken as a bug and if it does not exist it is taken as a CR. This is the point that achieved the main target of the system.

(SQL queries are provided in appendix E).

(Class diagram for Database are provided in appendix F).

The main advantage of this database design is you do not need to create a separate database When you just created a database with the expected name When you run the. Jar file the database is automatically created with all dependencies.

5.6 Limitations and the issues related to the implementation

Since the researchers selected this research question as to her industrial experiences and the problem already faced, the issue is completely solved with the design phase. But since the researcher is not an official developer. It is spent more time to research the development of concepts and tools and technologies. And also this is expected to add to full fill a limitation of an existing tool, for the solution has to selected the new technologies used. Since the researcher took much time to study the concepts, the whole system is unable to develop. Even the logical basis API has introduced the researcher unable to fix it with the user interfaces. But all the user interfaces are completed with full expected design. Since the researcher is a Quality assurance Engineer, some limitations happened due to the lack of technical knowledge.

5.7 Chapter Summary

This chapter has discussed the full implementation part of the solution. And id described what is the development architecture that used for development. And what that architecture means and why it is selected and what are the advantages of using that architecture, Then it explained the step by step of the architecture. And also explained what are the tools and technologies used on the development, And what each tool and technology has to do. Then described the implementation depends on each architecture step.it described the method of implementing the user interfaced and what are the highlighted points. Then it explained the logical layer implementation. What are the concepts and the tools are used for the implementation? Then it is described the final layer database layer and its methods and technologies and finally about the limitations related to the implementation.

CHAPTER SIX
CONCLUSIONS

This chapter concludes the thesis. So this is briefly revisited the motivation for the research and why this topic is selected. The challenge that identified from the industry. Technical background analysis with the advantages of used architectures.And also reflections on the main findings. And a cost-benefit analysis for the outcome, and how to deal with the actual data. Finally, the summary in section concludes the chapter.

6.1 Motivation Revisited

The research reported in this thesis sets out to explore the challenges of the software industry practitioners faced by managing requirements change in projects At the same time what are the associated role of collaborative technologies. This study sought to understand why managing requirements change is difficult in industrial practices, And why ongoing practices and efforts are still having the limitations. With this topic, it is identified the manage requirements change were analyzed to better understand with the clients' needs and minimize distance-related challenges. Another motivation was to develop a model based on the core requirements change management tool to support the limitation of existing tools and enhanced the standard process visibility. And here trying to manage the tasks related to requirements change management by the role of associated context. With the researcher's own experiences as a Quality Assurance Engineer, there are mainly find out 7 issues that have been evaluated and solved out with this research solution, the issues mentioned below are the issues based on the solution.

Issue 1 - Issues regarding reporting the client feedbacks in the concept of Agile Software Development and the Software as a service

Issue 2 - Issues regarding handling the old existing Requirements Vs New Reported Issues, And compare and contrast them

Issue 3 -Issues regarding defining parameters and standards to filter out Bugs and issues.

Issue 4- Issues regarding getting clarifications within a global team that spread around different geo-locations.

Issue 5 -Issues regarding finding out impact analysis.

Issue 6 -Varying the time estimations due to the wrong impact analysis.

Issue 7 -Having some arguments and Conflictions within the team

Issue 1 - Issues regarding reporting the client feedbacks in the concept of Agile Software Development and the Software as a service

The client will use different formats to report their feedbacks such as group calls, discussions, emails, etc. So the team will be faced with different issues when collecting the change requirements. When reporting through the emails, handling emails will be a huge mess. If one change is reported without CC other people, maybe that point is with that only person, IT will occur many issues if that resource it out of the tea when they need it. Maybe having some language-related issues due to the client will have a lack of knowledge with the system, So this solution makes a common platform and a common language to handle this issue.

Issue 2 - Issues regarding handling the old existing Requirements Vs New Reported Issues, And compare and contrast them

Usually, the SRS is written in a document format. So the team members who reding the full document is taken much time. When analyzing the issues the team has to read all points very carefully.

And manging the document also gets more effect. This will be a time wastage. But the solution will handle these issues very easily and readable to the requirement changes

Issue 3 -Issues regarding defining parameters and standards to filter out Bugs and issues.

Some conflictions were happening when filtered out change requirements and bugs, due to the uncertainty of the pre-requirements Even the main scenario of the defining change requirement is comparing with the initial requirements this thing is not handled properly on the project environment Most projects have no proper accepted standard for this within clients and team .Defining CR as a bug will be a loss to the company at the same time defining bug as a CR will be a loss to the client. So this solution makes it standardized.

Issue 4- Issues regarding getting clarifications within a global team that spread around different geo-locations.

In most situations, current software teams have distributed teams Different team members are in different locations, Maybe they have outsourced resources or virtual teams or Maybe they worked under the globalization concepts. So most of them are used tools. But even they used project management tools, they have some limitations with the CR management sector. So this web-based application makes them easy access from anywhere and easy work assigned facility.

Issue 5 -Issues regarding finding out impact analysis.

If there are new resources without having the proper idea about the pre-requirements and Changes they are unable to define actual impact analysis, So this system has automated the function of defining the Bug and CR automatically.

Issue 6 -Varying the time estimations due to the wrong impact analysis.

When the teammates are unable to find the impact analysis properly, the estimations such as time estimations and the cost benefits analysis will go wrong .Then the team has to work hard and achieve tuff schedules Overstressed teams. So the improved solution of this concept will display the number of impacted count of the components and fro which module they have belonged.

Issue 7 -Having some arguments and Conflictions within the team

They will have some doubts about the responsible person. Have no proper idea on, from whom it missed and all the blams comes to the QA team, So as a solution this solution includes an assigning part of the responsible person

The most important thing is with this solution from the client-side they don't know whether they are entering a bug or CR. They just entered an issue that they have seen. Or it may be a new suggestion. So the client is just entering a project issue as they trained with the user manuals. So the solution will be responsible for filtering out the Bug or CR, mainly this solution is with the main 3 modules that process.

1st module is to filter out the bug and CR, Then the 2nd module is to impact analysis, Finally, the 3rd module is to assign and updated the task status. With the limitations of the technology and the time the researcher is only able to implement the module of defining the bug or Cr when the issue is entered into the system. But the UI design for all the 3 modules is already developed. So for the future researchers can continue and develop the next 2 modules.

6.2 Research limitations

The main limitations are occurred by analyzing the JIRA and confluence structures. Some of these technologies are unable to filter out properly due to those pieces of information are private and confidential. So the researcher has got some assumptions to handle these limitations.

Assumption 01 – When the solution is used as a plugin to JIRA, the solution is using the same user management database and the roles created tables

Assumption 02 – And the concept that used in JIRA as project and the sprint is the same to this solution. this project table can be merged with the existing Jira project table and the concept of sprint also can be merged to the Jira sprints

Assumption 03 – Since JIRA has many plugins related to different tasks this solution can be introduced to JIRA as a separate plugging.

Assumption 04 – But if needed this can be used as a separate solution

When completing all findings when the researchers moved to the implementation many limitations have to face. The main thing is selecting the best technology for implementation. So the researcher has to spend more time finding the technologies. Since the researcher is not having a position related to the development, this will be the most difficult part of the research. For a quality assurance engineer have not many experiences related to the implementations. It takes a lot of time to study the technologies from top to bottom. After spent more than 2 months the researcher finalized the architecture with the 3 layer architecture depend on some assumptions.

Assumption 05 – with the 3 tiered architecture the user will be able to change the technology for any tier with the future technology improvements. Because each layer is independent with the other layer.

Then depending on the 3 tiered architecture, the researcher has to select the best tools and technology to implement. This also took a lot of time, When selecting the need a lot of tutorials and guidance needed to start the implementation step. And especially when this solution is introduced as an international level global solution to the industry this should be standards. So even there are some limitations the researchers have selected some new technologies such as Hibernate, Spring boot, etc, So it took much time to self-studying process to the researcher as a QA Engineer. Since the QA engineers are not involved in the development process, there were not many practical experiences with the development of the researcher. So have to start with some models and get some guidance from the experienced developers and recreated the modules. When it getting some practical technical level dependency issues such as Lombok issue, it seems very difficult to handle with the relative plugins and jar files. Since the researcher has tried a lot to solve some issues there are some points that unable to solve, due to that researcher has to change some logic and methods. With all these limitations it took more than 4 months to solve the basic levels of issues and complete the study on basics. So the main limitation happened with the ***Estimated Time Plan***. And also with ***the lack of technical knowledge.***

Then as the next point have to make another assumption from the client-side.

Assumption 06 – When handover the project for the development team the team should give the training to how to report the issue on the selected member from the client-side issued a documented agreement and a user manual to the client.

As usually when starting any project the team and the client signed for an agreement this expected process should continue with it.

Since the issue with a lack of implementation knowledge, few limitations happen. Even the researcher able to complete the one process successfully it takes a lot of processing time to give the output when there is a lot of data in the system. The points such as search Bug and CR and get the count of all the filtered results So this needs to get some expert knowledge on the data processing query optimization. This includes a huge data search functionality, So when completed the full development it is important to have proper query optimization. These search queries should be executed with some validations. So the researcher has to face some slowness issues. So the researcher needs support for the optimization of the solution. Since all these technical issues the researcher decided to implement this as a model solution and later this is handover for a proper development team and achieved the target as a proper implementation solution.

When analyzing the research topic the most difficult thing is become to narrow down the topic. Because mapping project requirements for a component hierarchy is not an easy task.it will become a wast area of the subject. So this hierarchy is started from the very high level of the project, And then moves the module and interface wise. So taking the next step to component types researcher moves for another assumption.

Assumption 07 – When developing an IT solution there will be many component types that we can define but, In here the solution gets narrow down under 4 component types such as Functional, Non- functional, UI, and Validations. And also assume that in future this can be expanded.

And the other hand this kind of an implementation project should have a good code review and the code optimization methods. So this has to handover to an industrial level expert to do the optimization

As the main limitations of this modeled solution, this is highlighted with the technical issues and the estimated time

6.3 Analysis of system advantages

6.1.1 Main advantages of the researcher's professional background

As a quality assurance engineer, it is easy to derive the test scenarios when the requirements are derived from components levels. Then the QA can cleary find out how many test scenarios are needed to verify the solution very easily. When it receives the requirements according to the component level. It makes it easier and will make fewer issues with missing the scenarios. And also this supported reusing the test cases. When the issues that identified as a Bug, the QA Engineer can reuse the old test cases, At the very 1st view with all the component that clearly defined for the reported bug, then it supports to clarify what are the scenarios that need to execute with the smoke test cycle and for which scenarios that should add to the extra effort.

And as another advantage, there are many conflicting occurs while it happens the bugs. And the blams are coming to the QA. But the actual thing is most of the issues are happening due to the complexity of the requirement and unable to track all requirements. And also when the changes are happening it is unable to get the impact analysis. So this helps to simplify the requirement complexity and make understands without getting any confusion. And also supported to avoid the language and communication issues.

And this solution will be able to maintain the requirements change management been endemic to software development and obtaining the extra effort pressures on development teams, And this will able to develop suitable guidelines and practices in requirement change management. Finally, this will become a useful purpose, in future essential software development strategies for introducing best practices to accommodate error-free products that meet clients exact requirements And satisfaction

6.1.2 Advantage of the used technologies

The researcher has selected the 3 tier architecture for the development process because it is easy to maintain the solution to any environment. Since this solution is expecting to add as a plugin it is easy to maintain the Interfaces, Logics, and the database separately. And the main advantage is this is designed to run a Jar file. Then when you handover this to the client there won't any code visibility to client-side

6.1.3 Cost-benefit analysis

Since in the software industry, the team is considered a bug as a loss and a CR as a benefit this is a huge problem that, filtered out the Bug and Change Requirement properly. So within this simplification of the requirements analysis, it makes it easier to understand the requirement. And the team can clearly be defined what is a bug and what is CR so they can manage the future issues with the clients. So this is supposed to manage the project cost and benefits. So this solution makes the best benefit for current industrial development teams and companies buy reduced the cost and make more profits

By using this solution companies can reduce the cost of printed papers that maintained to store the requirements. And also getting the proper impact analysis this will save the wast of time, extra efforts and wast for the resources

6. 4 Future work of the project

This study has been conducted as a conceptual tool and implemented only the module that able to derived Bug or change request, But presentation layer user interfaces are implemented for all 03 modules for all the targeted users, At the same time the database also fully developed will all the entities. So further futures researcher will be able to develop the implementation for issue allocation and impact analysis modules. And will be able to introduce new improvements for this concept by analyzing the limitations with this concept.

6.5 Chapter Summary

This final chapter of the thesis has reiterated the motivation for this study into the challenges of managing requirements change management and the limitations of the research and also what are the assumptions and finally how will it get the advantage for the software industry.

References:

[1] Hussin Ahmed, Azham Hussain, Fauziah Baharom. (2016).Current Challenges of Requirement Change Management, [Online].Available: https://www.researchgate.net/publication/313718011_Current_challenges_of_requirement_change_management

[2] Waqur Hussain(2016) Requirement_change_management in global software development[Online].Available: https://pdfs.semanticscholar.org/b163/f585275a742fd7a2e4e972ed.pdf

[3] Hassan Osman Ali1, Mohd Zaidi Abd Rozan 2, Abdullahi Mohamud Sharif ()Change Requirement Management Issues for a Large Software[Online].Available: https://pdfs.semanticscholar.org/8690/57a76d34e42dd3e1560b15b3f338.pdf\

[4] S. M. Ghosh1, H. R. Sharma1, V. Mohabay2 (2011), A Study of Software Change Management Problem [Online].Available: http://article.nadiapub.com/IJDTA/vol4_no3/4.pdf

[5] Ahmed Mateen1 and Hina Amir2 (2016) enhancement in the effectiveness of requirement change [Online].Available: https://arxiv.org/ftp/arxiv/papers/1605/1605.00770.pdf

[6] Nasir Mehmood Minhas, Qurat-ul-Ain, Zafar-ul-Islam, Atika Zulfiqar (2014).An Improved Framework for Requirement [Online].Available: https://www.scirp.org

[7] Domia Lloyid,Ramadan Moawad,Mona Kadry (2017). A-supporting-tool-for-requirements-change-manage [Online].Available: https://www.sciencedirect.com/science/article/pii/S2314728817300053

[8]MuhammadAzeemAkbara,b,JunSanga,b,⁎,Nasrullaha,b,ArifAliKhanc,⁎,SajjadMahmoodd, SyedFurqanQadrie,HaiboHua,b,HongXianga (2019).Success factors influencing requirements change management processin global software development [Online].Available: https://www.sciencedirect.com/science/article/pii/S1045926X1830141

[9] PinarEfe⁎,OnurDemirors (2019). A change management model and its application in software development projects [Online].Available: https://www.sciencedirect.com/science/article/abs/pii/S092054891830

[10] Ainhoa Aldave, DavidGranada, EsperanzaMarcos (2019).Leveraging creativity in requirements elicitation within agile software development A systematic literature review [Online]. Available:
https://www.sciencedirect.com/science/article/pii/S0164121219301712

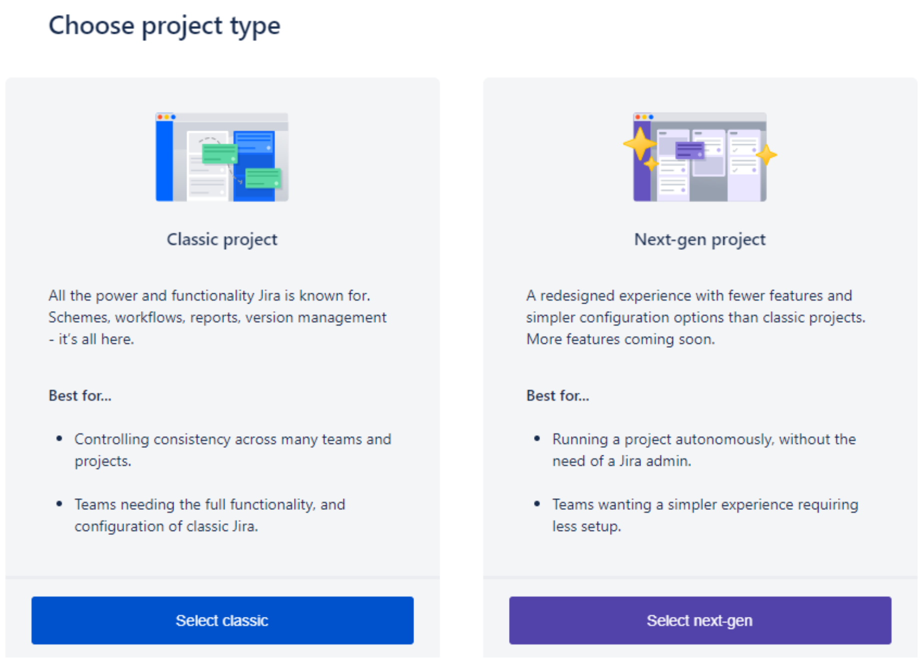
[11] Chen Yanga,b, Peng Lianga,*, Paris Avgerioub (2018).Assumptions and their management in software development A systematic mapping study [Online].Available: https://www.sciencedirect.com/science/article/abs/pii/S095058491630

[12] Reginald Putra Ghozalia, Herry Saputraa, M. Apriadin Ariadi Nugrohoa (2019). 12 - Systematic Literature Review on Decision-Making of Requirement Engineering from Agile Software Development [Online].Available:
https://www.sciencedirect.com/science/article/pii/S1877050919310853

[13]Ambler, S. W. (2014). Agile Requirements Change Management. Retrieved January 2, 2016, from http://agilemodeling.com/essays/changeManagement.htmManagement-In-Depth-Report.ashx

APPENDICES

7. 1 APPENDIX A - JIRA EXISTING INTERFACES


Choose project type

Classic project

All the power and functionality Jira is known for. Schemes, workflows, reports, version management - it's all here.

Best for...

- Controlling consistency across many teams and projects.
- Teams needing the full functionality, and configuration of classic Jira.

Select classic

Next-gen project

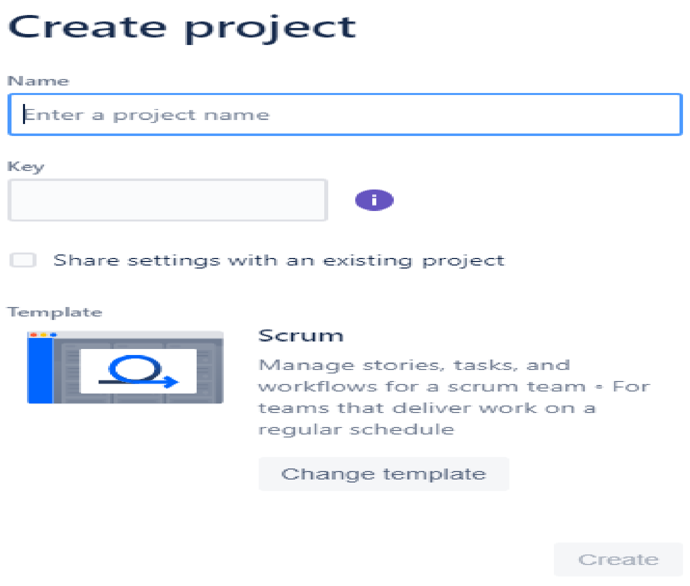
A redesigned experience with fewer features and simpler configuration options than classic projects. More features coming soon.

Best for...

- Running a project autonomously, without the need of a Jira admin.
- Teams wanting a simpler experience requiring less setup.

Select next-gen

Figure 7-1 – Jira Create project UI 01


Create project

Name

Enter a project name

Key

Share settings with an existing project

Template

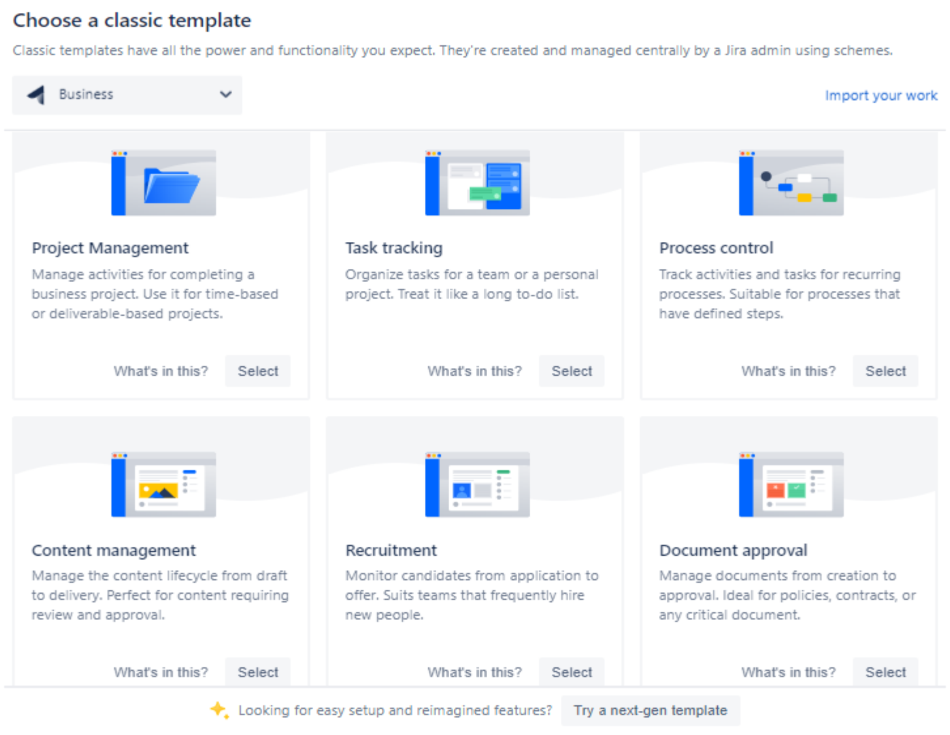
Scrum

Manage stories, tasks, and workflows for a scrum team • For teams that deliver work on a regular schedule

Change template

Create

Figure 7-2 Jira Create project UI 02


Choose a classic template

Classic templates have all the power and functionality you expect. They're created and managed centrally by a Jira admin using schemes.

Business

Import your work

Project Management
Manage activities for completing a business project. Use it for time-based or deliverable-based projects.
What's in this? Select

Task tracking
Organize tasks for a team or a personal project. Treat it like a long to-do list.
What's in this? Select

Process control
Track activities and tasks for recurring processes. Suitable for processes that have defined steps.
What's in this? Select

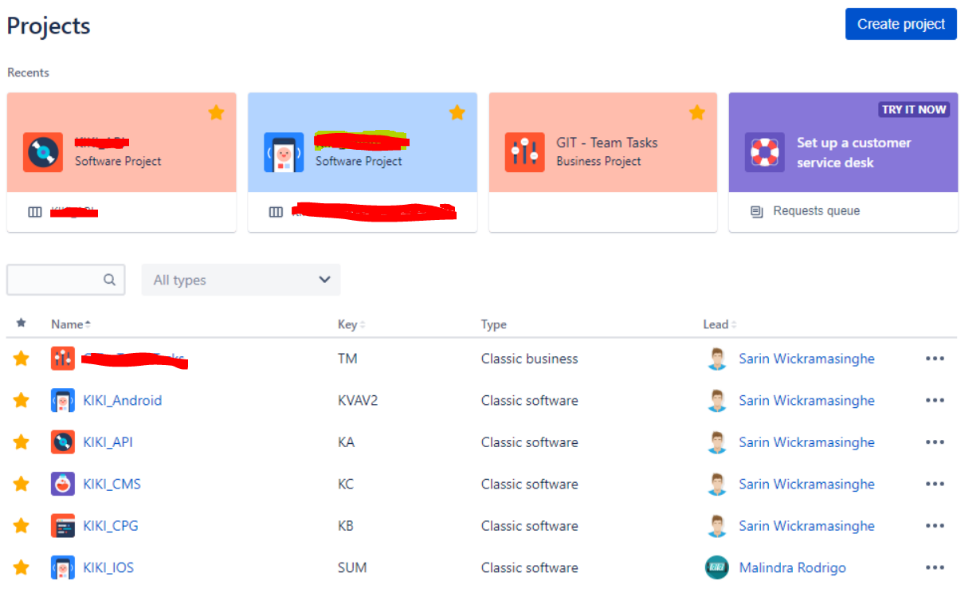
Content management
Manage the content lifecycle from draft to delivery. Perfect for content requiring review and approval.
What's in this? Select

Recruitment
Monitor candidates from application to offer. Suits teams that frequently hire new people.
What's in this? Select

Document approval
Manage documents from creation to approval. Ideal for policies, contracts, or any critical document.
What's in this? Select

Looking for easy setup and reimagined features? Try a next-gen template

Figure 7-3 Jira Create project UI 03


Projects

Create project

Recents

[redacted] Software Project — [redacted]

[redacted] Software Project — [redacted]

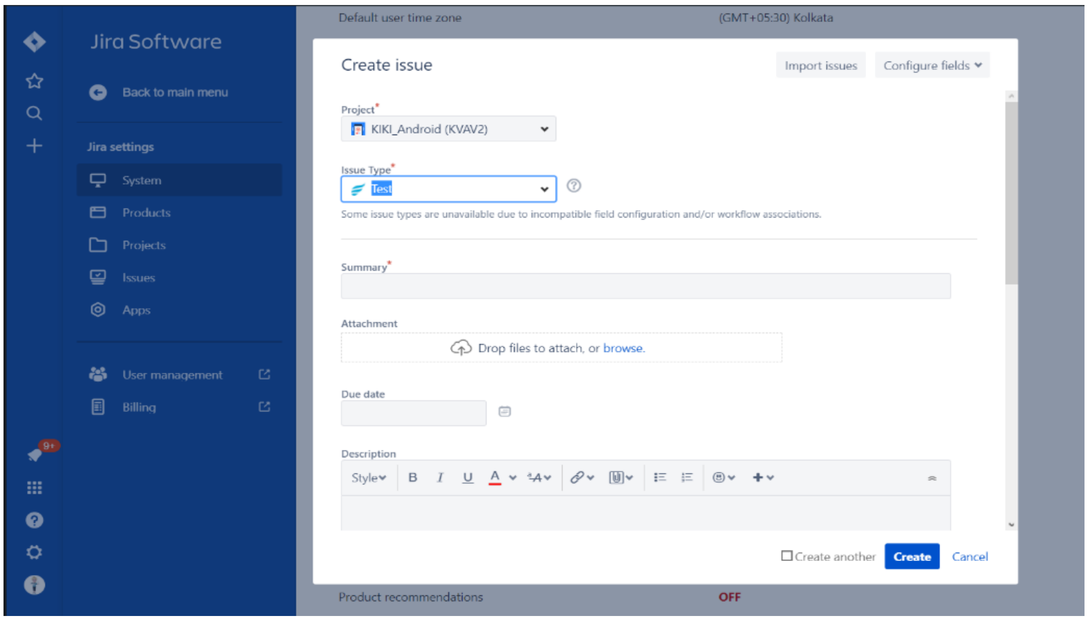
GIT - Team Tasks Business Project

TRY IT NOW Set up a customer service desk — Requests queue

All types

★	Name	Key	Type	Lead
★	[redacted]	TM	Classic business	Sarin Wickramasinghe
★	KIKI_Android	KVAV2	Classic software	Sarin Wickramasinghe
★	KIKI_API	KA	Classic software	Sarin Wickramasinghe
★	KIKI_CMS	KC	Classic software	Sarin Wickramasinghe
★	KIKI_CPG	KB	Classic software	Sarin Wickramasinghe
★	KIKI_IOS	SUM	Classic software	Malindra Rodrigo

Figure 7-4 Jira Create project UI 04



Default user time zone (GMT+05:30) Kolkata

Create issue

Project* KIKI_Android (KVAV2)

Issue Type* Test

Some issue types are unavailable due to incompatible field configuration and/or workflow associations.

Summary*

Attachment

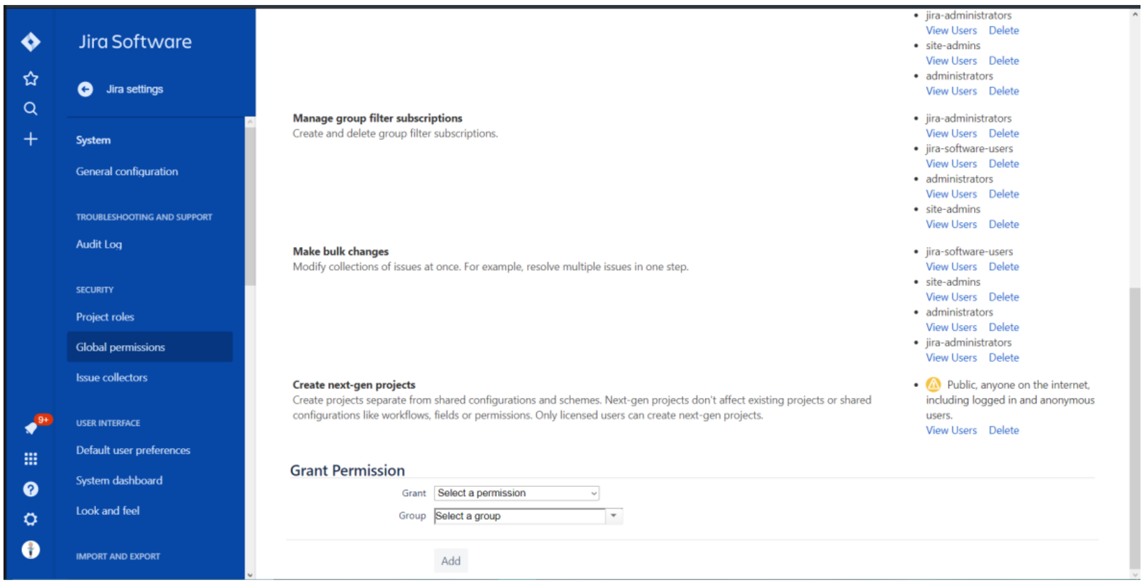
Drop files to attach, or browse.

Due date

Description

Product recommendations OFF

Figure 7-5 Jira Create tasks, test, bugs UI



- jira-administrators View Users Delete
- site-admins View Users Delete
- administrators View Users Delete

Manage group filter subscriptions
Create and delete group filter subscriptions.

- jira-administrators View Users Delete
- jira-software-users View Users Delete
- administrators View Users Delete
- site-admins View Users Delete

Make bulk changes
Modify collections of issues at once. For example, resolve multiple issues in one step.

- jira-software-users View Users Delete
- site-admins View Users Delete
- administrators View Users Delete
- jira-administrators View Users Delete

Create next-gen projects
Create projects separate from shared configurations and schemes. Next-gen projects don't affect existing projects or shared configurations like workflows, fields or permissions. Only licensed users can create next-gen projects.

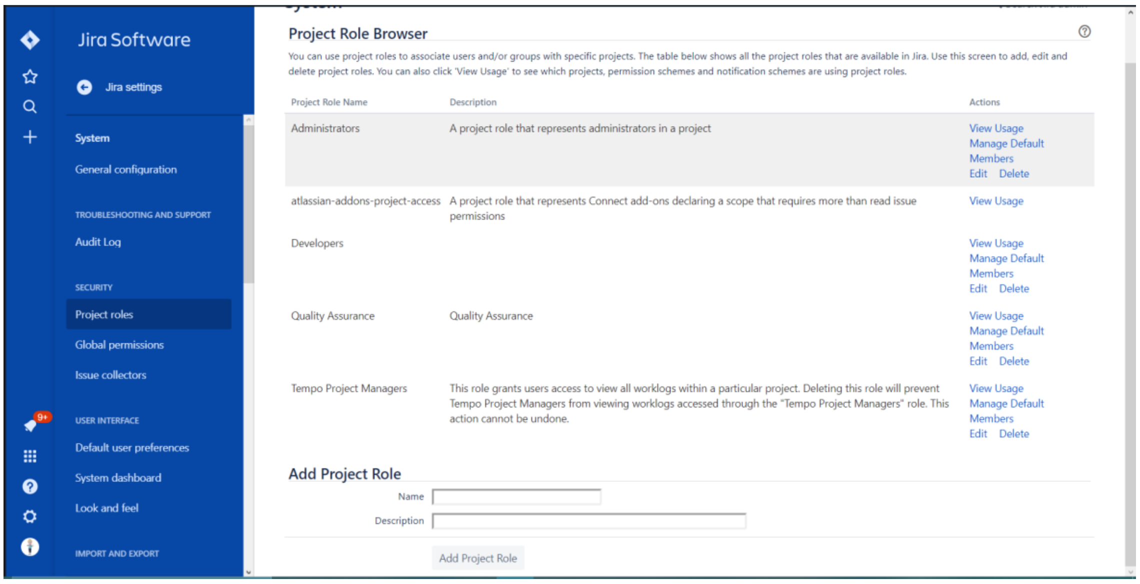
- Public, anyone on the internet, including logged in and anonymous users. View Users Delete

Grant Permission

Grant Select a permission

Group Select a group

Figure 7-5 Jira Manage permission UI



Project Role Browser

You can use project roles to associate users and/or groups with specific projects. The table below shows all the project roles that are available in Jira. Use this screen to add, edit and delete project roles. You can also click 'View Usage' to see which projects, permission schemes and notification schemes are using project roles.

Project Role Name	Description	Actions
Administrators	A project role that represents administrators in a project	View Usage Manage Default Members Edit Delete
atlassian-addons-project-access	A project role that represents Connect add-ons declaring a scope that requires more than read issue permissions	View Usage
Developers		View Usage Manage Default Members Edit Delete
Quality Assurance	Quality Assurance	View Usage Manage Default Members Edit Delete
Tempo Project Managers	This role grants users access to view all worklogs within a particular project. Deleting this role will prevent Tempo Project Managers from viewing worklogs accessed through the "Tempo Project Managers" role. This action cannot be undone.	View Usage Manage Default Members Edit Delete

Add Project Role

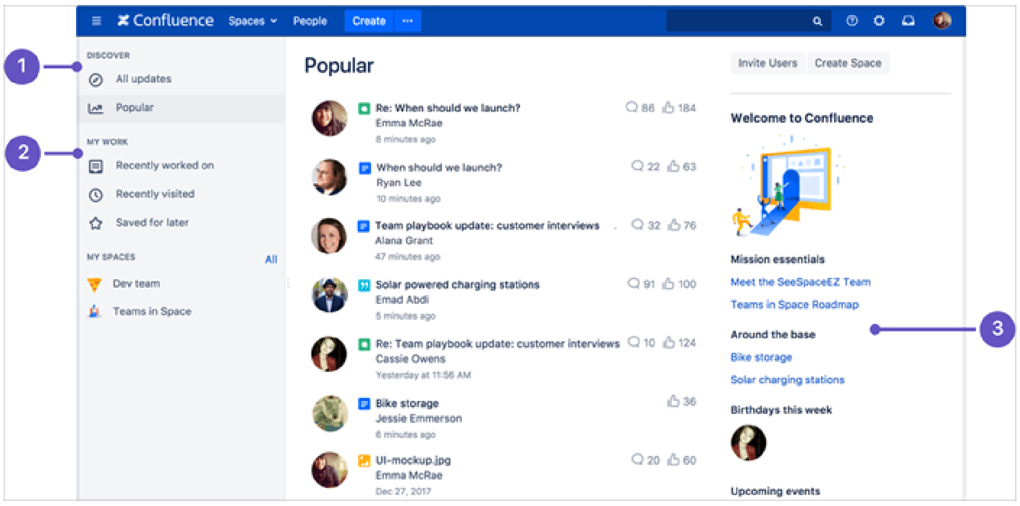
Name

Description

Add Project Role

Figure 7-6 Jira Create roles

7.2 Appendix B - Confluence existing interfaces



Popular

Re: When should we launch? Emma McRae 8 minutes ago 86 184

When should we launch? Ryan Lee 10 minutes ago 22 63

Team playbook update: customer interviews Alana Grant 47 minutes ago 32 76

Solar powered charging stations Emad Abdi 5 minutes ago 91 100

Re: Team playbook update: customer interviews Cassie Owens Yesterday at 11:56 AM 10 124

Bike storage Jessie Emmerson 6 minutes ago 36

UI-mockup.jpg Emma McRae Dec 27, 2017 20 60

Welcome to Confluence

Mission essentials

Meet the SeeSpaceEZ Team

Teams in Space Roadmap

Around the base

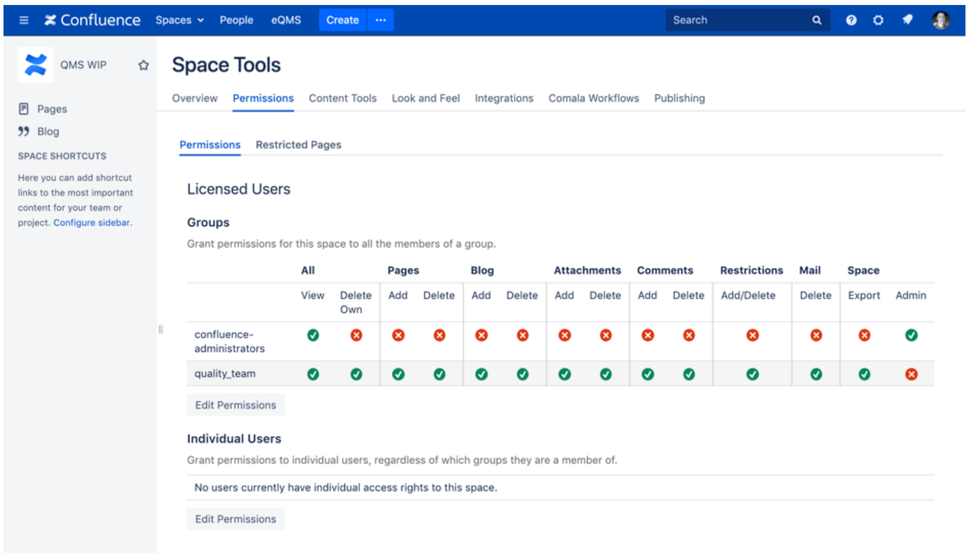
Bike storage

Solar charging stations

Birthdays this week

Upcoming events

Figure 7-7 Confluence Manage permissions


Confluence · Spaces · People · eQMS · Create · Search

QMS WIP

Pages

Blog

SPACE SHORTCUTS

Here you can add shortcut links to the most important content for your team or project. Configure sidebar.

Space Tools

Overview · Permissions · Content Tools · Look and Feel · Integrations · Comala Workflows · Publishing

Permissions · Restricted Pages

Licensed Users

Groups

Grant permissions for this space to all the members of a group.

	All		Pages		Blog		Attachments		Comments		Restrictions	Mail	Space	
	View	Delete Own	Add	Delete	Add	Delete	Add	Delete	Add	Delete	Add/Delete	Delete	Export	Admin
confluence-administrators	✓	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✓
quality_team	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✗

Edit Permissions

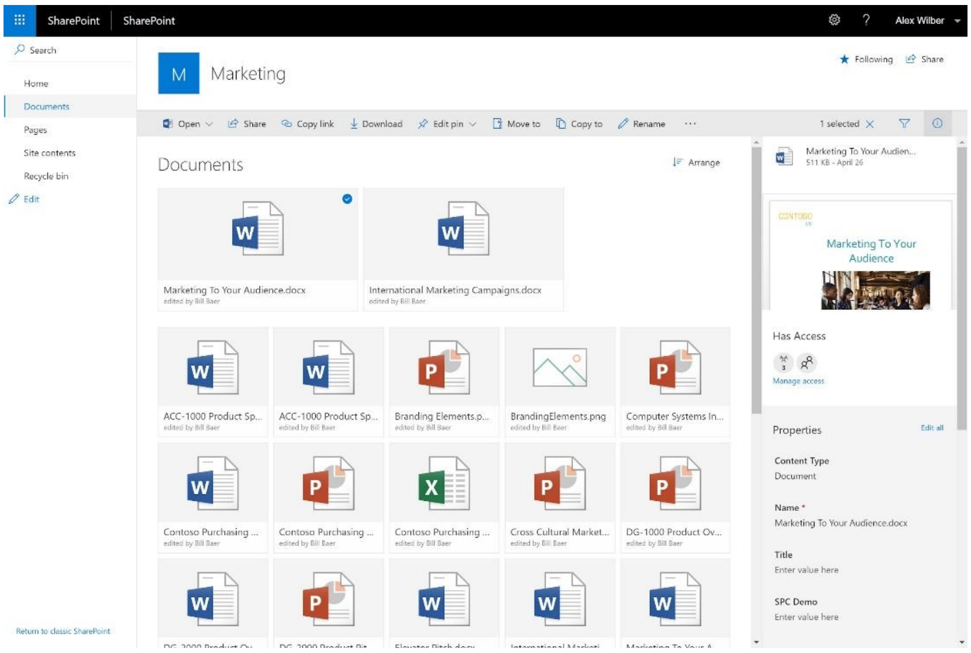
Individual Users

Grant permissions to individual users, regardless of which groups they are a member of.

No users currently have individual access rights to this space.

Edit Permissions

Figure 7-8 Confluence Web page UI of requirement details


SharePoint · SharePoint · Alex Wilber

Search · Home · Documents · Pages · Site contents · Recycle bin · Edit · Return to classic SharePoint

Marketing · Following · Share

Open · Share · Copy link · Download · Edit pin · Move to · Copy to · Rename · 1 selected

Documents · Arrange

Marketing To Your Audience.docx — edited by Bill Baer

International Marketing Campaigns.docx — edited by Bill Baer

ACC-1000 Product Sp... · ACC-1000 Product Sp... · Branding Elements.p... · BrandingElements.png · Computer Systems In...

Contoso Purchasing ... · Contoso Purchasing ... · Contoso Purchasing ... · Cross Cultural Market... · DG-1000 Product Ov...

Marketing To Your Audien... 511 KB - April 26

Marketing To Your Audience

Has Access · Manage access

Properties · Edit all

Content Type: Document

Name *: Marketing To Your Audience.docx

Title: Enter value here

SPC Demo: Enter value here

Figure 7-9 - Confluence UI for uploaded documents

7. 3 Appendix C - Actual Developed user interfaces

<u>Client section</u>

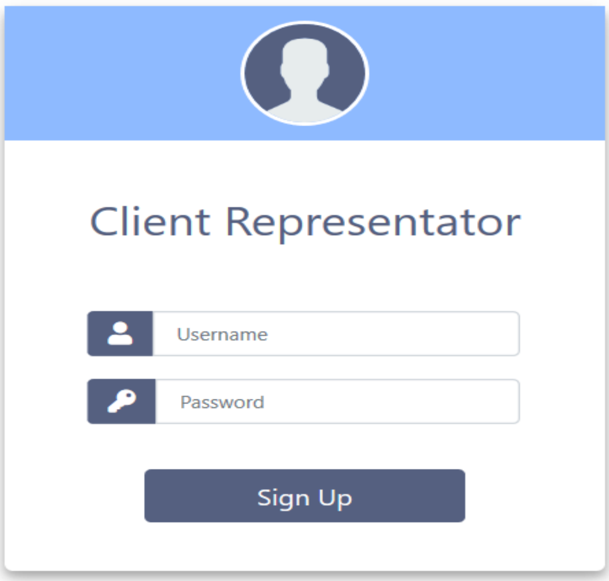


Figure 7-10 - Client login page

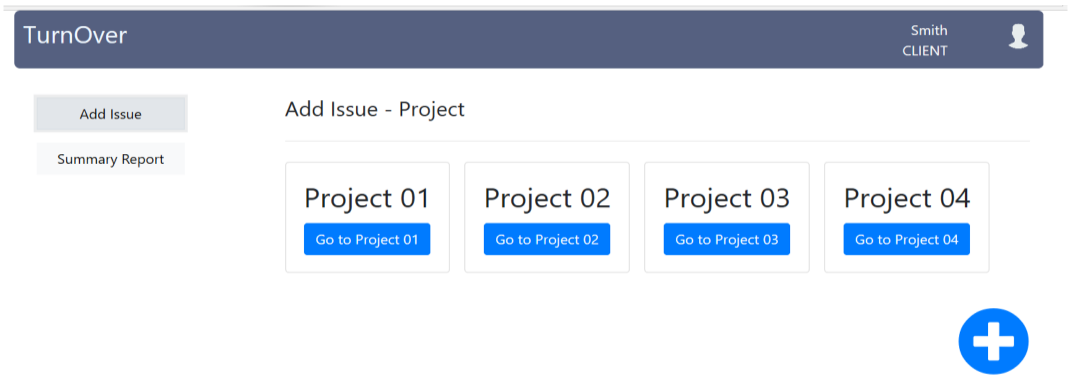


Figure 7-10 - Select project

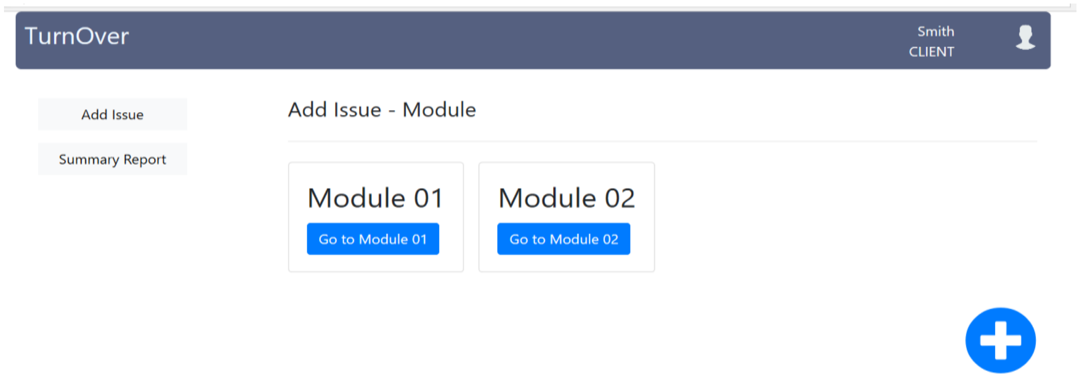


Figure 7-11 Select module

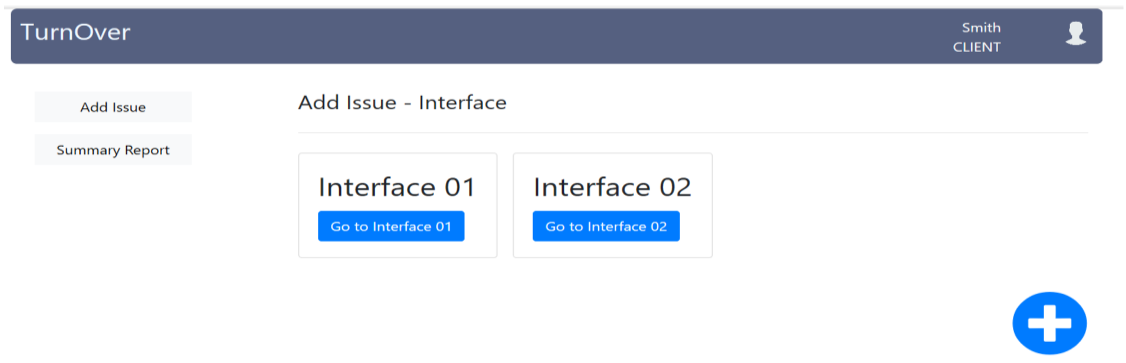


Figure 7-12 Select interface

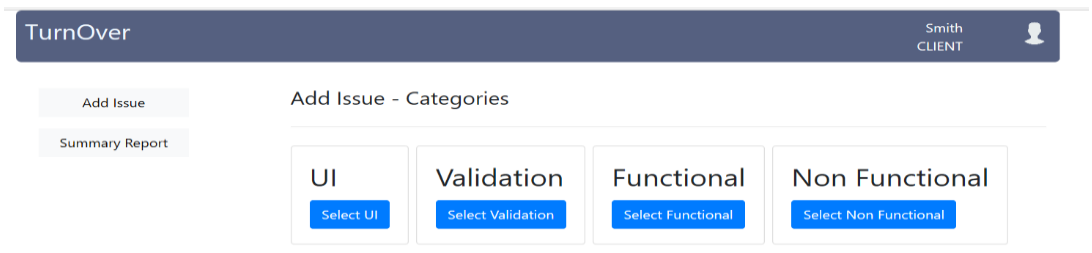


Figure 7-13 Select Component type

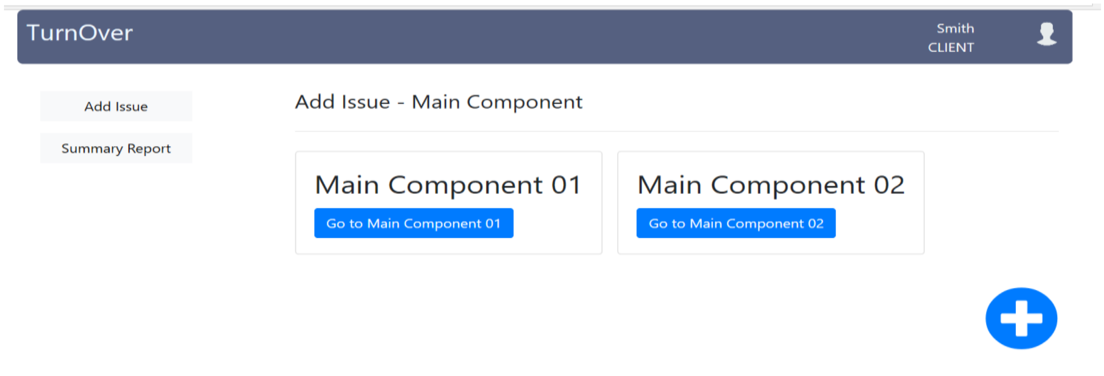


Figure 7-14 Select the main component

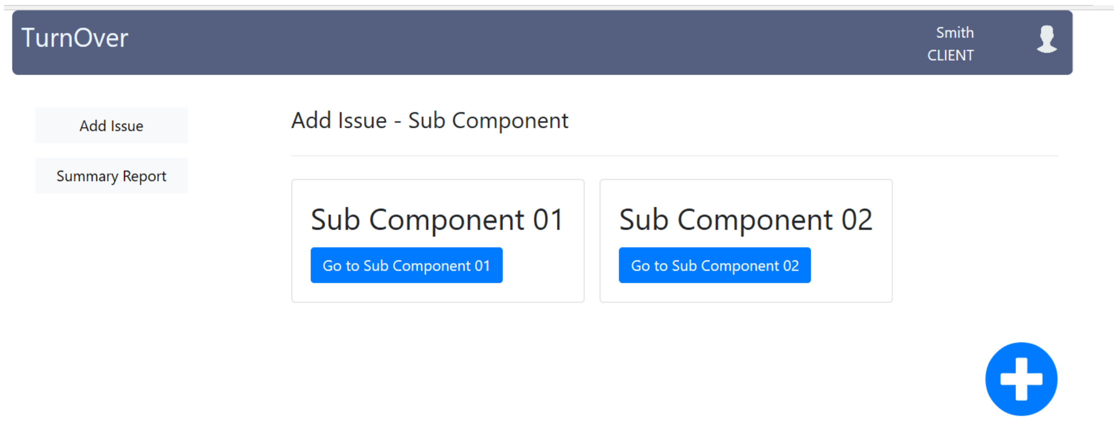


Figure 7-15 Select the subcomponent

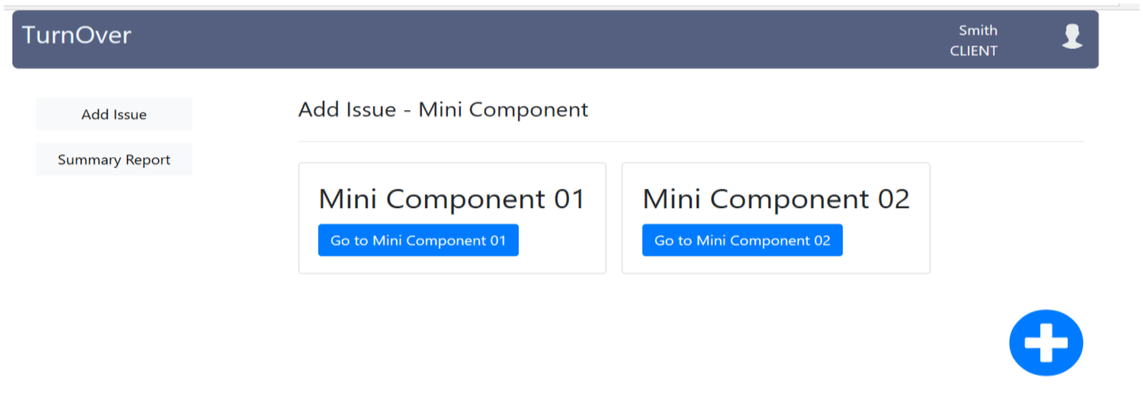


Figure 7-16 Select the mini component

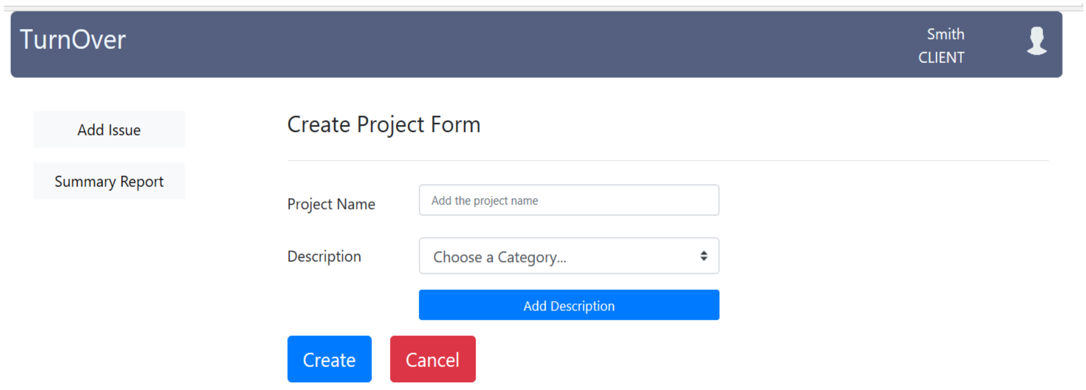


Figure 7-17 Add Project

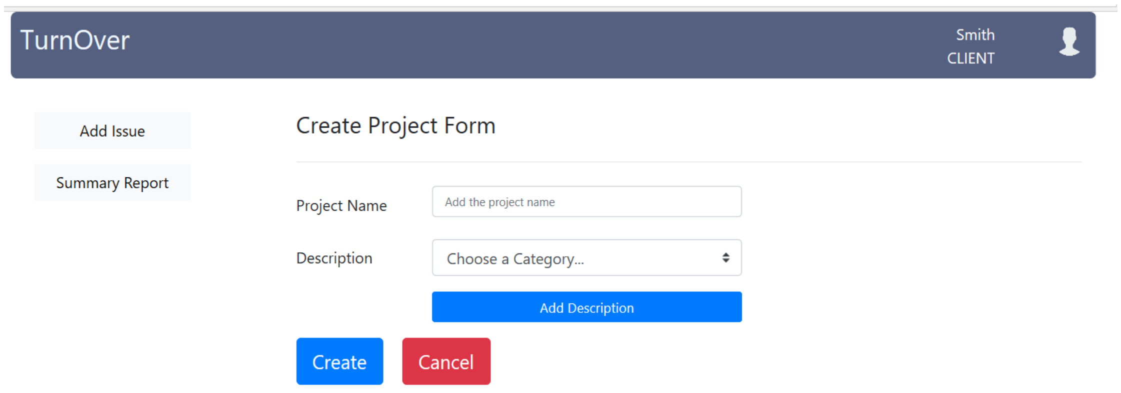


Figure 7-18 Add Module

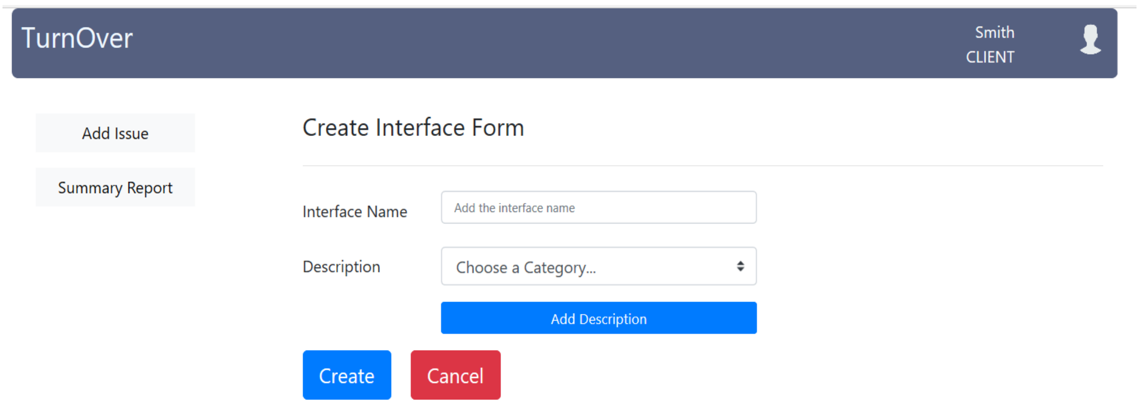
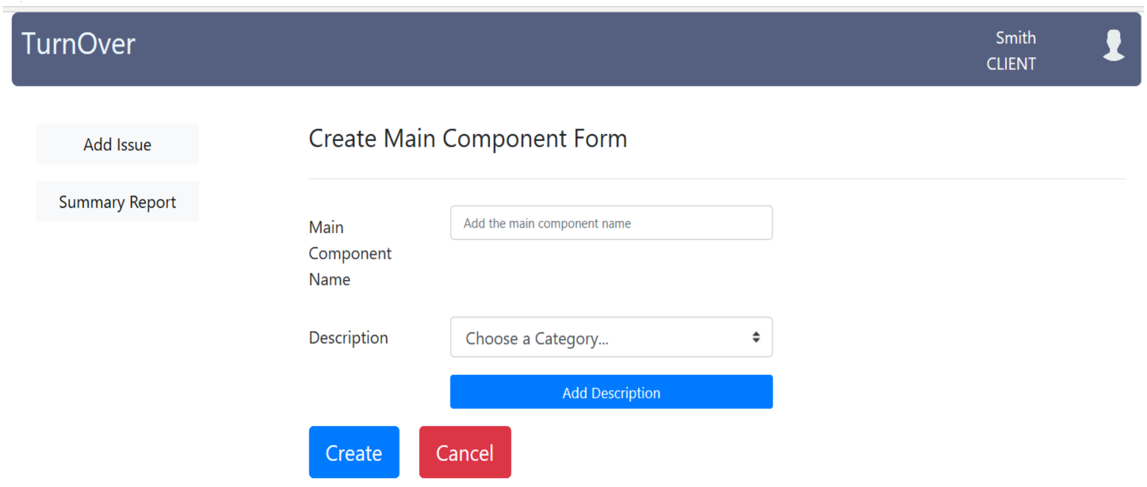


Figure 7-19 Add Interface

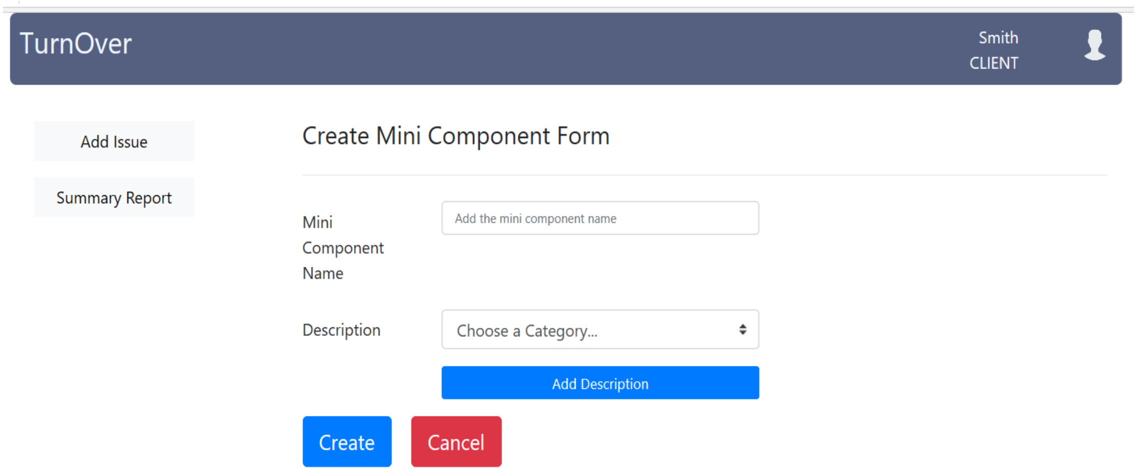


Create Main Component Form

Main Component Name: Add the main component name

Description: Choose a Category...

Figure 7-20 Add the main component

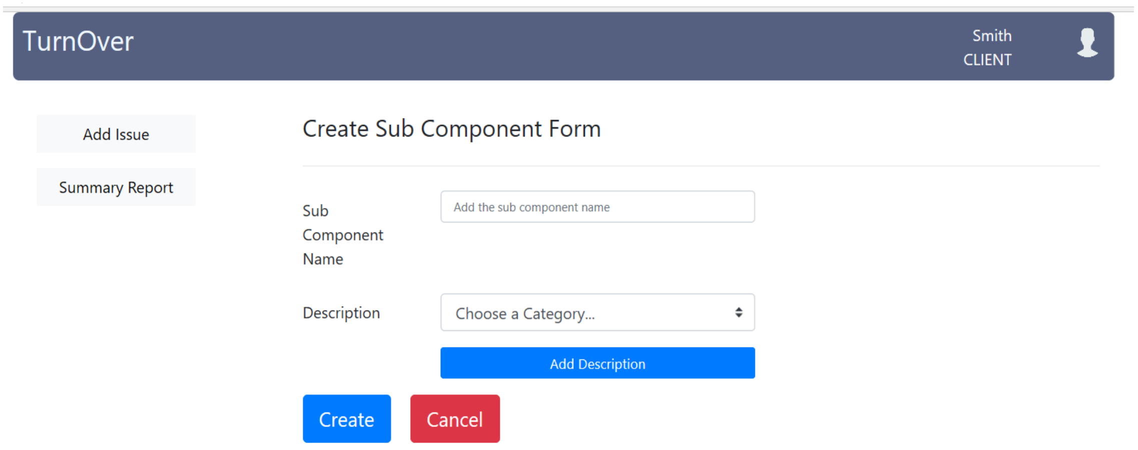


Create Mini Component Form

Mini Component Name: Add the mini component name

Description: Choose a Category...

Figure 7-21 Add the Subcomponent

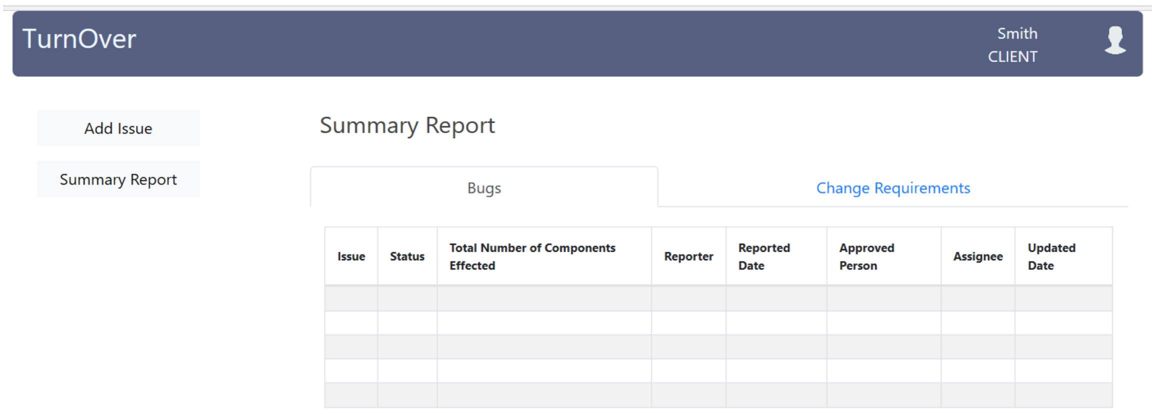


Create Sub Component Form

Sub Component Name: Add the sub component name

Description: Choose a Category...

Figure 7-22 Add the Mini component



Issue	Status	Total Number of Components Effected	Reporter	Reported Date	Approved Person	Assignee	Updated Date

Figure 7-23 Summary report UI

Business Analyst Section

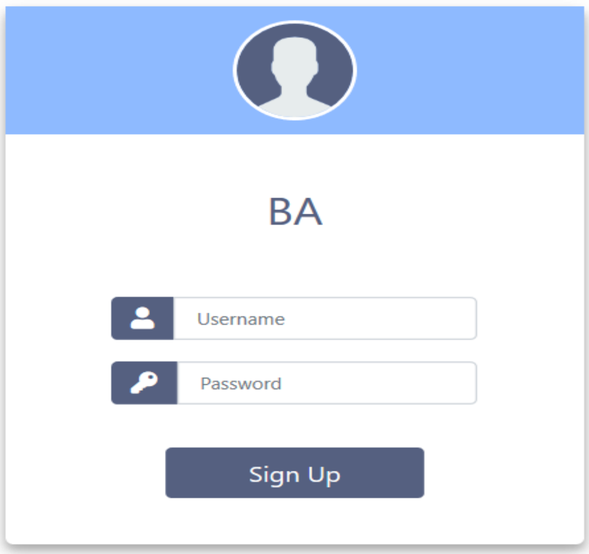


Figure 7-24 BA login page

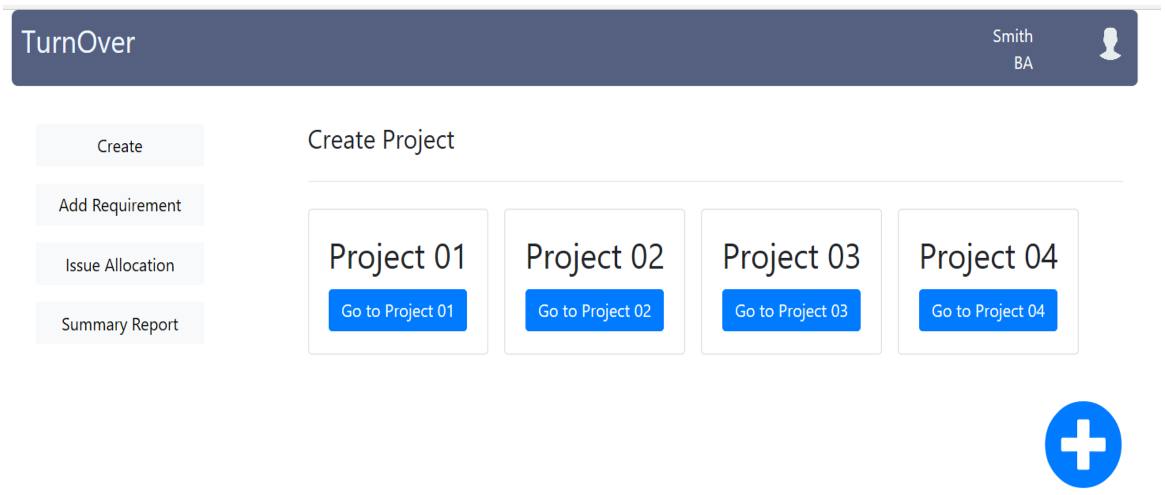


Figure 7-25 Select Project

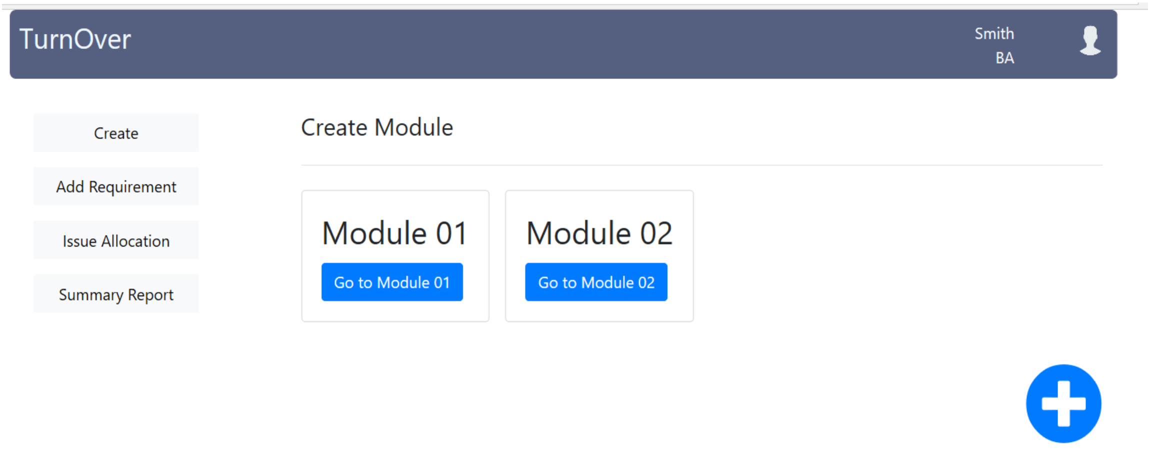


Figure 7-26 Select Module

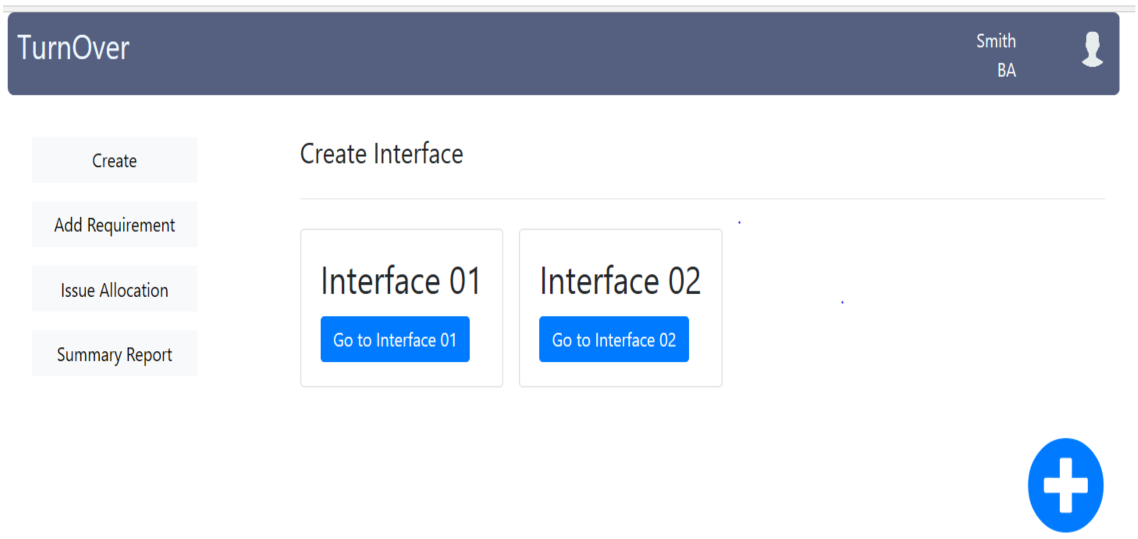


Figure 7-27 Select Interface

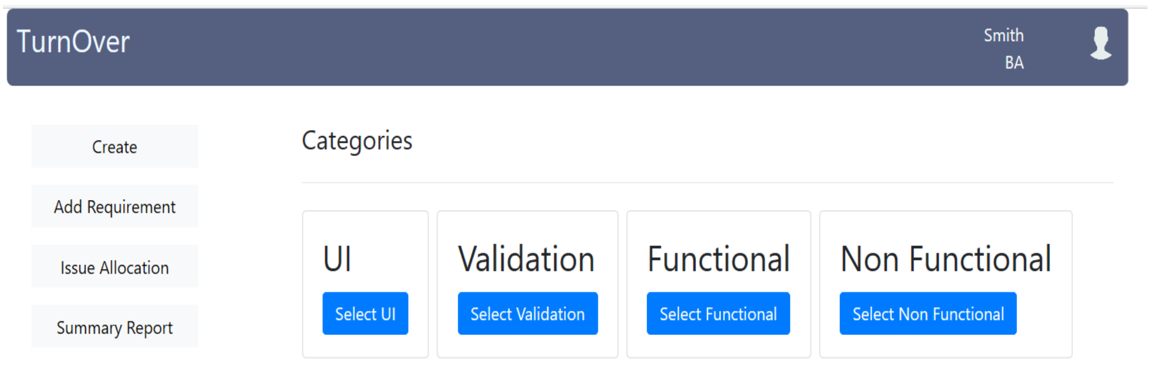


Figure 7-28 Select component type

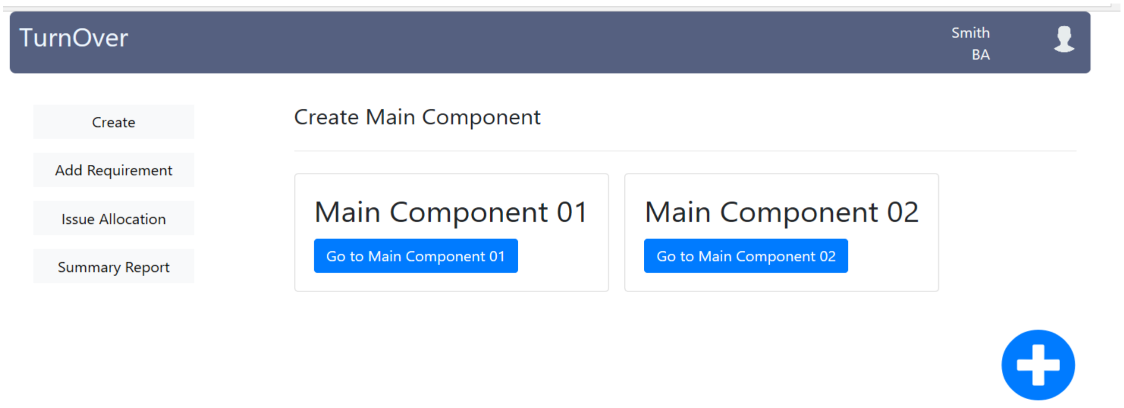


Figure 7-29 Select Main Component

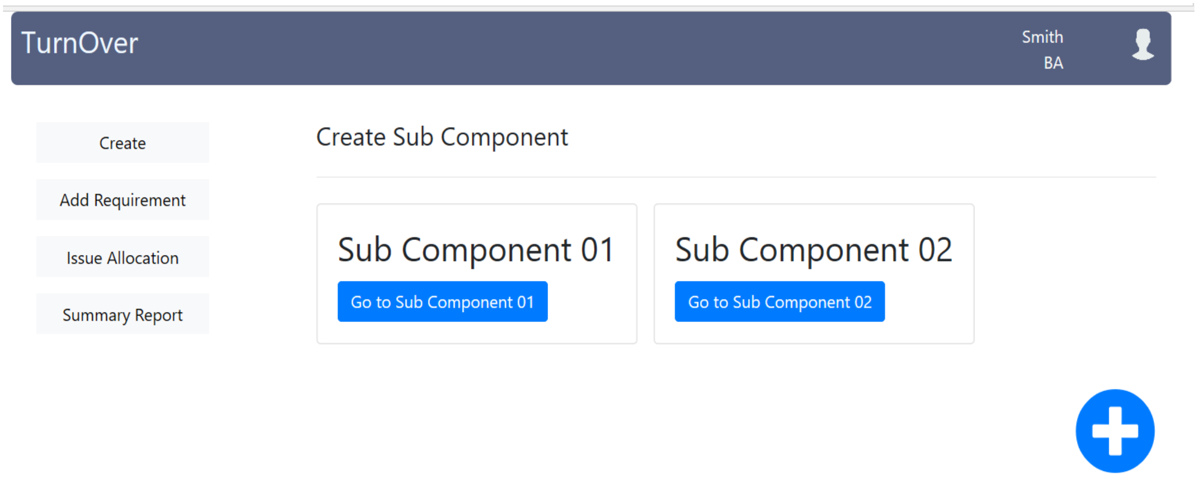


Figure 7-30 Select sub Component

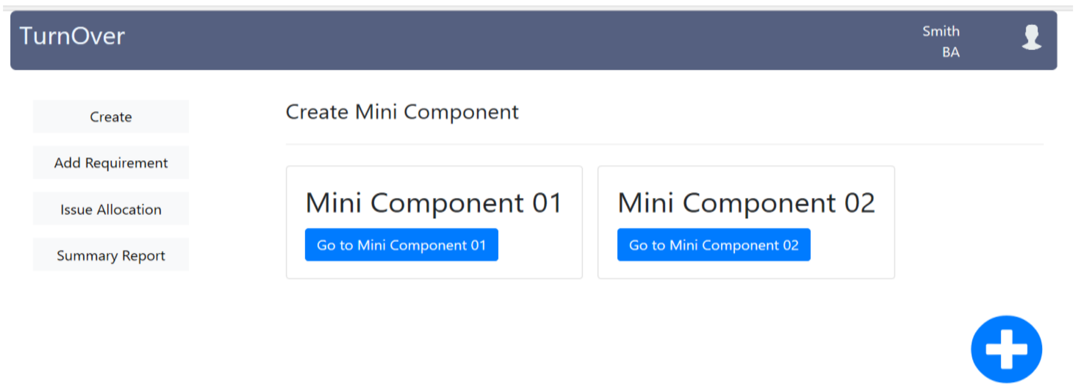


Figure 7-31 Select sub Component

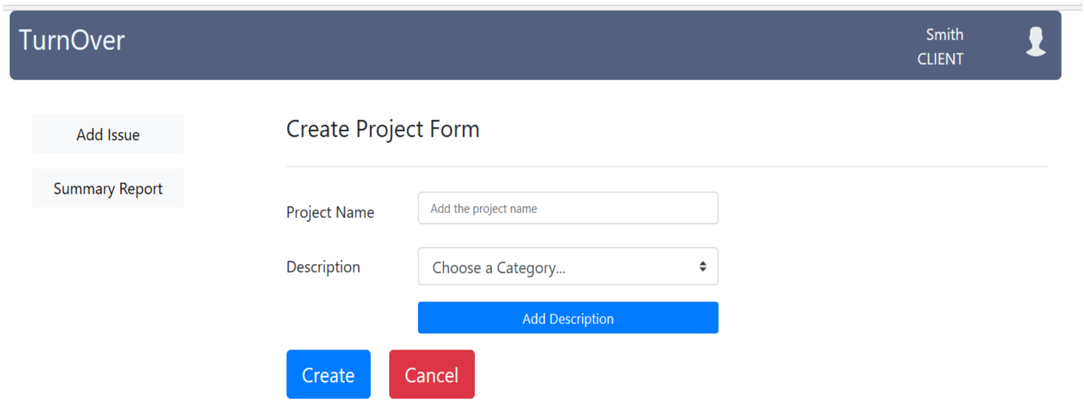


Figure 7-32 Add project

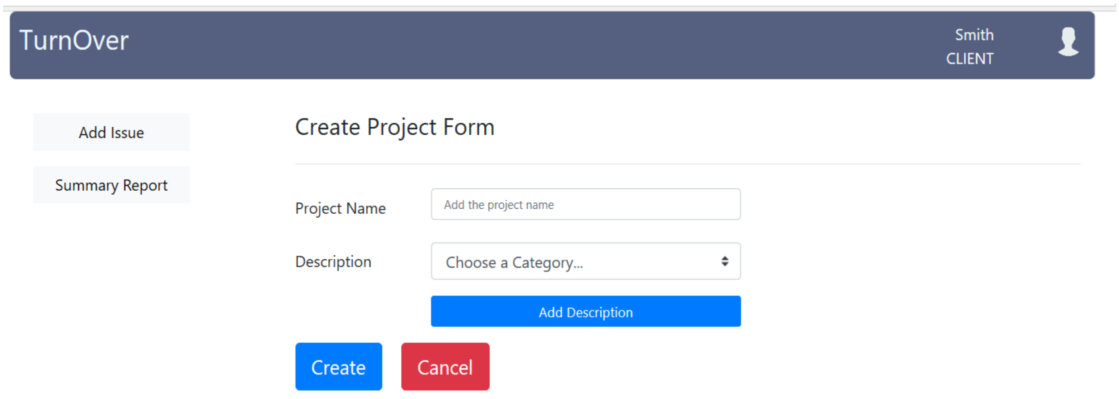


Figure 7-33 Add Module

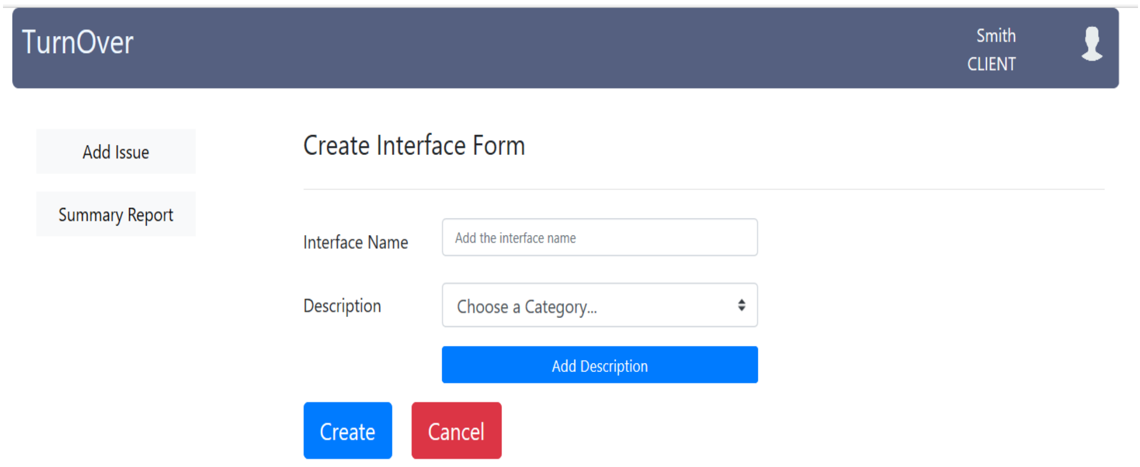
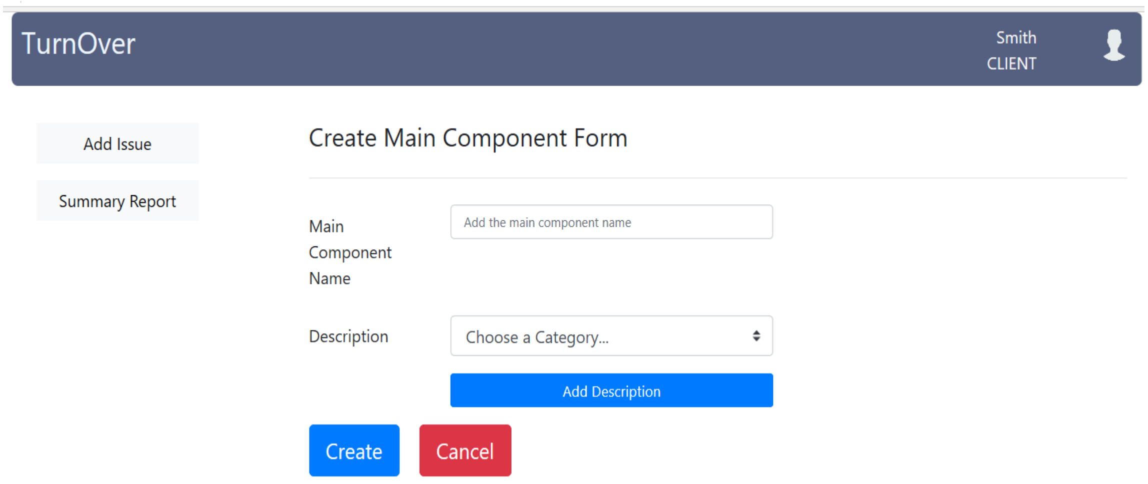


Figure 7-34 Add Interface

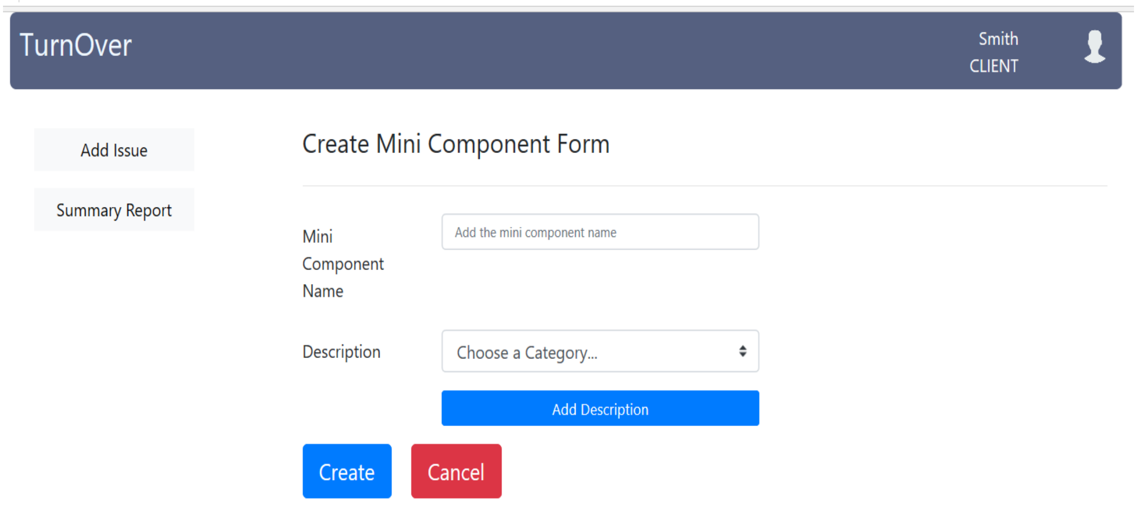

TurnOver

Smith
CLIENT

Create Main Component Form

Main Component Name	Add the main component name
Description	Choose a Category...

Figure 7-35 Add Main Component

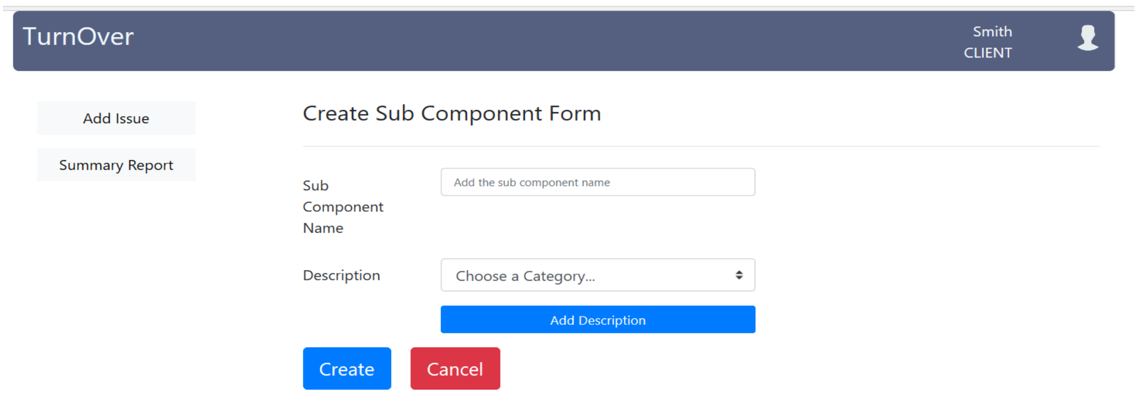

TurnOver

Smith
CLIENT

Create Mini Component Form

Mini Component Name	Add the mini component name
Description	Choose a Category...

Figure 7-36 Add sub Component

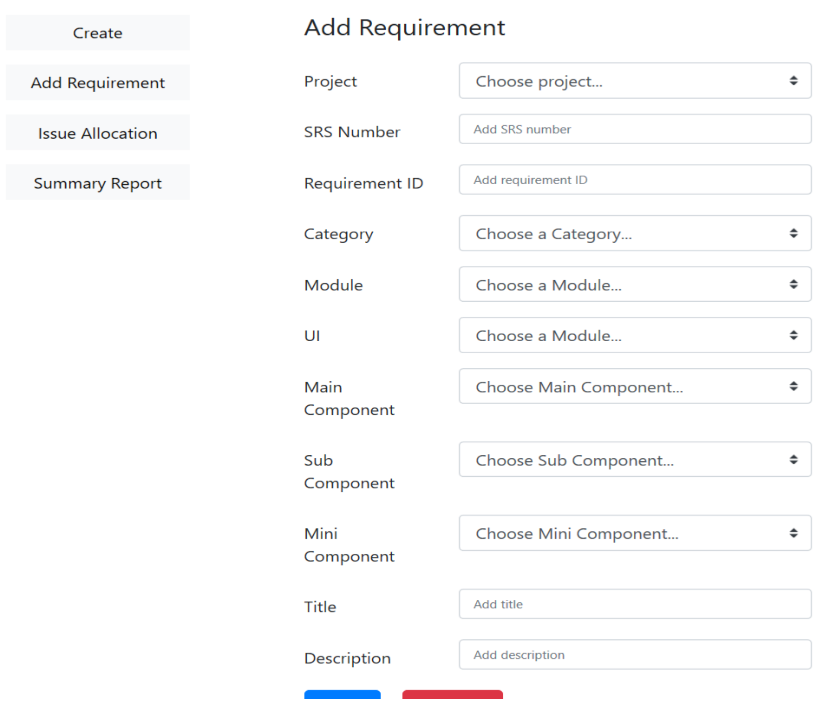

TurnOver

Smith
CLIENT

Create Sub Component Form

Sub Component Name	Add the sub component name
Description	Choose a Category...

Figure 7-37 Add mini Component

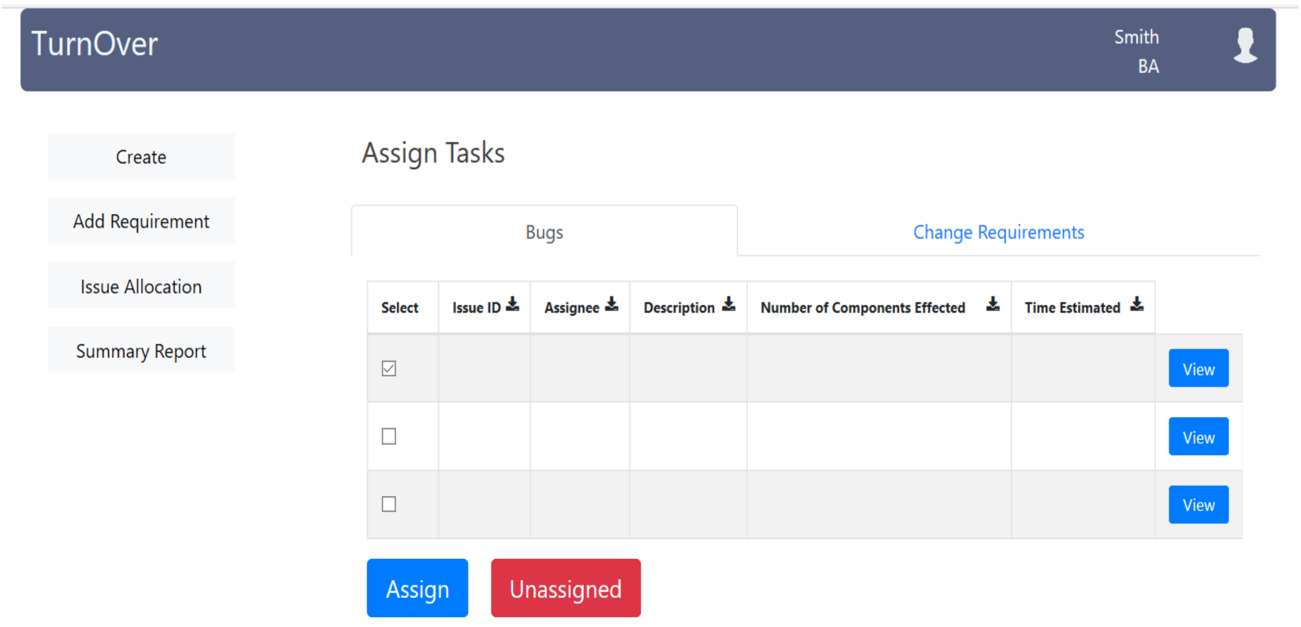


Create
Add Requirement
Issue Allocation
Summary Report

Add Requirement

Project	Choose project...
SRS Number	Add SRS number
Requirement ID	Add requirement ID
Category	Choose a Category...
Module	Choose a Module...
UI	Choose a Module...
Main Component	Choose Main Component...
Sub Component	Choose Sub Component...
Mini Component	Choose Mini Component...
Title	Add title
Description	Add description

Figure 7-38 Add Requirement



TurnOver
Smith
BA

Create
Add Requirement
Issue Allocation
Summary Report

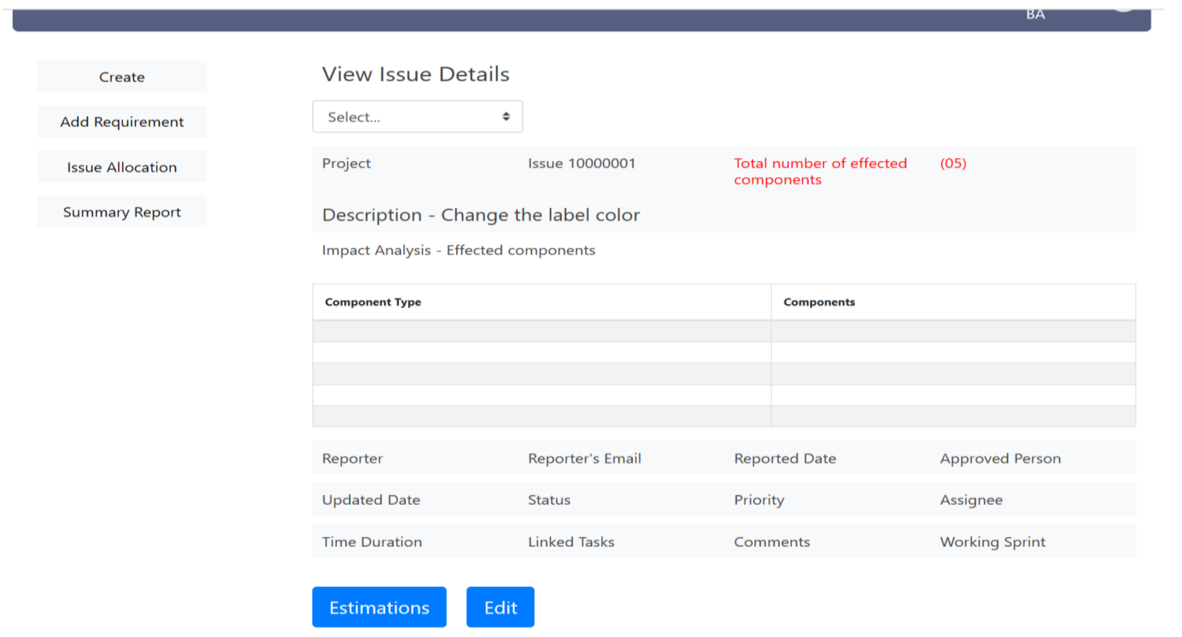
Assign Tasks

Bugs | Change Requirements

Select	Issue ID	Assignee	Description	Number of Components Effected	Time Estimated	
☑						View
☐						View
☐						View

Assign | Unassigned

Figure 7-39 Issue allocation


BA

Create
Add Requirement
Issue Allocation
Summary Report

View Issue Details

Select...

Project	Issue 10000001	Total number of effected components	(05)

Description - Change the label color

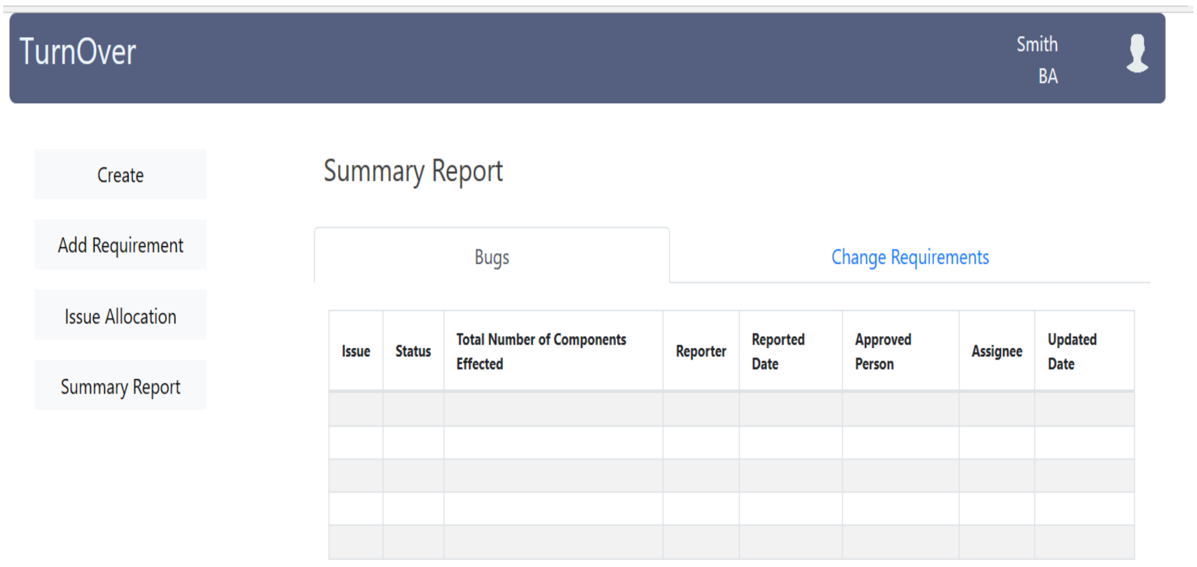
Impact Analysis - Effected components

Component Type	Components

Reporter	Reporter's Email	Reported Date	Approved Person
Updated Date	Status	Priority	Assignee
Time Duration	Linked Tasks	Comments	Working Sprint

Estimations Edit

Figure 7-40 Issue detail view


TurnOver

Smith
BA

Create
Add Requirement
Issue Allocation
Summary Report

Summary Report

Bugs | Change Requirements

Issue	Status	Total Number of Components Effected	Reporter	Reported Date	Approved Person	Assignee	Updated Date

Figure 7-23 Summary report UI

PM Section

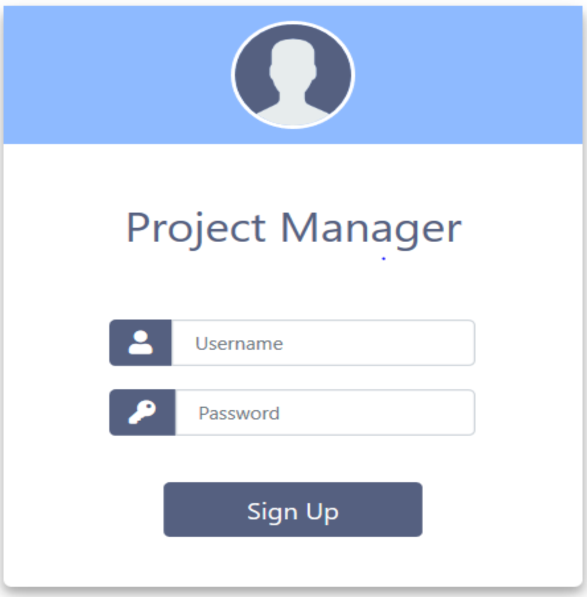
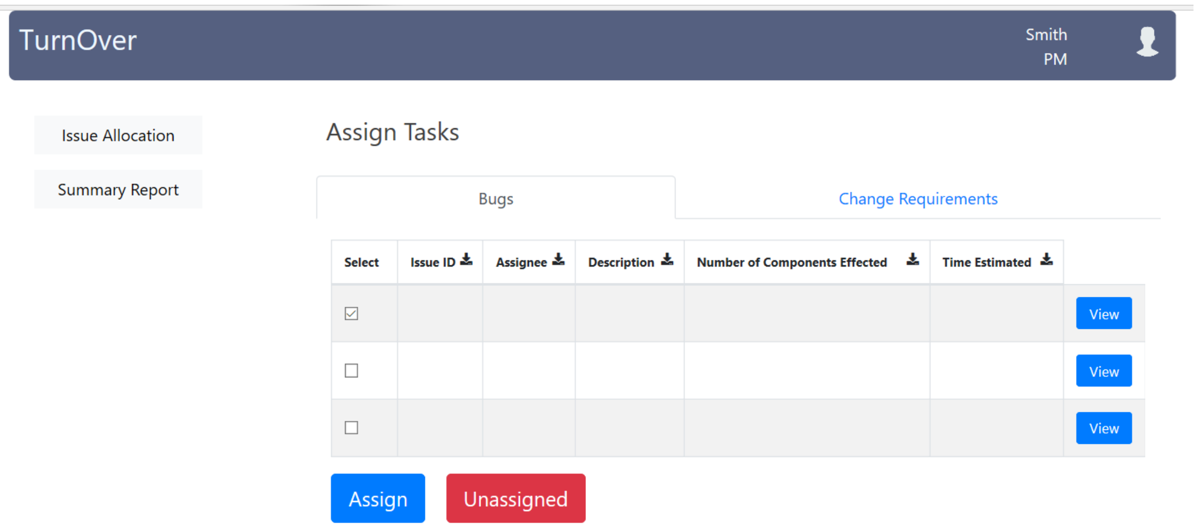


Figure 7-41 PM login page



Select	Issue ID	Assignee	Description	Number of Components Effected	Time Estimated	
☑						View
☐						View
☐						View

Figure 7-39 Issue allocation

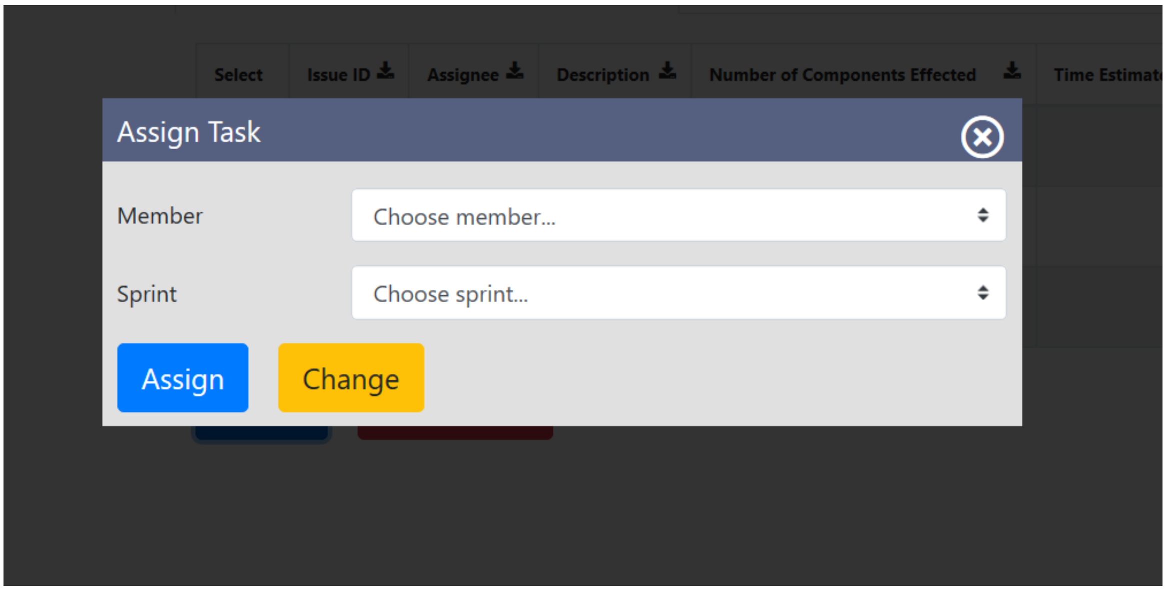
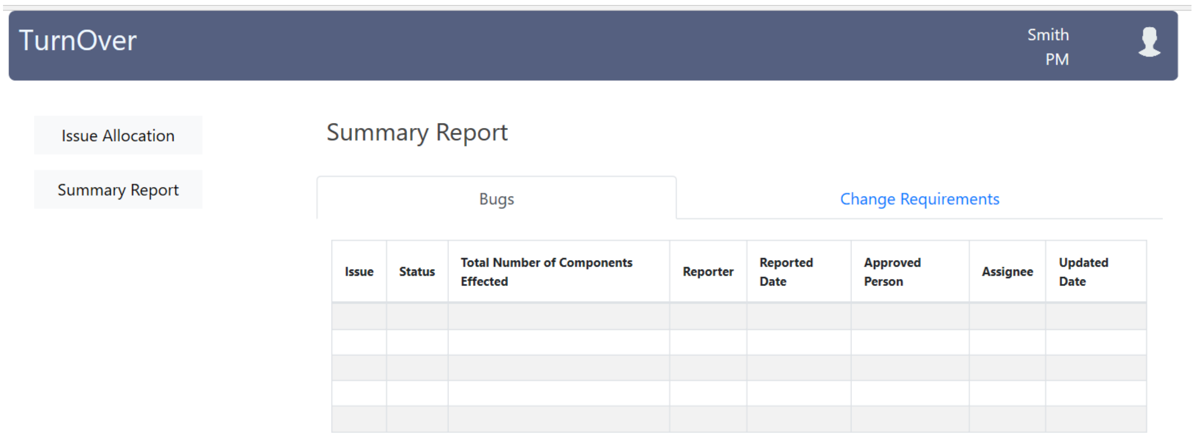
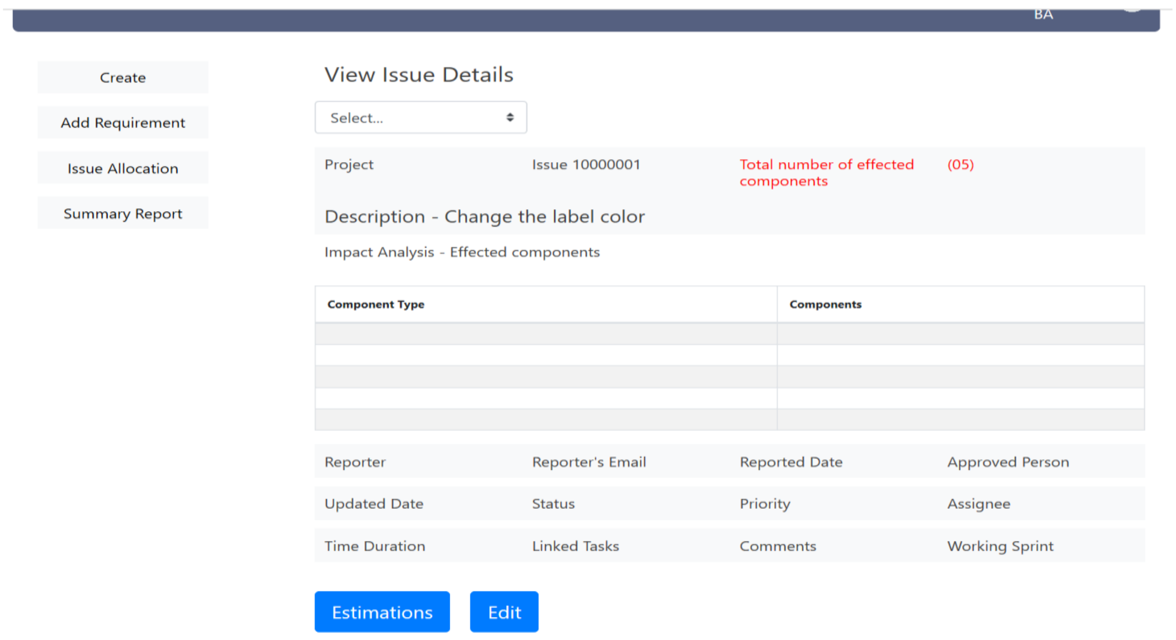


Figure 7-42 Assign task



Issue	Status	Total Number of Components Effected	Reporter	Reported Date	Approved Person	Assignee	Updated Date

Figure 7-23 Summary report UI



BA

Create

Add Requirement

Issue Allocation

Summary Report

View Issue Details

Select...

Project	Issue 10000001	Total number of effected components	(05)

Description - Change the label color

Impact Analysis - Effected components

Component Type	Components

Reporter	Reporter's Email	Reported Date	Approved Person
Updated Date	Status	Priority	Assignee
Time Duration	Linked Tasks	Comments	Working Sprint

Estimations Edit

Figure 7-40 Issue detail view

QA Section

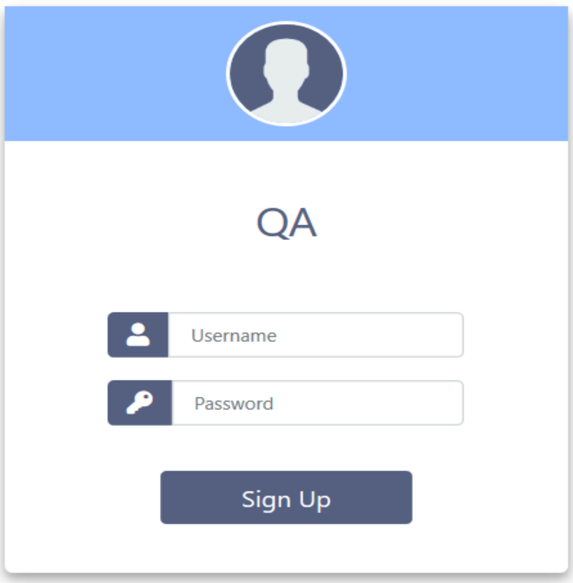
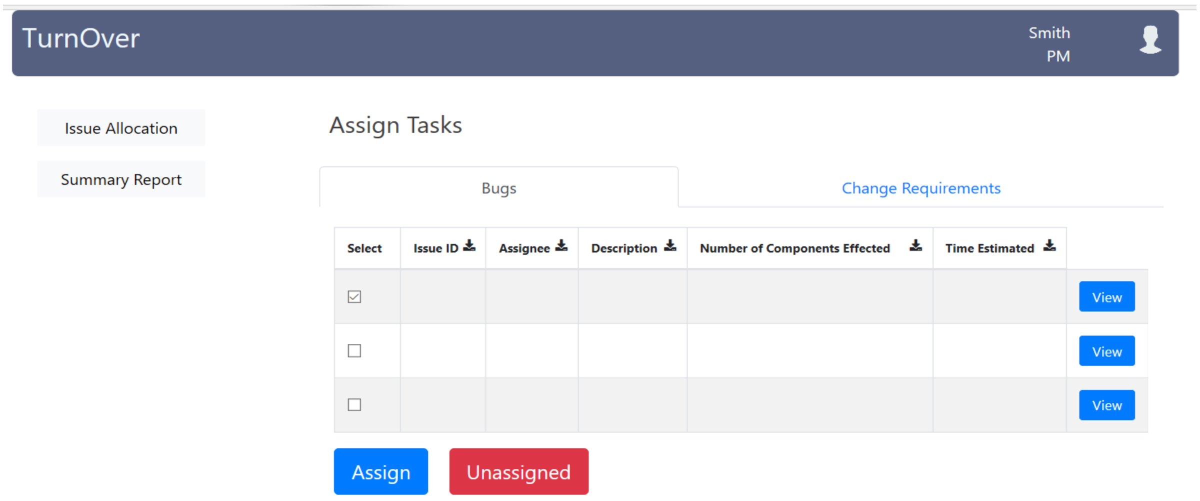


Figure 7-43 QA login



Select	Issue ID	Assignee	Description	Number of Components Effected	Time Estimated	
☑						View
☐						View
☐						View

Figure 7-39 Issue allocation

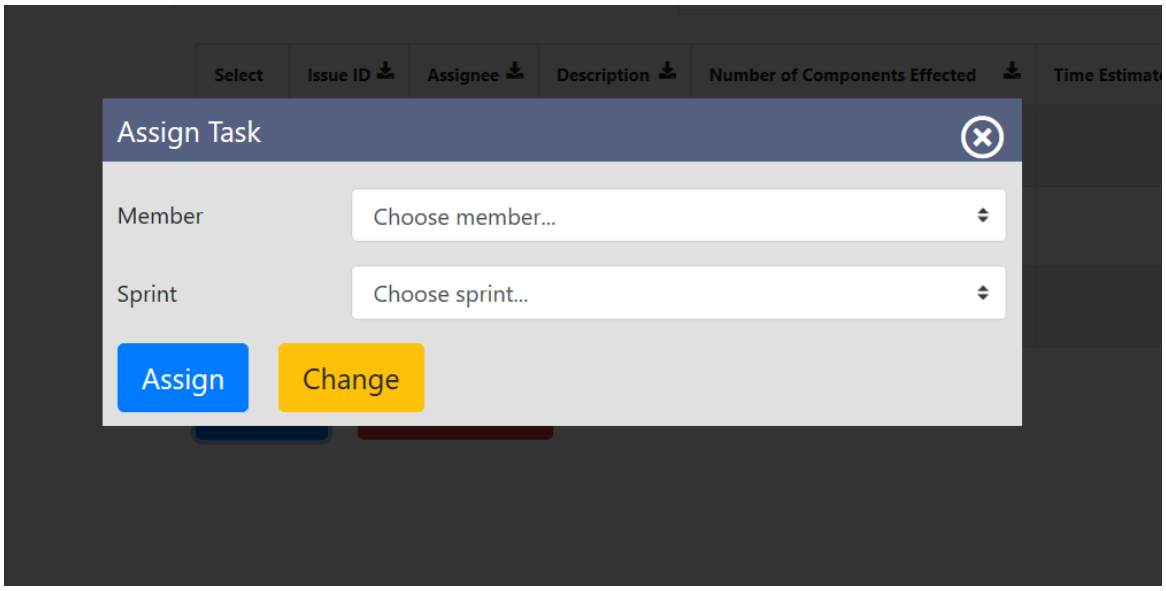
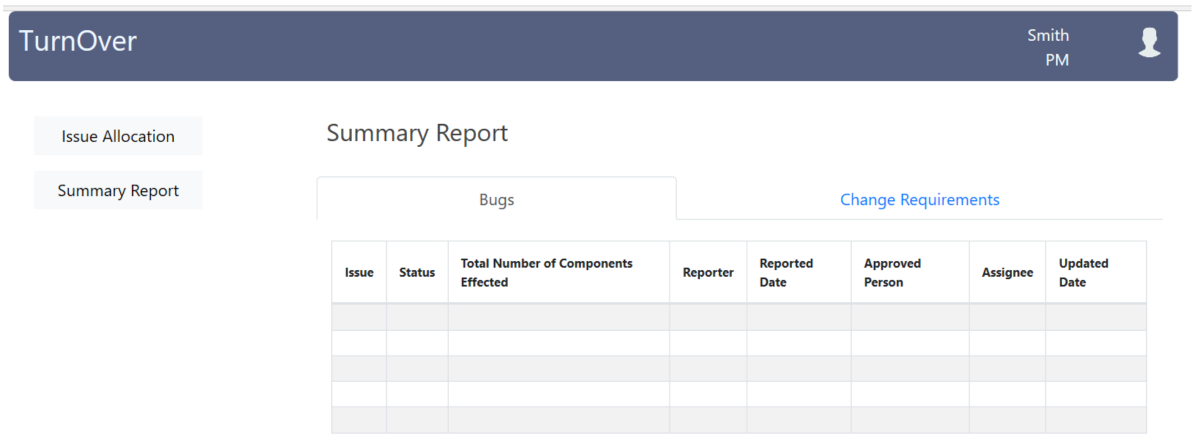
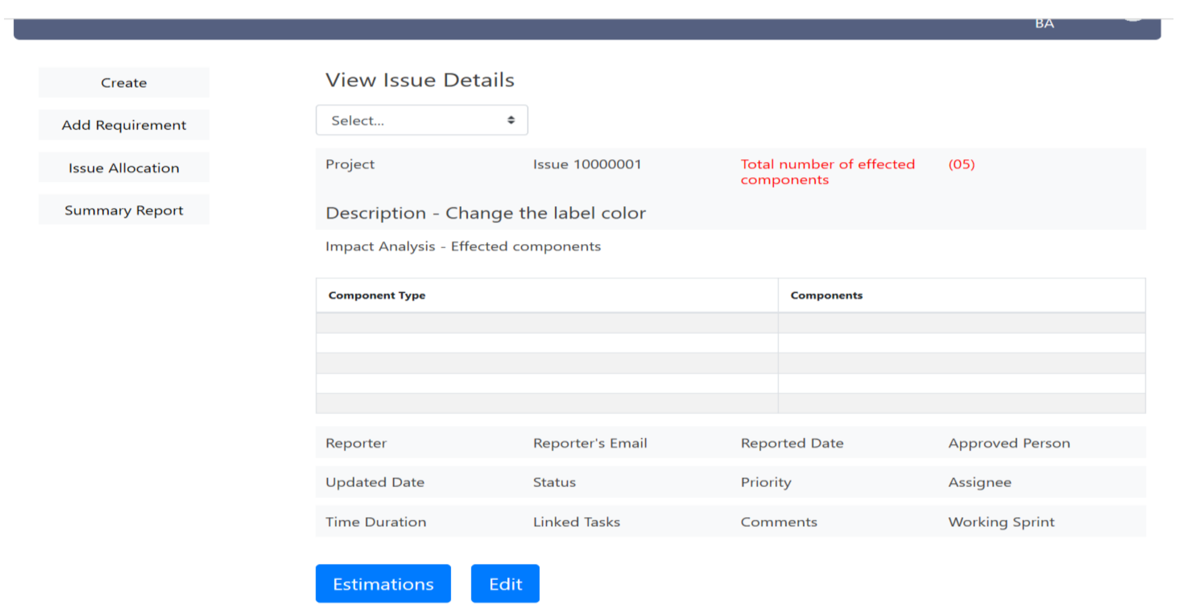


Figure 7-42 Assign task



Issue	Status	Total Number of Components Effected	Reporter	Reported Date	Approved Person	Assignee	Updated Date

Figure 7-23 Summary report UI



Project Issue 10000001 Total number of effected components (05)

Description - Change the label color

Impact Analysis - Effected components

Component Type	Components

Reporter	Reporter's Email	Reported Date	Approved Person
Updated Date	Status	Priority	Assignee
Time Duration	Linked Tasks	Comments	Working Sprint

Figure 7-40 Issue detail view

<u>Dev_Section</u>

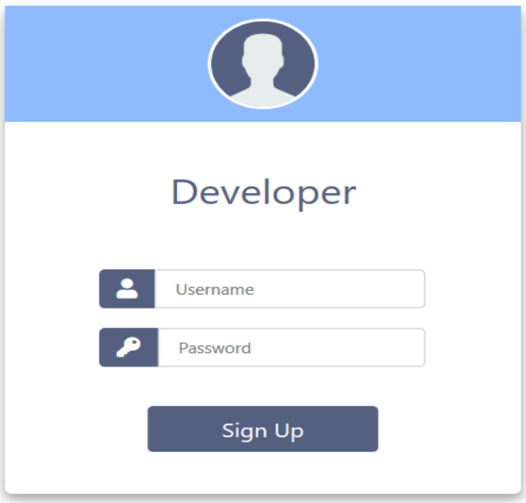
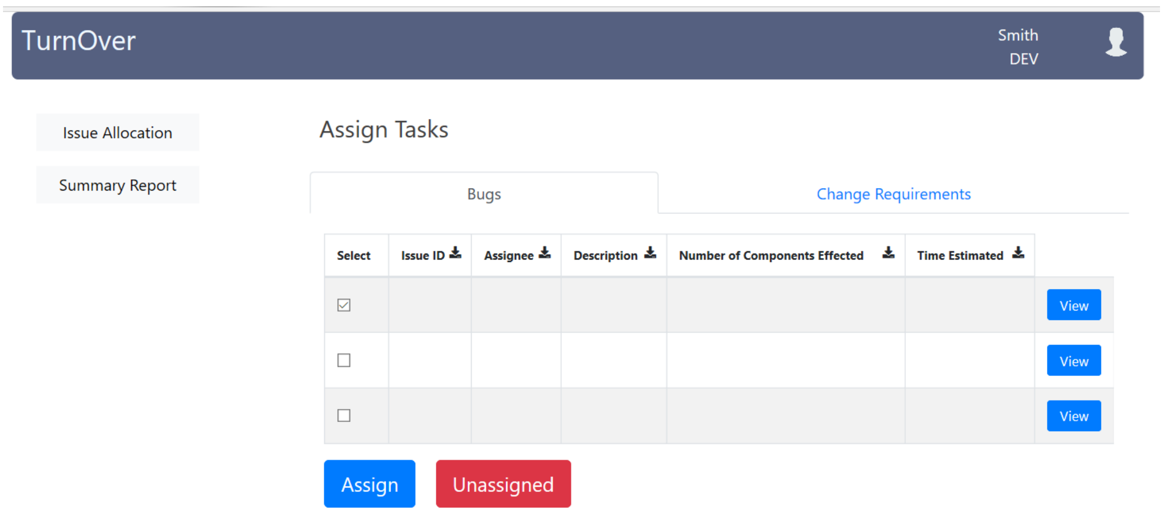


Figure 7-44 Dev login page



Select	Issue ID	Assignee	Description	Number of Components Effected	Time Estimated	
☑						View
☐						View
☐						View

Figure 7-39 Issue allocation

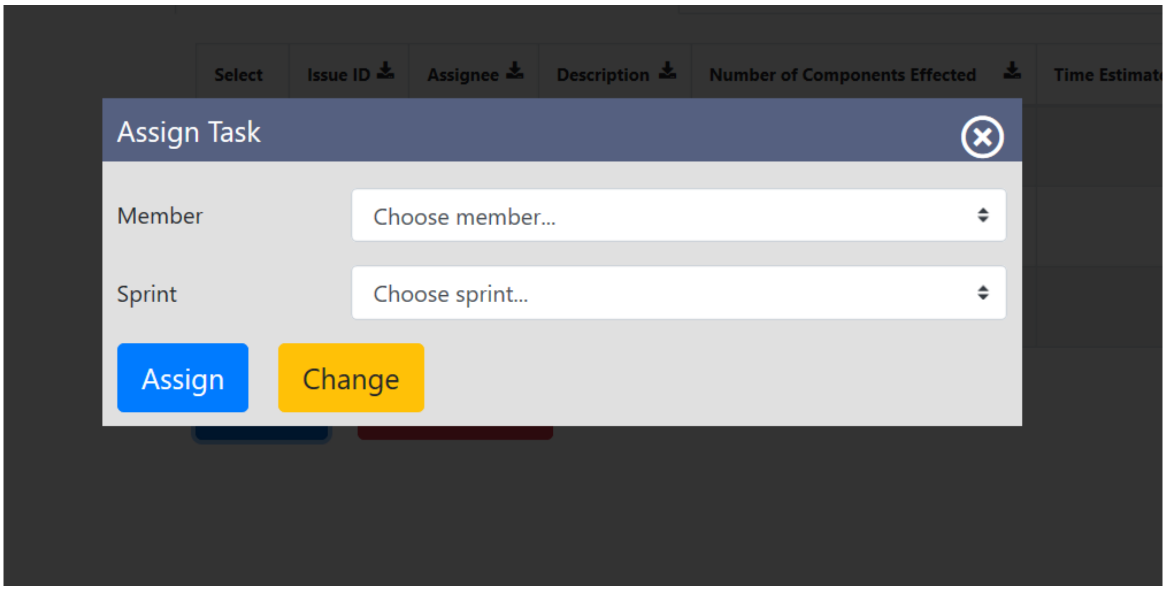
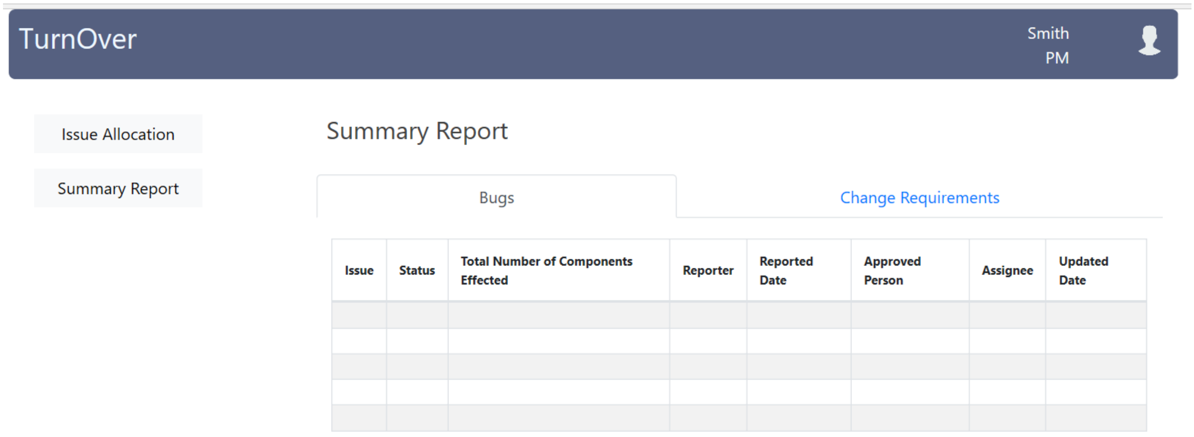


Figure 7-42 Assign task



TurnOver

Smith
PM

Issue Allocation

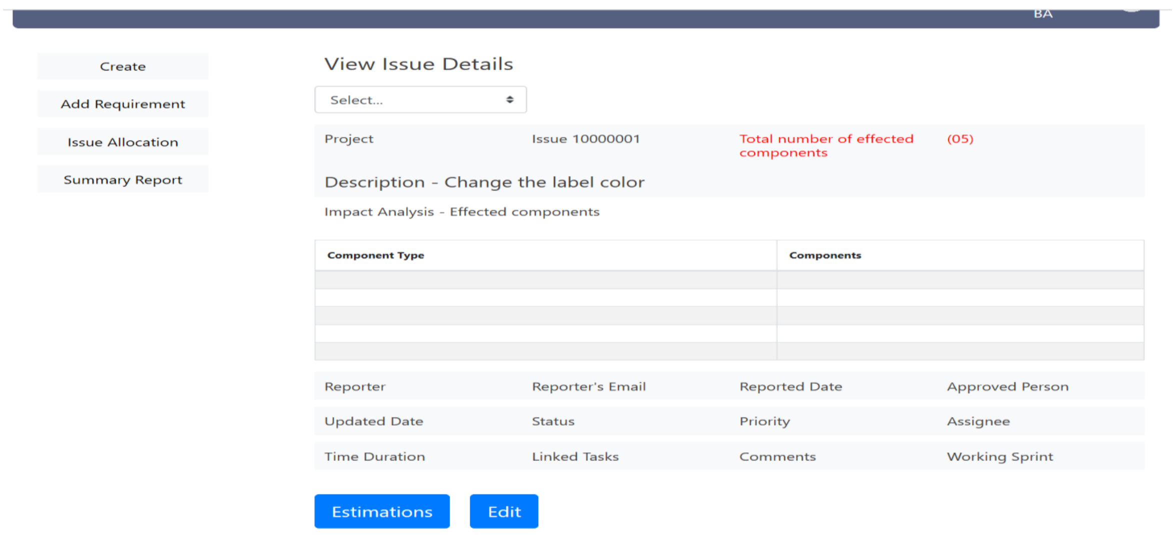
Summary Report

Summary Report

Bugs | Change Requirements

Issue	Status	Total Number of Components Effected	Reporter	Reported Date	Approved Person	Assignee	Updated Date

Figure 7-23 Summary report UI



BA

Create

Add Requirement

Issue Allocation

Summary Report

View Issue Details

Select...

Project | Issue 10000001 | Total number of effected components | (05)

Description - Change the label color

Impact Analysis - Effected components

Component Type	Components

Reporter	Reporter's Email	Reported Date	Approved Person
Updated Date	Status	Priority	Assignee
Time Duration	Linked Tasks	Comments	Working Sprint

Estimations | Edit

Figure 7-40 Issue detail view

7. 4 APPENDIX D - MAIN LOGIC RELATED SOURCE CODES

Main Class

```
package com.pm.tool;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication
public class PmtoolApplication {

        public static void main(String[] args) {
                SpringApplication.run(PmtoolApplication.class, args);
        }
```

Save Component

```
        package com.pm.tool.components;

        import com.pm.tool.entities.*;
        import com.pm.tool.models.*;
        import com.pm.tool.repositories.*;
        import com.pm.tool.services.Pm_Tool_Service;
        import org.springframework.beans.factory.annotation.Autowired;
        import org.springframework.stereotype.Component;

        import java.util.List;

        @Component
        public class SaveComponent {
            @Autowired
            ProjectRepo projectRepo;

            @Autowired
            ModuleRepo moduleRepo;

            @Autowired
            InterfaceRepo interfaceRepo;

            @Autowired
            CompoTypeRepo compoTypeRepo;
            @Autowired
            MainCompoRepo mainCompoRepo;
```

```
    @Autowired UserRepo userRepo;

    @Autowired RoleRepo roleRepo;

    @Autowired SubCompoRepo subCompoRepo;

    @Autowired MiniCompoRepo miniCompoRepo;

    @Autowired RequirementsRepo requirementsRepo;

    @Autowired ProjectIssueRepo projectIssueRepo;

    @Autowired Pm_Tool_Service pm_tool_service;

    public CommonResponse saveProject(ProjectsReqBean reqBean){
        UserRoleEntity role = new UserRoleEntity();
        role.setRole("client");

        UsersEntity usersEntity = new UsersEntity();
        usersEntity.setUserName(reqBean.getOwnedBy());
        usersEntity.setUserRoleEntity(role);

        ProjectsEntity projectsEntity = new ProjectsEntity();
        projectsEntity.setProjectName(reqBean.getProjectName());
        projectsEntity.setProjectDescription(reqBean.getProjectDescription());
        projectsEntity.setProjectStatus(true);
        projectsEntity.setUsersEntity(usersEntity);

//        roleRepo.save(role);
//        userRepo.save(usersEntity);
        Integer projectID = projectRepo.save(projectsEntity).getProjectId();
        String project = projectRepo.save(projectsEntity).getProjectName();

        return new CommonResponse(Pm_Tool_Service.STATUS.CREATED.getValue(),project +
" created.!", projectID);
    }

    public CommonResponse saveModule(ModuleReqBean moduleReqBean){

        ProjectsEntity                     project                     =
pm_tool_service.getProjectByName(moduleReqBean.getProjectName());

        ComponentStatusEntity compoStatus = new ComponentStatusEntity();
        ComponentStatusEntity newStatus = new ComponentStatusEntity();
        try {
            compoStatus = pm_tool_service.getCStatBYStat(true);
            if (compoStatus==null){
```

```
            newStatus.setStatus(true);
        }
    } catch (Exception e){
        compoStatus.setStatus(true);
    }

    ModulesEntity modulesEntity = new ModulesEntity();

    modulesEntity.setModuleName(moduleReqBean.getModuleName());
    modulesEntity.setModuleDescription(moduleReqBean.getModuleDescription());
    if (compoStatus == null){
        modulesEntity.setComponentStatusEntity(newStatus);
    }else {
        modulesEntity.setComponentStatusEntity(compoStatus);
    }
    modulesEntity.setProjectsEntity(project);

    Integer createdModuleId = moduleRepo.save(modulesEntity).getModuleId();

    return new CommonResponse(Pm_Tool_Service.STATUS.CREATED.getValue(),
moduleReqBean.getModuleName() + " created.!", createdModuleId);
  }

  public CommonResponse saveInterface(InterfacesReqBean request){
    ModulesEntity module = pm_tool_service.getModuleByName(request.getModuleName());

    ComponentStatusEntity compoStatus = new ComponentStatusEntity();
    ComponentStatusEntity newStatus = new ComponentStatusEntity();
    try {
        compoStatus = pm_tool_service.getCStatBYStat(true);
        if (compoStatus==null){
            newStatus.setStatus(true);
        }
    } catch (Exception e){
        compoStatus.setStatus(true);
    }

    InterfacesEntity newInterface = new InterfacesEntity();

    newInterface.setInterfaceName(request.getInterfaceName());
    newInterface.setDescription(request.getInterDescription());
    if (compoStatus == null){
        newInterface.setComponentStatusEntity(newStatus);
    }else {
        newInterface.setComponentStatusEntity(compoStatus);
    }
```

```
        newInterface.setModulesEntity(module);

        Integer createModId = interfaceRepo.save(newInterface).getInterfaceId();

        return new CommonResponse(Pm_Tool_Service.STATUS.CREATED.getValue(),
request.getInterfaceName()+" created.!", createModId);
    }

    public CommonResponse saveCompoType(CompoTypeReqBean request){
//        InterfacesEntity oldInterface =
pm_tool_service.getInterfaceByName(request.getInterfaceName());

        ComponentTypeEntity componentTypeEntity = new ComponentTypeEntity();

        componentTypeEntity.setTypeName(request.getTypeName());
        componentTypeEntity.setDescription(request.getTypeDescription());
//        componentTypeEntity.setInterfacesEntity(oldInterface);

        Integer createIntId = compoTypeRepo.save(componentTypeEntity).getComponentTypeId();

        return new CommonResponse(Pm_Tool_Service.STATUS.CREATED.getValue(),
request.getTypeName()+" created.!",createIntId);
    }

    public CommonResponse saveNewMainCompo(MainCompoReqBean req){
        InterfacesEntity oldInterface =
pm_tool_service.getInterfaceByName(req.getInterfaceName());

        ComponentStatusEntity compoStatus = new ComponentStatusEntity();
        ComponentStatusEntity newStatus = new ComponentStatusEntity();
        try {
            compoStatus = pm_tool_service.getCStatBYStat(true);
            if (compoStatus==null){
                newStatus.setStatus(true);
            }
        } catch (Exception e){
            compoStatus.setStatus(true);
        }

        MainComponentsEntity component = new MainComponentsEntity();

        ComponentTypeEntity type = new ComponentTypeEntity();
        type.setTypeName(req.getComponentTypeName());

        component.setComponentName(req.getMainCompotName());
        component.setDescription(req.getDescription());
        component.setInterfacesEntity(oldInterface);
        component.setComponentTypeEntity(type);

        if (compoStatus == null){
            component.setComponentStatusEntity(newStatus);
        }else {
            component.setComponentStatusEntity(compoStatus);
        }

        Integer createdCompoId = mainCompoRepo.save(component).getComponentId();

        return                                                                         new
CommonResponse(Pm_Tool_Service.STATUS.CREATED.getValue(),req.getMainCompotName
()+" created",createdCompoId);
    }

    public CommonResponse saveSubCompo(SupCompoReqBean req){
        MainComponentsEntity                    mainCompo                           =
pm_tool_service.getMainCompoByName(req.getMainCompoName());

        ComponentStatusEntity compoStatus = new ComponentStatusEntity();
        ComponentStatusEntity newStatus = new ComponentStatusEntity();
        try {
            compoStatus = pm_tool_service.getCStatBYStat(true);
            if (compoStatus==null){
                newStatus.setStatus(true);
            }
        } catch (Exception e){
            compoStatus.setStatus(true);
        }

        SubComponentsEntity subComponentsEntity = new SubComponentsEntity();

        subComponentsEntity.setComponentName(req.getSubComponentName());
        subComponentsEntity.setDescription(req.getSubCompoDes());
        subComponentsEntity.setMainComponentsEntity(mainCompo);

        if (compoStatus == null){
            subComponentsEntity.setComponentStatusEntity(newStatus);
        }else {
            subComponentsEntity.setComponentStatusEntity(compoStatus);
        }

        Integer createdId = subCompoRepo.save(subComponentsEntity).getSubComponentId();

        return     new     CommonResponse(Pm_Tool_Service.STATUS.CREATED.getValue(),
req.getSubComponentName()+ " created.!",createdId);
    }
    public CommonResponse saveMiniCompo(MiniCompoReqBean req){
        SubComponentsEntity subCompo = pm_tool_service.getSubComByName(req.getSubComponentName());

        ComponentStatusEntity compoStatus = new ComponentStatusEntity();
        ComponentStatusEntity newStatus = new ComponentStatusEntity();
        try {
            compoStatus = pm_tool_service.getCStatBYStat(true);
            if (compoStatus==null){
                newStatus.setStatus(true);
            }
        } catch (Exception e){
            compoStatus.setStatus(true);
        }

        MiniComponentsEntity miniCompo = new MiniComponentsEntity();

        miniCompo.setMiniCompoName(req.getMiniCompoName());
        miniCompo.setDescription(req.getMiniDescription());
        miniCompo.setSubComponentsEntity(subCompo);
        if (compoStatus == null){
            miniCompo.setComponentStatusEntity(newStatus);
        }else {
            miniCompo.setComponentStatusEntity(compoStatus);
        }

        Integer createId = miniCompoRepo.save(miniCompo).getMiniCompoId();

        return new CommonResponse(Pm_Tool_Service.STATUS.CREATED.getValue(),req.getMiniCompoName()+" created.!",createId);
    }

    public CommonResponse saveRequirement(RequiremnetsReqBean req){
        ProjectsEntity project = pm_tool_service.getProjectByName(req.getProjectName());
        ModulesEntity module = pm_tool_service.getModuleByName(req.getModuleName());
        InterfacesEntity interfaceSelected = pm_tool_service.getInterfaceByName(req.getInterName());
        ComponentTypeEntity category = pm_tool_service.getTypeByName(req.getCompType());
        MainComponentsEntity mainComp = pm_tool_service.getMainCompoByName(req.getMainComName());
        SubComponentsEntity subComp = pm_tool_service.getSubComByName(req.getSubCompName());
        MiniComponentsEntity miniComp = pm_tool_service.getMiniByName(req.getMiniCompName());

        RequiremnetsEntity requiremnetsEntity = new RequiremnetsEntity();
        requiremnetsEntity.setRequestTitle(req.getTitle());
        requiremnetsEntity.setDetailedDescription(req.getDescription());
        requiremnetsEntity.setProjectsEntity(project);
        requiremnetsEntity.setModulesEntity(module);
        requiremnetsEntity.setInterfacesEntity(interfaceSelected);
        requiremnetsEntity.setComponentTypeEntity(category);
        requiremnetsEntity.setMainComponentsEntity(mainComp);
        requiremnetsEntity.setSubComponentsEntity(subComp);
        requiremnetsEntity.setMiniComponentsEntity(miniComp);

        Integer createdId = requirementsRepo.save(requiremnetsEntity).getReqId();

        return new
CommonResponse(Pm_Tool_Service.STATUS.CREATED.getValue(),req.getTitle()+"
created.!",createdId);
    }

    public CommonResponse saveProjectIssue(RequiremnetsReqBean req){
        ProjectsEntity project = pm_tool_service.getProjectByName(req.getProjectName());
        ModulesEntity module = pm_tool_service.getModuleByName(req.getModuleName());
        InterfacesEntity interfaceSelected =
pm_tool_service.getInterfaceByName(req.getInterName());
        ComponentTypeEntity category = pm_tool_service.getTypeByName(req.getCompType());
        MainComponentsEntity mainComp =
pm_tool_service.getMainCompoByName(req.getMainComName());
        SubComponentsEntity subComp =
pm_tool_service.getSubComByName(req.getSubCompName());
        MiniComponentsEntity miniComp =
pm_tool_service.getMiniByName(req.getMiniCompName());

        TitleReqBean titleReq = new TitleReqBean();

        titleReq.setProjectName(req.getProjectName());
        titleReq.setModuleName(req.getModuleName());
        titleReq.setInterName(req.getInterName());
        titleReq.setCompType(req.getCompType());
        titleReq.setMainComName(req.getMainComName());
        titleReq.setSubCompName(req.getSubCompName());
        titleReq.setMiniCompName(req.getMiniCompName());
        titleReq.setTitle(req.getTitle());

        IssueTypesEntity type = new IssueTypesEntity();
        IssueTypesEntity newType = new IssueTypesEntity();

        if (pm_tool_service.getIssueType(titleReq)=="Bug"){
            try {
                /*get existing bug entity from issue type table*/
                type= pm_tool_service.getPIsuTypeByType("Bug");
            if (type==null){
                /*create a bug field if there is not any bug in issue type table*/
                newType.setIssueType("Bug");
            }
        }catch (Exception e) {
            type.setIssueType("Bug");
        }
    }else {
        try {
            /*get existing cr entity from issue type table*/
            type= pm_tool_service.getPIsuTypeByType("Change Request");
            if (type==null){
                /*create a cr field if there is not any cr in issue type table*/
                newType.setIssueType("Change Request");
            }
        }catch (Exception e) {
            type.setIssueType("Change Request");
        }
    }

    ProjectIssuesEntity issue = new ProjectIssuesEntity();
    issue.setIssueTitle(req.getTitle());
    issue.setIssueDescription(req.getDescription());
    issue.setProjectsEntity(project);
    issue.setModulesEntity(module);
    issue.setInterfacesEntity(interfaceSelected);
    issue.setComponentTypeEntity(category);
    issue.setMainComponentsEntity(mainComp);
    issue.setSubComponentsEntity(subComp);
    issue.setMiniComponentsEntity(miniComp);
    if (type == null){
        issue.setIssueTypesEntity(newType);
    }else {
        issue.setIssueTypesEntity(type);
    }

    String issueType = issue.getIssueTypesEntity().getIssueType();

    Integer createdId = projectIssueRepo.save(issue).getIssueId();

    return                                                                      new
CommonResponse(Pm_Tool_Service.STATUS.CREATED.getValue(),issueType              +"
created.!",createdId);
  }
```

Retrieve Component

```
package com.pm.tool.components;

import com.pm.tool.entities.*;
import com.pm.tool.models.RequiremnetsReqBean;
import com.pm.tool.models.TitleReqBean;
import com.pm.tool.repositories.*;
import com.pm.tool.services.Pm_Tool_Service;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.data.domain.Example;
import org.springframework.data.domain.Pageable;
import org.springframework.stereotype.Component;

import java.util.*;

@Component
public class RetriveComponents {
    @Autowired
    ProjectRepo projectRepo;

    @Autowired
    ModuleRepo moduleRepo;

    @Autowired
    InterfaceRepo interfaceRepo;

    @Autowired
    CompoTypeRepo compoTypeRepo;

    @Autowired
    MainCompoRepo mainCompoRepo;

    @Autowired SubCompoRepo subCompoRepo;

    @Autowired MiniCompoRepo miniCompoRepo;

    @Autowired RequirementsRepo requirementsRepo;

    @Autowired ProjectIssueRepo projectIssueRepo;

    @Autowired CompoStatusRepo compoStatusRepo;

    @Autowired IssueTypeRepo issueTypeRepo;

    @Autowired
```

```
Pm_Tool_Service pm_tool_service;
public Optional<ProjectsEntity> getProjectById(Integer id){
    Optional<ProjectsEntity> res;
    res = projectRepo.findById(id);
    return res;
}

public List<ProjectsEntity> getAllProjects(){
    Iterator<ProjectsEntity> iterator = projectRepo.findAll().iterator();
    List<ProjectsEntity> projectList = new ArrayList<ProjectsEntity>();
    while ((iterator.hasNext())){
        projectList.add(iterator.next());
    }
    return projectList;
}
public ProjectsEntity getProjectByName(String projectName){
    ProjectsEntity project = projectRepo.getProjectByName(projectName);
    return project;
}

public List<ModulesEntity> getModuleByProject(String req){
    ProjectsEntity project = pm_tool_service.getProjectByName(req);
    Integer projectId = project.getProjectId();
    List<ModulesEntity> res = moduleRepo.getModuleByProject(projectId);
    return res;
}

public ComponentTypeEntity getTypeByName(String typeName){
    return compoTypeRepo.getTypeByName(typeName);
}

public ModulesEntity getModuleByName(String moduleName){
    return moduleRepo.getModuleByName(moduleName);
}

public MainComponentsEntity getMainCompoByName(String compoName){
    return mainCompoRepo.getMainCompoByName(compoName);
}

public SubComponentsEntity getSubCompoByName(String subCompoName){
    return subCompoRepo.getSubCompoByName(subCompoName);
}

public MiniComponentsEntity getMiniByName(String miniName){
    return miniCompoRepo.getMiniByName(miniName);
}
```

```
public InterfacesEntity getInterfaceByName(String interfaceName){
    return interfaceRepo.getInterfaceByName(interfaceName);
}

public ComponentStatusEntity getComStatByNewSta(Boolean status){
    return compoStatusRepo.getCompoStatusByOldStatus(status);
}

public IssueTypesEntity getIsuTypeByType(String type){
    return issueTypeRepo.getPIssueTypeByType(type);
}

public List<ModulesEntity> getAllModules(){
    Iterator<ModulesEntity> iterator = moduleRepo.findAll().iterator();
    List<ModulesEntity> moduleList = new ArrayList<ModulesEntity>();
    while (iterator.hasNext()){
        moduleList.add(iterator.next());
    }
    return moduleList;
}

public List<InterfacesEntity> getAllInterfaces(){
    Iterator<InterfacesEntity> iterator = interfaceRepo.findAll().iterator();
    List<InterfacesEntity> interfaceList = new ArrayList<InterfacesEntity>();
    while (iterator.hasNext()){
        interfaceList.add(iterator.next());
    }
    return interfaceList;
}

public List<ComponentTypeEntity> getAllCompoTypes(){
    Iterator<ComponentTypeEntity> iterator = compoTypeRepo.findAll().iterator();
    List<ComponentTypeEntity> typeList = new ArrayList<ComponentTypeEntity>();
    while (iterator.hasNext()){
        typeList.add(iterator.next());
    }
    return typeList;
}

public List<MainComponentsEntity> getAllMainCompo(){
    Iterator<MainComponentsEntity> iterator = mainCompoRepo.findAll().iterator();
    List<MainComponentsEntity> compoList = new ArrayList<MainComponentsEntity>();
    while (iterator.hasNext()){
        compoList.add(iterator.next());
    }
    return compoList;
}

public List<SubComponentsEntity> getAllSubCompo(){
    Iterator<SubComponentsEntity> iterator = subCompoRepo.findAll().iterator();
    List<SubComponentsEntity> subCompoList = new ArrayList<SubComponentsEntity>();
    while (iterator.hasNext()){
        subCompoList.add(iterator.next());
    }
    return subCompoList;
}

public List<MiniComponentsEntity> getAllMiniCompo(){
    Iterator<MiniComponentsEntity> iterator = miniCompoRepo.findAll().iterator();
    List<MiniComponentsEntity> miniCompoList = new ArrayList<MiniComponentsEntity>();
    while (iterator.hasNext()){
        miniCompoList.add(iterator.next());
    }
    return miniCompoList;
}

public List<InterfacesEntity> getIntByModName(String moduleName){
    ModulesEntity module = pm_tool_service.getModuleByName(moduleName);
    Integer modId = module.getModuleId();
    return interfaceRepo.getIntByModName(modId);
}

public List<String> getComTyByInterName(String interName){
    return compoTypeRepo.getTypeByInterName(interName);
}

public List<String> getMainCompNames(String interName){
    return mainCompoRepo.getComByInterName(interName);
}

public List<String> getSubByMain(String mainComName){
    return subCompoRepo.getSubByMainName(mainComName);
}

public List<String> getMiniBySub(String subName){
    return miniCompoRepo.getMiniBySub(subName);
}

public HashMap<String, Integer> findReqId(TitleReqBean req){

    HashMap<String, Integer> reqIdMap = new HashMap<>();
```

```
        Integer projectId=null, moduleId=null, interfaceId=null, typeId=null, mainId=null, subId=null, miniId=null;


        try {
            projectId = pm_tool_service.getProjectByName(req.getProjectName()).getProjectId();
            moduleId = pm_tool_service.getModuleByName(req.getModuleName()).getModuleId();
            interfaceId = pm_tool_service.getInterfaceByName(req.getInterName()).getInterfaceId();
            typeId = pm_tool_service.getTypeByName(req.getCompType()).getComponentTypeId();
            mainId = pm_tool_service.getModuleByName(req.getModuleName()).getModuleId();
            subId = pm_tool_service.getSubComByName(req.getSubCompName()).getSubComponentId();
            miniId = pm_tool_service.getMiniByName(req.getMiniCompName()).getMiniCompoId();
        }catch (Exception e){
            System.out.println("some ids can't find in db");
        }

        reqIdMap.put("projectId",projectId);
        reqIdMap.put("moduleId",moduleId);
        reqIdMap.put("interfaceId",interfaceId);
        reqIdMap.put("typeId",typeId);
        reqIdMap.put("mainId",mainId);
        reqIdMap.put("subId",subId);
        reqIdMap.put("miniId",miniId);

        return reqIdMap;
    }

    public List<String> getReqPITitle(TitleReqBean req){
        List<String> titleList = new ArrayList<>();

        HashMap<String, Integer> reqIds = findReqId(req);

        /*prepare a list of title using current parameters which client has entered for show a drop down current title list*/
        if (reqIds.get("miniId")==null){

titleList.addAll(projectIssueRepo.getTitleWithoutMini(reqIds.get("projectId"),reqIds.get("moduleId"),reqIds.get("typeId"),reqIds.get("interfaceId"),reqIds.get("mainId"),reqIds.get("subId")));

titleList.addAll(requirementsRepo.getTitleWithoutMini(reqIds.get("projectId"),reqIds.get("moduleId"),reqIds.get("typeId"),reqIds.get("interfaceId"),reqIds.get("mainId"),reqIds.get("subId")));
        }else if (reqIds.get("subId")==null){

titleList.addAll(projectIssueRepo.getTitleWithoutSub(reqIds.get("projectId"),reqIds.get("moduleId"),reqIds.get("typeId"),reqIds.get("interfaceId"),reqIds.get("mainId")));

titleList.addAll(requirementsRepo.getTitleWithoutSub(reqIds.get("projectId"),reqIds.get("module
Id"),reqIds.get("typeId"),reqIds.get("interfaceId"),reqIds.get("mainId")));
        }else if (reqIds.get("mainId")==null){

titleList.addAll(projectIssueRepo.getTitleWithoutMain(reqIds.get("projectId"),reqIds.get("modul
eId"),reqIds.get("typeId"),reqIds.get("interfaceId")));

titleList.addAll(requirementsRepo.getTitleWithoutMain(reqIds.get("projectId"),reqIds.get("modu
leId"),reqIds.get("typeId"),reqIds.get("interfaceId")));
        }else if (reqIds.get("interfaceId")==null){

titleList.addAll(projectIssueRepo.getTitleWithoutInter(reqIds.get("projectId"),reqIds.get("module
Id"),reqIds.get("typeId")));

titleList.addAll(requirementsRepo.getTitleWithoutInter(reqIds.get("projectId"),reqIds.get("modul
eId"),reqIds.get("typeId")));
        }else if (reqIds.get("typeId")==null){

titleList.addAll(projectIssueRepo.getTitleWithoutType(reqIds.get("projectId"),reqIds.get("modul
eId")));

titleList.addAll(requirementsRepo.getTitleWithoutType(reqIds.get("projectId"),reqIds.get("modu
leId")));
        }else if (reqIds.get("moduleId")==null){
            titleList.addAll(projectIssueRepo.getTitleWithoutModule(reqIds.get("projectId")));
            titleList.addAll(requirementsRepo.getTitleWithoutModule(reqIds.get("projectId")));
        }else {

titleList.addAll(projectIssueRepo.getTitleByProjectIssue(reqIds.get("projectId"),reqIds.get("mod
uleId"),reqIds.get("typeId"),reqIds.get("interfaceId"),reqIds.get("mainId"),reqIds.get("subId"),req
Ids.get("miniId")));

titleList.addAll(requirementsRepo.getTitleByRequirement(reqIds.get("projectId"),reqIds.get("mo
duleId"),reqIds.get("typeId"),reqIds.get("interfaceId"),reqIds.get("mainId"),reqIds.get("subId"),re
qIds.get("miniId")));
        }
        return titleList;
    }
    /*Figre out the request type bug or cr*/
    public String issueType(TitleReqBean request){
        List<String> titleList = new ArrayList<>();


        HashMap<String, Integer> reqIds = findReqId(request);


        if (reqIds.get("miniId")==null){

titleList.addAll(projectIssueRepo.isBugOrCrOutMini(request.getTitle(),reqIds.get("projectId"),re
qIds.get("moduleId"),reqIds.get("typeId"),reqIds.get("interfaceId"),reqIds.get("mainId"),reqIds.ge
t("subId")));

titleList.addAll(requirementsRepo.isBugOrCrOutMini(request.getTitle(),reqIds.get("projectId"),r
eqIds.get("moduleId"),reqIds.get("typeId"),reqIds.get("interfaceId"),reqIds.get("mainId"),reqIds.g
et("subId")));
```

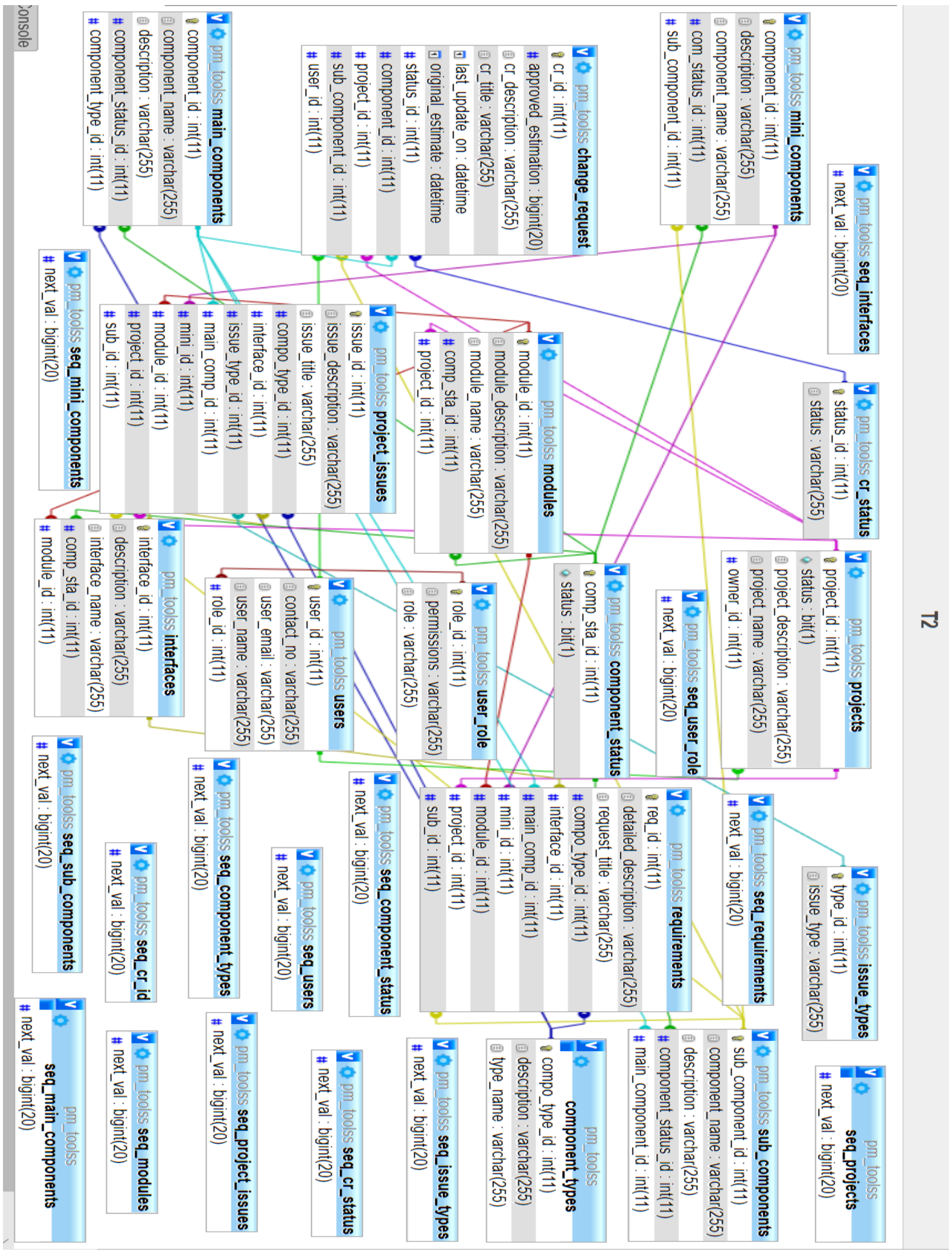
```
        }else if (reqIds.get("subId")==null){

titleList.addAll(projectIssueRepo.isBugOrCrOutSub(request.getTitle(),reqIds.get("projectId"),req
Ids.get("moduleId"),reqIds.get("typeId"),reqIds.get("interfaceId"),reqIds.get("mainId")));

titleList.addAll(requirementsRepo.isBugOrCrOutSub(request.getTitle(),reqIds.get("projectId"),re
qIds.get("moduleId"),reqIds.get("typeId"),reqIds.get("interfaceId"),reqIds.get("mainId")));
        }else if (reqIds.get("mainId")==null){

titleList.addAll(projectIssueRepo.isBugOrCrOutMain(request.getTitle(),reqIds.get("projectId"),re
qIds.get("moduleId"),reqIds.get("typeId"),reqIds.get("interfaceId")));

titleList.addAll(requirementsRepo.isBugOrCrOutMain(request.getTitle(),reqIds.get("projectId"),r
eqIds.get("moduleId"),reqIds.get("typeId"),reqIds.get("interfaceId")));
        }else if (reqIds.get("interfaceId")==null){

titleList.addAll(projectIssueRepo.isBugOrCrOutInter(request.getTitle(),reqIds.get("projectId"),re
qIds.get("moduleId"),reqIds.get("typeId")));

titleList.addAll(requirementsRepo.isBugOrCrOutInter(request.getTitle(),reqIds.get("projectId"),r
eqIds.get("moduleId"),reqIds.get("typeId")));
        }else if (reqIds.get("typeId")==null){

titleList.addAll(projectIssueRepo.isBugOrCrOutType(request.getTitle(),reqIds.get("projectId"),re
qIds.get("moduleId")));

titleList.addAll(requirementsRepo.isBugOrCrOutType(request.getTitle(),reqIds.get("projectId"),r
eqIds.get("moduleId")));
        }else if (reqIds.get("moduleId")==null){

titleList.addAll(projectIssueRepo.isBugOrCrOutModule(request.getTitle(),reqIds.get("projectId")
));

titleList.addAll(requirementsRepo.isBugOrCrOutModule(request.getTitle(),reqIds.get("projectId"
)));
        }else {

titleList.addAll(projectIssueRepo.isBugOrCrByProject(request.getTitle(),reqIds.get("projectId"),r
eqIds.get("moduleId"),reqIds.get("typeId"),reqIds.get("interfaceId"),reqIds.get("mainId"),reqIds.g
et("subId"),reqIds.get("miniId")));
titleList.addAll(requirementsRepo.isBugOrCrByProject(request.getTitle(),reqIds.get("projectId"),
reqIds.get("moduleId"),reqIds.get("typeId"),reqIds.get("interfaceId"),reqIds.get("mainId"),reqIds.
get("subId"),reqIds.get("miniId")));
        }
        String issueType;
        if (titleList.isEmpty()){
            issueType="Change Request";
        }else {
            issueType="Bug";
        }
        return issueType;
    }
}
```

7. 5 APPENDIX E SQL QUERIES

```
CREATE TABLE `change_request` (
  `cr_id` int(11) NOT NULL,
  `approved_estimation` bigint(20) DEFAULT NULL,
  `cr_description` varchar(255) DEFAULT NULL,
  `cr_title` varchar(255) DEFAULT NULL,
  `last_update_on` datetime DEFAULT NULL,
  `original_estimate` datetime DEFAULT NULL,
  `status_id` int(11) DEFAULT NULL,
  `component_id` int(11) DEFAULT NULL,
  `project_id` int(11) DEFAULT NULL,
  `sub_component_id` int(11) DEFAULT NULL,
  `user_id` int(11) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;


CREATE TABLE `component_types` (
  `compo_type_id` int(11) NOT NULL,
  `description` varchar(255) DEFAULT NULL,
  `type_name` varchar(255) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;


CREATE TABLE `component_types` (
  `compo_type_id` int(11) NOT NULL,
  `description` varchar(255) DEFAULT NULL,
  `type_name` varchar(255) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;


CREATE TABLE `cr_status` (
  `status_id` int(11) NOT NULL,
  `status` varchar(255) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;


CREATE TABLE `interfaces` (
  `interface_id` int(11) NOT NULL,
  `description` varchar(255) DEFAULT NULL,
  `interface_name` varchar(255) DEFAULT NULL,
  `comp_sta_id` int(11) DEFAULT NULL,
  `module_id` int(11) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;


CREATE TABLE `issue_types` (
  `type_id` int(11) NOT NULL,
  `issue_type` varchar(255) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;


CREATE TABLE `main_components` (
  `component_id` int(11) NOT NULL,
  `component_name` varchar(255) DEFAULT NULL,
  `description` varchar(255) DEFAULT NULL,
  `component_status_id` int(11) DEFAULT NULL,
  `component_type_id` int(11) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;


CREATE TABLE `mini_components` (
  `component_id` int(11) NOT NULL,
  `description` varchar(255) DEFAULT NULL,
  `component_name` varchar(255) DEFAULT NULL,
  `com_status_id` int(11) DEFAULT NULL,
  `sub_component_id` int(11) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;


CREATE TABLE `modules` (
  `module_id` int(11) NOT NULL,
  `module_description` varchar(255) DEFAULT NULL,
  `module_name` varchar(255) DEFAULT NULL,
  `comp_sta_id` int(11) DEFAULT NULL,
  `project_id` int(11) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

CREATE TABLE `projects` (
  `project_id` int(11) NOT NULL,
  `status` bit(1) DEFAULT NULL,
  `project_description` varchar(255) DEFAULT NULL,
  `project_name` varchar(255) DEFAULT NULL,
  `owner_id` int(11) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

```
CREATE TABLE `project_issues` (
  `issue_id` int(11) NOT NULL,
  `issue_description` varchar(255) DEFAULT NULL,
  `issue_title` varchar(255) DEFAULT NULL,
  `compo_type_id` int(11) DEFAULT NULL,
  `interface_id` int(11) DEFAULT NULL,
  `issue_type_id` int(11) DEFAULT NULL,
  `main_comp_id` int(11) DEFAULT NULL,
  `mini_id` int(11) DEFAULT NULL,
  `module_id` int(11) DEFAULT NULL,
  `project_id` int(11) DEFAULT NULL,
  `sub_id` int(11) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;


CREATE TABLE `requirements` (
  `req_id` int(11) NOT NULL,
  `detailed_description` varchar(255) DEFAULT NULL,
  `request_title` varchar(255) DEFAULT NULL,
  `compo_type_id` int(11) DEFAULT NULL,
  `interface_id` int(11) DEFAULT NULL,
  `main_comp_id` int(11) DEFAULT NULL,
  `mini_id` int(11) DEFAULT NULL,
  `module_id` int(11) DEFAULT NULL,
  `project_id` int(11) DEFAULT NULL,
  `sub_id` int(11) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

7. 6 - APPENDIX F - CLASS DIAGRAM FOR DATABASE



The End