A Web Based System to Manage Software Quality Assurance Process

A dissertation submitted for the Degree of Master of Information Technology

R M L S Rathnayake

University of Colombo School of Computing

2019

ABSTRACT

Software quality assurance verifies the activities of the software development life cycle and the goal is to ensure that the development and maintenance processes are continuously improved to produce products that meet specifications. The quality assurance of software plays a vital role in the field of software engineering, compared to many other fields such as electronic engineering, transportation etc. The QA in the software field is very much more complex due to the intangibility of the product, invisibility of the product and the opportunity to detect defects are minimum.

The problem identified is that in order to ensure the quality of software there should be proper management systems that is used for the entire development cycle of the software in order to plan, track and manage the software development projects. Most companies developing software is recording the tasks related to SQA manually this causes data duplication, data dependence and incompatibility of files. In this project an attempt is made to develop a Software Quality Assurance web application.

The implemented web-based portal can be used to create and store all the test artifacts such as test cases, test execution results, defect tracking, result comparison and day to days tasks of SQA engineers (sprint board) in the software industry. Primary users of this web application will be software quality assurance engineers who will be designing test cases, executing the designed test cases, maintaining results of the executed test cases, reporting defects, verifying defects. The secondary users of this application will be project managers, software engineers, business analysts and the top management of software companies.

This system can enable organizations worldwide to test high-quality software which meets the stringent demands of various industries and meet high customer expectations. It encompasses the software industry experts to maintain the highest level of business integrity and service.

The high-level architecture of the developed web application is the client end will consist of an angular 5 project that interacts with the user and makes API calls. The server end consists of a spring boot +JPA (Hibernate) app MySQL DB as database. The web application developed has been successfully implemented on site. Since the system was designed to be portable, it can be easily implemented on any organization.

DECLARATION

The thesis is my original work and has not been submitted previously for a degree at this or any other university/institute.
To the best of my knowledge it does not contain any material published or written by another person, except as acknowledged in the text.

Student Name: R. M. L. S. Rathnayake
Registration Number: 2016/MIT/061
Index Number: 16550612

_____

Signature: Date:

This is to certify that this thesis is based on the work of
Mr./Ms.
under my supervision. The thesis has been prepared according to the format stipulated and is of acceptable standard.

Certified by:
Supervisor Name:

_____

Signature: Date:

ACKNOWLEDGEMENT

This dissertation arose as an effort of number of people whose contribution in assorted ways in the realization of this project deserves special mention. It is a pleasure to convey my gratitude to them all through this humble acknowledgment.

I gratefully acknowledge Mr. K. P. M. K. Silva, supervisor of the project, for his supervision, advice, and guidance from the very early stage of this project. He provided me with unflinching encouragement and support in various ways. He was a constant oasis of ideas and passionate in teaching, which exceptionally inspire and enrich our growth as students, and professionals. I am indebted to him more than he knows.

I would also acknowledge my parents and all the friends who provided their support in making this project a success.

Finally, a big thank to each person for the support and the assistance given to make this a success.

TABLE OF CONTENTS

List of Figure

List of Tables

List of Acronyms

UCSC - University of Colombo School of Computing

MIT - Masters in Information Technology

UML - Unified Modeling Language

SDLC - Software Development Life Cycle

UI - User Interfaces

SQA – Software Quality Assurance

HP ALM – Hewlett-Packard -Application Life Cycle Management

EER – Enhanced Entity Relationship

SQL – Structured Query Language

UFT – Unified Functional Testing

CLI – Command Line Interface

IT – Information Technology

API – Application Programming Interface

JPA – Java Persistence API

Chapter 01

INTRODUCTION

1.1 Introduction

Software quality assurance (SQA) is a process that ensures that the developed software meets and complies with defined or standardized quality specifications. SQA is an ongoing process within the software development life cycle (SDLC) that routinely checks the developed software to ensure it meets desired quality measures. SQA helps ensure the development of high-quality software. SQA practices are implemented in most types of software development, regardless of the underlying software development model being used. In a broader sense, SQA incorporates and implements software testing methodologies to test software. Rather than checking for quality after completion, SQA processes test for quality in each phase of development until the software is complete. With SQA, the software development process moves into the next phase only once the current/previous phase complies with the required quality standards [1].

A QA web application helps development teams ensure software quality. It helps managing software test cases, delegating test executions to testers, tracking test metrics and test data and much more. The workflow is flexible so the teams can determine themselves how and when testing would take place in the application development lifecycle.

In some instances, a manual process is used for SQA practices. Spreadsheets are what most companies use as a duct-tape solution to a big problem. It might work for small projects or products, but as many projects grow software field specialists and management need to look for more sophisticated and productive solutions that will ensure a more systematic process.

To put this in perspective, if an application handles a simple micro service that only handles a very few functionalities, you might be able to get away with using spreadsheets, as simple functions translates to very limited amount of test cases.

Also, the development team might be a single person or few developers for such a project. However, trouble occurs when new functions are added and the applications grow both logically and in design, and how internal modules interact with each other. Therefore, it will be required to document expected behavior for end points in all configurations/given parameters.

Implementing a quality management system is similar to an art as it is to a science. It is an art to manage people and a science to have a systematic process approach to quality and verification.

Software quality assurance is a planned effort to ensure that a software product fulfills these criteria and has additional attributes specific to the project, e.g., portability, efficiency, reusability, and flexibility. It is the collection of activities and functions used to monitor and control a software project so that specific objectives are achieved with the desired level of confidence. It is not the sole responsibility of the software quality assurance group but is determined by the consensus of the project manager, project leader, project personnel, and users. A formal definition of software quality assurance is that is 'the systematic activities providing evidence of the fitness for use of the total software product."

Software quality assurance is achieved through the use of established guidelines for quality control to ensure the Integrity and prolonged life of software. The relationships between quality assurance, quality control, the auditing function, and software testing are often confused.

Web based software quality assurance systems are designed to manage and store project QA information used as web-based applications. Different groups of people such as, Software SQA engineers, business analysts, programmers or project managers will be let by project applications a controlled access to information and automated distribution of information. The objective for collaboration has been getting things done faster, cheaper and better by applying common knowledge by bringing together a selection of resources and attainments in a project. Valid collaboration with teams improves productivity, speeds up result-making and optimizes making of right decisions, it also helps to intercept precious intellectual fortune and time. To prove such kind of improvement to productivity and to make easier our everyday working life, it was needed to make an inside system for QA management. Namely, having troubles of finding right Defect Reports and wasting useful time for sending and searching documents,

describing and instructing new employers of the whole QA Process and steps that needs to be done before beginning to make changes in projects or code in the next cycles has to be solved.

1.2 Problem Identified

In today's software industry, quality assurance of software plays a vital role. Due to this factor it is very important that SQA records are properly maintained and recorded with backup for future use or for a risk mitigation. The results of the tested software are mainly maintained as test case results in today's SQA industry.

The problem is that there should be one system to cater to the entire software quality engineering process, which includes both defect and test case management, and the project management process.

The existing systems now cater to one of these therefore if there is a system that includes multiple tasks such as defect and test case management and the project management process it will simplify day to day activities of QA engineers and other stakeholder employees such as Project Managers, Developers and Business Analysts.

Following main services should be taken into our consideration when selecting a test management tool.

- Project management –Create and manage project planning activities, draft sprint plan, create backlogs against the features and requirements, members.
- Test management and plan – Creating test plans, test suits, test cases, test scenarios or user stories and activating them as per required duration.
- Test run – Creating test runs, execute test cases, managing members of the testing process.
- Making test reports - On selected project or against the test plan etc.
- Task management – Create features, manage user stories, Agile sprint task board, burn down charts etc.
- Capacity planning - Estimate both the amount of work and types of work required to complete in sprint plan.

- Additional features – import/export test cases, integration with other systems, Dashboards etc.

The proposed solution is to implement a web-based portal that can be used to create and store all the test artifacts such as test cases, execution results. Furthermore, SSQA engineers can report defects, change the status of the defects and also compare test results. The application will enable end users to plan agile related tasks and support an organization using an Agile Software Development Cycle.

1.3 Aims and objectives

The main objective is to build a web-based system to manage software test management activities which includes writing manual test cases, executing manual test cases with results, creating test plans, creating relevant test suites.

A separate feature is developed for defect management the purpose of defect management is to provide information to improve the development process. This way of defect management is important in the continuous monitoring of product quality throughout the whole lifecycle of the product. This feature assists development team and management to know when operational is support needed for solving and retesting defects. Then the team can identify and analyses the causes of defect and classify them.

The users that intended to use this software are SSQA engineers, developers, management and business analysts. This is aimed at simplifying many days-to-day's activities of SSQA engineers of a software c company. Customizable, highly configurable dashboards provide to the management and teams with the flexibility to share information, monitor progress and trends, and improve the workflow processes.

This application is not restricted to use for a single project and can be used across multiple projects.

Task Board provides a visualization of flow and status of each sprint task. With it, team can focus on the status of backlog items as well as work assigned to each team member. Apart from these, using the capacity-planning tool, team can estimate both the amount of work and types of work required to complete its sprint plan.

1.4 Scope

This project is aimed at implementing a web-based system, which will give complete traceability for the test management process.

- The developed system facilitate management of testing activities of the software development projects and collaborate on projects all in one place. It includes create tasks, user stories and backlogs and features against the epics.

- Create test plans, create manual test cases to check that each of the deliverables meet the requirements. Organize test cases by adding test cases to test suites, test scenarios based on a feature and separate them accordingly with sprints and iterations.

A feature is developed to run manual test cases and record the test results for each test step by denoting their results (Pass / Fail / Block / On hold). Since this is aimed at a software company that uses the agile software development methodology, it includes test case execution results as per Sprints and Iterations. Test case maintenance view of the software QA web application is shown in figure 1.1.

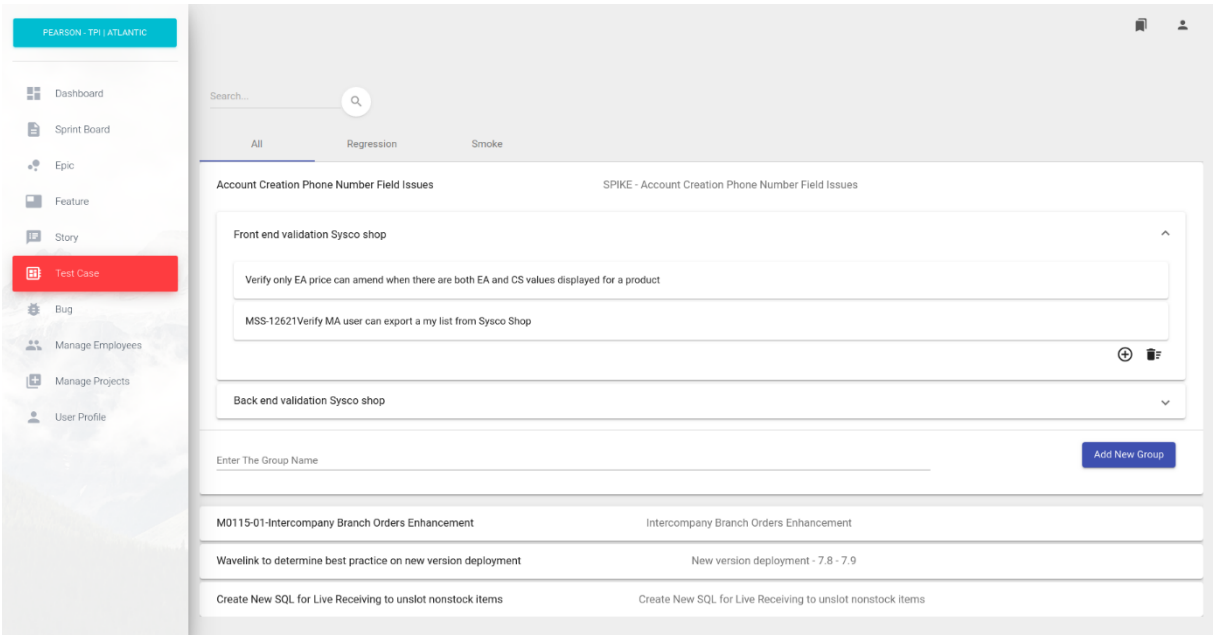

Figure 1.1 Maintenance of test cases in the software QA web application

- If users find an issue when testing, users can create bugs with test steps, screenshots, and comments. Users can also maintain the entire life cycle of bugs and link them with test cases and backlogs.
- The user can build up different types of test suits Ex: requirement-based test suites, regression suite, smoke suite, query-based test suites.
- A dashboard is to be developed to propagate test results for the Higher Management and monitor the ongoing process of the testing activities.

1.5 Structure of the dissertation

Introduction of the project "A Web Based System to Manage Software Quality Assurance Process" was described in this chapter. In chapter 02, the background information of the project and information about similar approaches is described. Chapter 03 describes the designing phase of the project with the methodologies used. And it provides a detailed explanation about the proposed solution details and the implementation of the project. Project evaluation criteria and the findings are described in the 04th Chapter. In chapter 05 further works is described along with the conclusion of the project. References are stated at the end of the dissertation, which is followed by the appendix.

Chapter 2

BACKGROUND

2.1 Introduction

Everyday life has become dependent on software and software-based systems. This strong dependency on software requires higher investments in quality assurance activities to enable IT systems to perform reliably. Hence, immense attention should be exercised while developing a test management tool.

Test management is the process of organizing & controlling test processes, test assets and artifacts for manual or automated testing projects. The goal of quality assurance process is to improve test success and thereby increase the software quality. Due to this factor it is very important that Quality Assurance records are properly maintained and recorded with backup for future use.

A comprehensive test management tool should be able to track how testing is planned, report the status of QA activities, and organize test artifacts in a systematic manner through entire software testing life cycle. The phases of software testing life cycle is shown in figure 2.1.

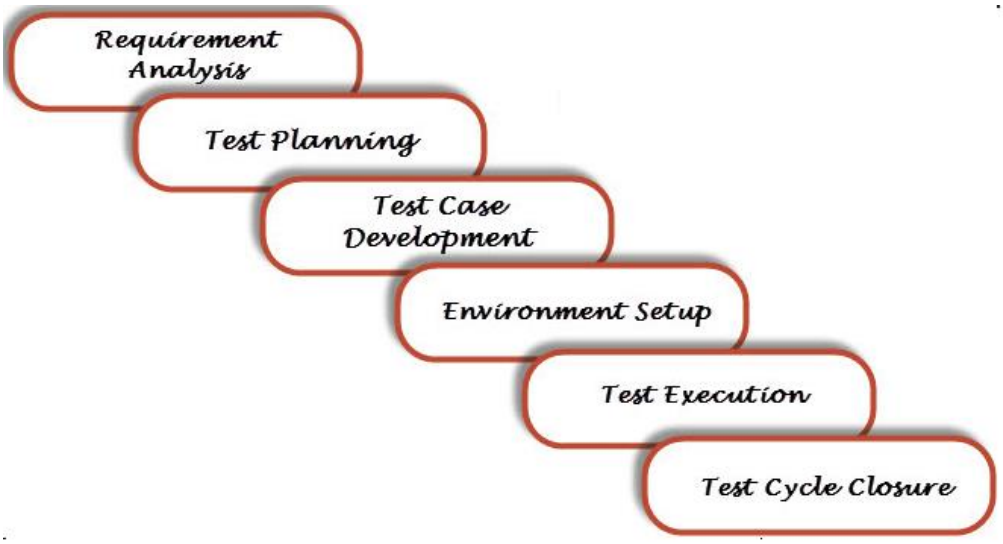

Figure 2.1 Phases of Software Testing Lifecycle [2]

Below are some tasks and activities that are involved in quality assurance process:

Analysis

- Requirement analysis
- Creating documentation related to testing activities
- Identifying the scope of improvement in Application testing

Planning

- Test plan development
- Test execution planning
- Test scheduling
- Test efforts estimation
- Measuring & tracking test cycles

  Test case Development
- Test case preparation
- Organizing test cases

Execution

- Test execution monitoring
- Defect tracking
- Test result reporting

Test cycle closure

- Test result analysis
- Managing test assets and artifacts
- Checking the testing tasks and test result completeness

There should be one system to cater to the entire software quality engineering process which includes both defect and test case management and also the project management process.
The problem is existing systems now cater to one of these. Therefore, if there is a system that includes both defect and test case management and also the project management process it will simplify the day to day activities of SQA engineers and also other stakeholder employees such as project managers, developers etc.

2.2 Analysis of existing systems

General requirements of the system are listed below:

- Project management

  In the recent past most software projects are adopting the agile methodology, so the proposed project management tool should also be able to support agile. It includes support for features like Project creation and different types of work items such as User story creation, Sprints, Capabilities, Epics, Features, Tasks and create Backlogs against the features.

- Capacity planning and time tracking

  Estimate both the amount of work and types of work required to complete in sprint plan. Easily track how much work the team has completed and has left to do in a sprint by adding the sprint capacity charts to dashboard.

  Estimates and time tracking: Track estimated, completed, and remaining work for tasks and other work items.

- Test management and plan

  The system should allow user to create manual test cases, test scenarios to check that each of the deliverables meet users' needs and organize test cases by adding test cases to test suites. Test suites provides a way to group test cases for separate testing scenarios within a single test plan.

  Test plans are used to group together test suites and individual test cases. This includes regression test suites, smoke test suites, and functional test suites. User can add individual test cases to a test suite and then link it to the test plan.

- Test run

  A feature developed that was to execute Manual Test Cases and record the test results for each test step by denoting their results (Pass / Fail / Block / On Hold etc.). Summary of the test results, including pass/fail percentages grouped or filtered against selected criteria E.g. Test plan or Test suite.

- Making test reports on selected project or against the test plan etc. Track the status of test progress and test runs.

- Task management
  Visualize user stories and link tasks to backlog work items. This Agile board can be used to add work hours to a task and change the status of tasks (New, In-progress, Closed).

- Bug tracking - Each team can manage bugs on their backlog or along with test plan.

- Charts and dashboards – A Dashboard is developed to escalate test results for the higher management and monitor the ongoing process of the testing activities it helps to keep both the team and stakeholders in sync.

2.3 Review of existing systems

1. HP Application Lifecycle Management

HP ALM is well suited for waterfall projects specifically if the teams are novice. It provides excellent support for project planning, tracking & test management. Top leadership can efficiently track, measure and report on project milestones & key performance indicators [3].

ALM includes quality assurance features for risk-based test planning and management, version control, base lining, quality release and cycle management, test scheduling and execution, integrated manual testing and defect management. HP Quality Center is a quality management platform that can be used for a single project or across multiple IT projects to manage application quality across the entire application lifecycle [4].

ALM provides requirements management, release and cycle management, test management, defect management and integration to all other HP products such as UFT and Load Runner.

HP ALM workflow.

- Release specification:

ALM user can organize and track releases and cycles in HP ALM. A cycle which falls within a release has a set of development and testing efforts in order to achieve a common goal. ALM users can track the progress of the project in real time by analyzing the releases tree to ensure if it matches the release goals [5]. Figure 2.2 displays the releases view screen of HP ALM application.

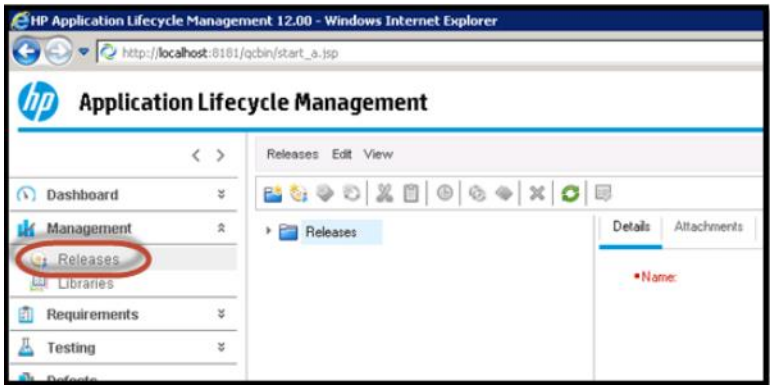

Figure 2.2 Creating releases the first step in ALM [5]

- Requirements Specifications.

This module in ALM enables users to define, manage and track requirements [5]. Figure 2.3 displays requirement management feature of the HP ALM application.

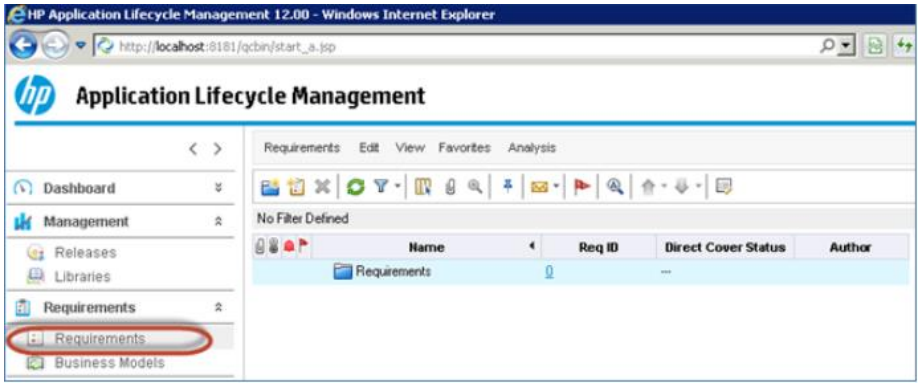

Figure 2.3 Tracking Requirements [5]

- Test Planning

ALM supports maintenance and execution of manual, automation and performance tests as ALM is seamlessly integrated with all HP products such as HP UFT and HP Load Runner [5]. Figure 2.4 describes the test plan creation steps in ALM.

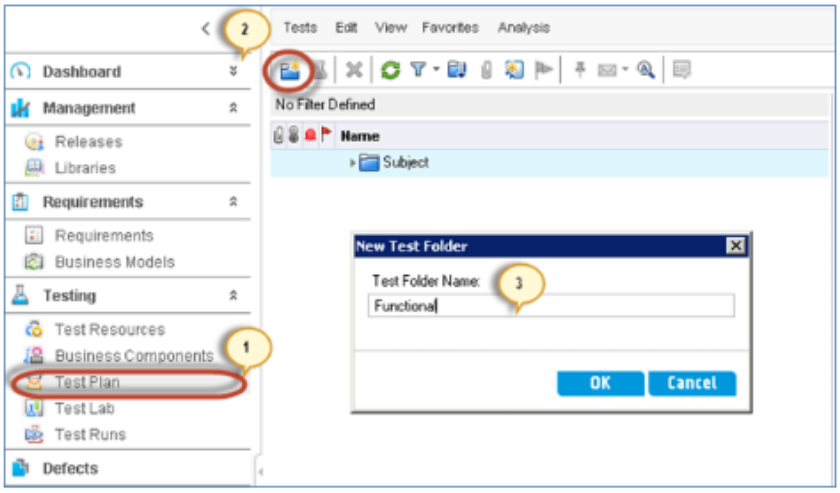

Figure 2.4 Creating Test Plans on ALM [5]

- Test Execution

Once the test design is completed, test execution will take place with the help of Test Lab module [5]. Below figure 2.5 displays the test execution process in HP ALM system.

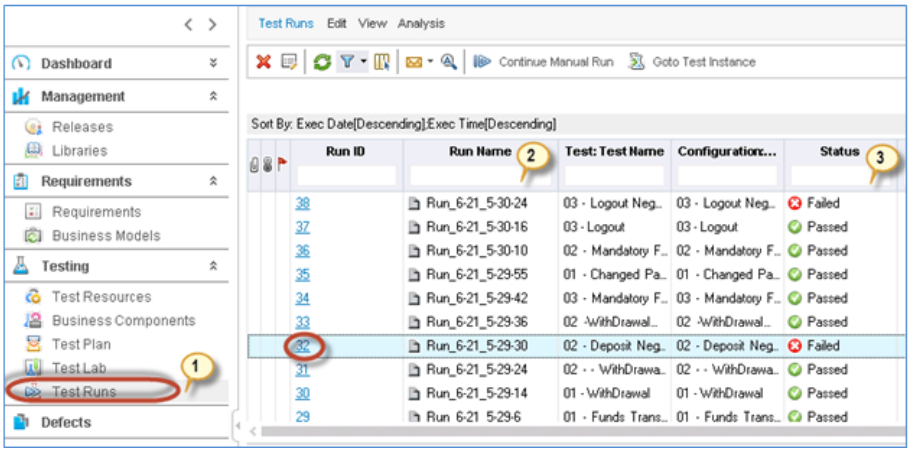

Figure 2.5 Test execution on ALM [5]

- Defect tracking.

  Defect module in HP ALM not only helps users to post the defects but also enables them to track and gives the overall quality of the release at any stage of the development process [5]. Below figure 2.6 is the defect management view of HP ALP application.

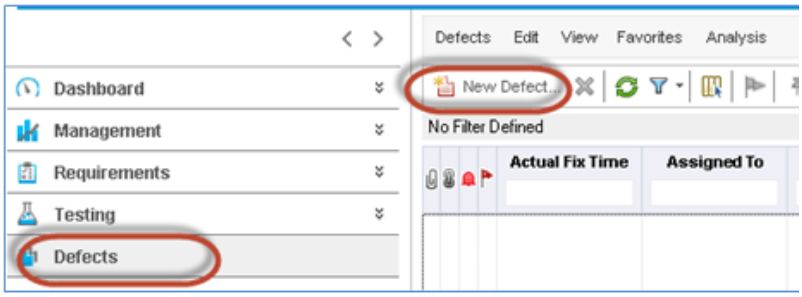

Figure 2.6 Defect tracking on ALM [5]

2. Microsoft Team Foundation Server

Team Foundation Server is a Microsoft product that provides source code management, reporting, requirements management, project management, automated builds, lab management, testing and release management capabilities. It covers the entire application lifecycle, and enables DevOps capabilities [6].

Available Features

- Agile tools to plan and track work

  Create user stories and backlogs: Plan project by adding a work item for each user story or requirement that the user intends to develop.

  Storyboard: A user story is the smallest unit of work in an agile framework. It's an end goal, expressed from the software user's perspective.

- Tasks and issue tracking: User is able to create different types of work items such as User story creation, Sprints, Capabilities, Epics, Features, Tasks and create Backlogs against the Features [7]. Figure 2.7 displays available work items in the Microsoft team foundation server.

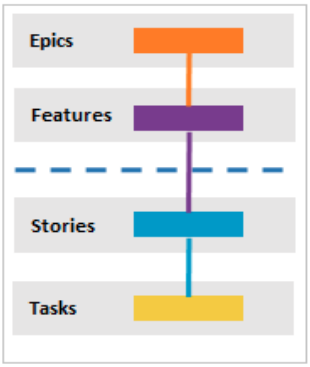

Figure 2.7 Types of work items on a sprint [7]

- Estimates and time tracking: Track estimated, completed, and remaining work for tasks and other work items.

- Plan sprints, Velocity & forecasting, Manage resources and Sprint burn down charts.
    - Monitor progress and review team patterns from sprint burn down charts [8]. The following figure 2.8 shows an example of a sprint burn down graph.

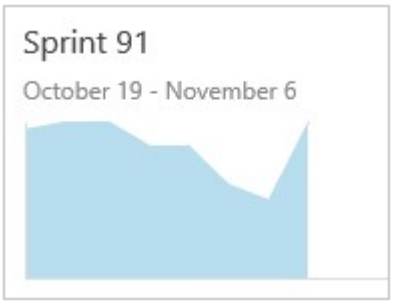

Figure 2.8 Sprint burndown chart [8]

    - Build sprint backlog, add tasks, and load balance work across team as you plan your sprint.

- Use velocity charts and forecast tools to estimate work that can be completed in future sprints [9]. Below figure 2.9 displays an example of a velocity planning graph.

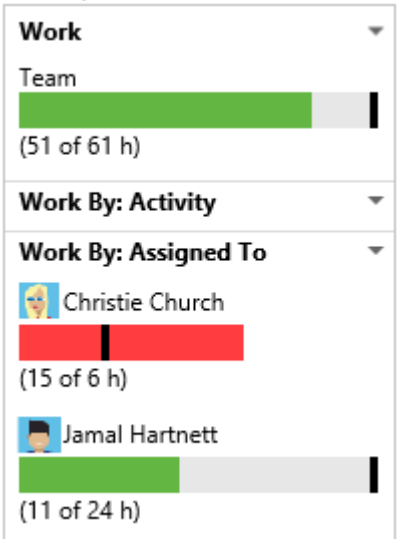

Figure 2.9 Capacity planning [9]

- Test Management

  User can create manual test cases, test scenarios by adding test cases to test suites. Test suites separate testing scenarios within a single test plan [10]. Below figure 2.10 displays test suite management process in Microsoft team foundation sever.

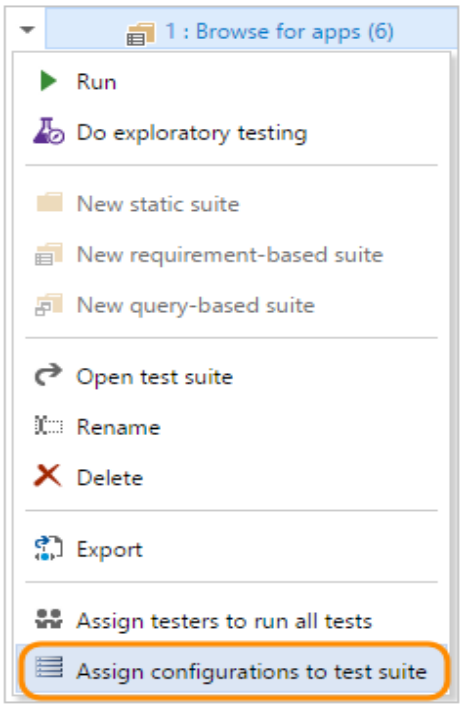

Figure 2.10 Test Suite Management [10]

- Bug tracking feature.

  User can track bugs on the Agile board and link them to a test case. Below figure 2.11 is the bug creation page in TFS.

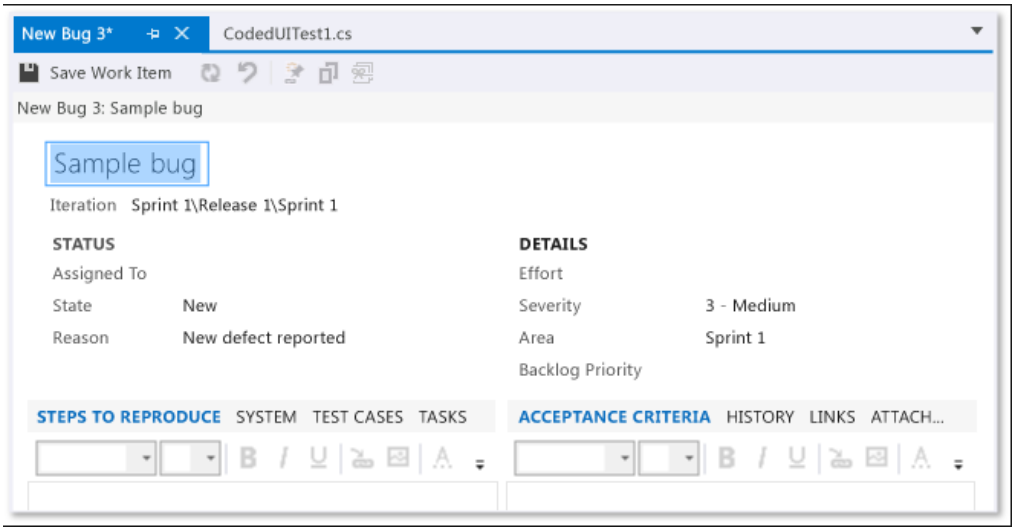

Figure 2.11 Bug Tracking Feature [11].

2.4 Existing Web Based Test Management Tools

1. Jira

   Jira is a proprietary issue tracking product developed by Atlassian which allows bug tracking and agile project management. Jira is a commercial software product that can be licensed for running on-premises or available as a hosted application [12].

2. qTest

   qTest provides software testing and development teams with an easy to learn, easy to use, lightning fast, scalable test management that seamlessly integrates with JIRA, other ALMs, and automation tools. qTest is one of the fastest growing test management solutions on the market today amongst the other Agile testing and development teams [13].

3. TestLink

   This is one of the very few open source test management tools that is available for use in the market. It is a web-based tool with typical features like requirement management, test case creation, and maintenance, test runs, tracking bugs, reports, integration with common issue trackers etc. [13]

4. PractiTest

   PractiTest is a SaaS end-to-end QA management system with some of the most advanced and interesting features. With PractiTest, Testers are able to focus on quality and their actual work rather than side tasks [13].

   Following table 2.1 contains a summarized comparison among the features available in developed system and the prevailing systems.

Table 2.1 Comparison of proposed system with prevailing system

Feature	Developed System	TFS	ALM	Jira	Test Link
Agile support	Yes	Yes	No	Yes	Yes
Test case, Test suite and Test Plan creation	Yes	Yes	Yes	No	Yes
Requirement traceability	Yes	Yes	Yes	Yes	Yes
Defect Reporting	Yes	Yes	Yes	Yes	Yes
Dashboard	Yes	Yes	Yes	Yes	No

Most Systems mentioned above are enterprise solutions and require a large financial allocation in order to use it. Therefore, the application developed will be a good solution for organizations who are looking forward to manage both Test Case management and project management without purchasing the application and is most suitable to start up IT companies and also small-scale software companies.

ALM and TestLink does not have project management functions such as sprint planning, agile dashboards etc. JIRA does not have test management functionalities. Therefore the developed web application can be redefined as a hybrid solution for both test case management and project management functionalities.

Chapter 3

METHODOLOGY

3.1 Introduction

In any project, analyzing and design are the core aspects. That is the main idea, which is being generated throughout the previous chapters. This chapter contains information about the project analyzing and the design. Development methodology is further discussed in order to provide a clear idea about the developed system. The analyzing part of this chapter contains the details about the system and the justifications. The design section explains the architecture of the system and a brief explanation of the workflow.

Designing the correct system must be carried out through selecting proper design techniques and methods. Agile Methodology is used to develop the system. Testing is integrated throughout the project lifecycle, enabling regular inspection of the working product as it develops. This allows us to make necessary adjustments. In agile development, change is accepted. Instead the timescale is fixed and requirements emerge and evolve as the product is developed.

3.2 System Analysis

The final outcome of this project is a a web based system developed to manage the process of software quality engineering and project management process which includes writing test cases, executing Test Cases with results, creating test plans, creating relevant test suites etc. The users who intended to use this software are SSQA engineers, Developers, Management and Business Analysts. This is aimed at simplifying many day to day activities of SQA engineers and other stake holders of a Software Company.

3.2.1 Requirement Analysis

Requirements analysis is critical to the success of a development project. Requirements must be actionable, measurable, testable, related to identified business needs or opportunities, and defined to a level of detail sufficient for system design

Testing is the process or activity that checks the functionality and correctness of software according to specified user requirements in order to improve the quality and reliability of

system. It is an expensive, time consuming, and critical approach in system development, which requires proper planning of overall testing process.

A successful test is one that finds the errors. It executes the program with explicit intention of finding error, i.e., making the program fail. It is a process of evaluating system with an intention of creating a strong system and mainly focuses on the weak areas of the system or software

3.2.2 Functional Requirements

This requirement for this system is to build a Software Quality Assurance Management system including the software project management activities. Which is inclusive of the process of software testing such as:
Create Test Plans, create manual test cases to check that each of the deliverables meet the requirement, execute test cases, create bugs, organize test cases by adding test cases to test suites, test scenarios based on a feature. Figure 3.1 displays three main types of test management artifacts.

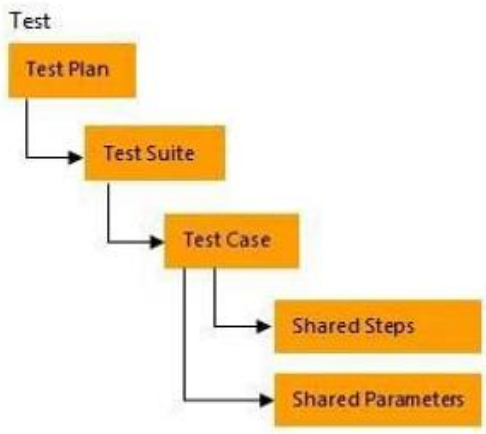

Figure 3.1 Agile Process of Testing [14]

The system which facilitates project management activities of the software development projects and collaborate on projects all in one place. It includes create Epics, create Features against the Epics, create User Stories and Backlogs. User will be able to create tasks and report time daily basis.

3.2.3 Use case diagrams for developed system

The below figure 3.2 is the high level use case diagram of the developed software QA application.

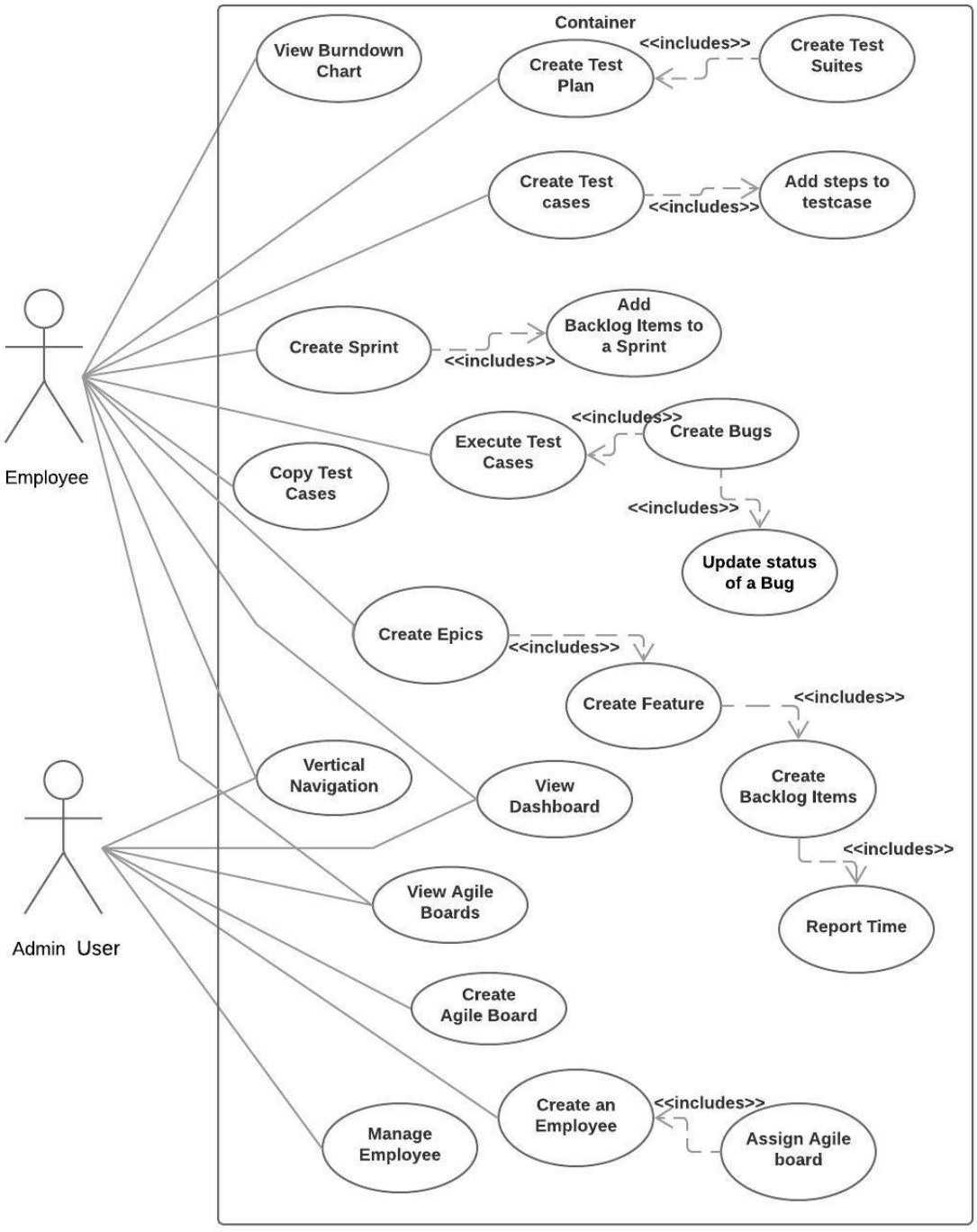

Figure 3.2 Use case diagram for the intended system

3.2.4 Use cases

The users in the system are the employees and the administrators. Only the authenticated users have been allowed to perform the software operations. The following use cases are the high level functional requirements.

Employee view

- Employee can login to the system
- Employee will have a vertical navigation panel
- Employee can create test plans
- Employee can organize test cases by adding test cases to test suites
- Employee can create test cases to each feature
- Employee can add test steps to each test case
- Employee can copy test cases to different suites
- Employee can execute test cases
- Employee can create bugs to each test cases
- Employee can update status of a defect according to defect lifecycle
- Employee can create epics
- Employee can create feature list against the Epics
- Employee will see assigned Agile boards
- Employee can go in to the Agile board and create backlog items
- Employee can add backlog items to each sprint.
- Employee can create sprint to each Agile board
- Employee will see burndown chart
- Employee can view dashboard

Admin view

- Admin user can login to the system.
- Admin user will have a vertical navigation panel.

- Admin user has an Agile board list.
- Admin user can create an Agile board.
- Admin user can assign employees to each Agile board.
- Admin user can view employee list.
- Admin user can create an employee.
- Admin user can view dashboard.

3.3 Non - Functional Requirements

As the basic requirement of this project it can be defines as a process which assists SQA engineers in Software companies to track the daily activities and record their work.

- **Performance Requirements**

Efficiency of the system is another main requirement of the project. The output of the system needed to be provided to the user real time. Lateness of the system will not be good since the user has to wait more time to get the result from the system. The SQA engineers use these data that are tracked as test case results.

Response Time – Response time of the system should be within an acceptable time period (2000ms – 3000ms)

Capacity - The System must support several people at a time.

User-interface - The user-interface screen shall respond within 5000 milliseconds.

- **Accuracy**

Accuracy of the system is considered as a main requirement of the project. SQA engineers depend on the output that is being provided by the system, Because of risk in delivering a Project.

- **Security**

Security is another main aspect that needs to be achieved from the developed system

The system should be more reliable because the SQA engineers use these data that are tracked as test case results they depend on the application in identifying the process and the correct result should be transmitted to them when they use the application. The system should be tested perfectly before the application is released.

Errors - The system shall keep a log of all the errors.

The system should have the ability to evolve with the time. Once the changes that need to be done are identified while the user use the system, they can be applied and provided to the users as a new release of the application.

3.4 High-level architecture

Client end consists of an angular 5 project that interacts with the user and makes API calls. The server end consists of a spring boot +JPA (Hibernate) app MySQL DB as database. Below figure 3.3 describes the high level architecture of the developed system.

Figure 3.3 High level architecture

This consist of main two user roles Admin and Engineer (employee). Admin can create an employee. Admin can create an agile board (Agile board represents each project that employee

assign to). Employee can log on to the system and see the agile board list (Agile board represents each project that employee assign to). On click agile board employee goes to backlog item list. Employee can create a backlog item. Employee can create sprints for each agile board. Backlog items be added to each sprint in sprint planning. Employee can create a feature list for each sprint. That each feature can have test cases and each test case can have bugs.

3.5 Database design

Good designing is very important in developing a good system. To convert the analyzed requirements into code, designing should be done in a proper way. Database design illustrates the table structure of the database, the relationships among tables and how each entity joins with other entities of the database. This information has been depicted using EER Diagram. Below figure 3.4 displays the database design of the system.

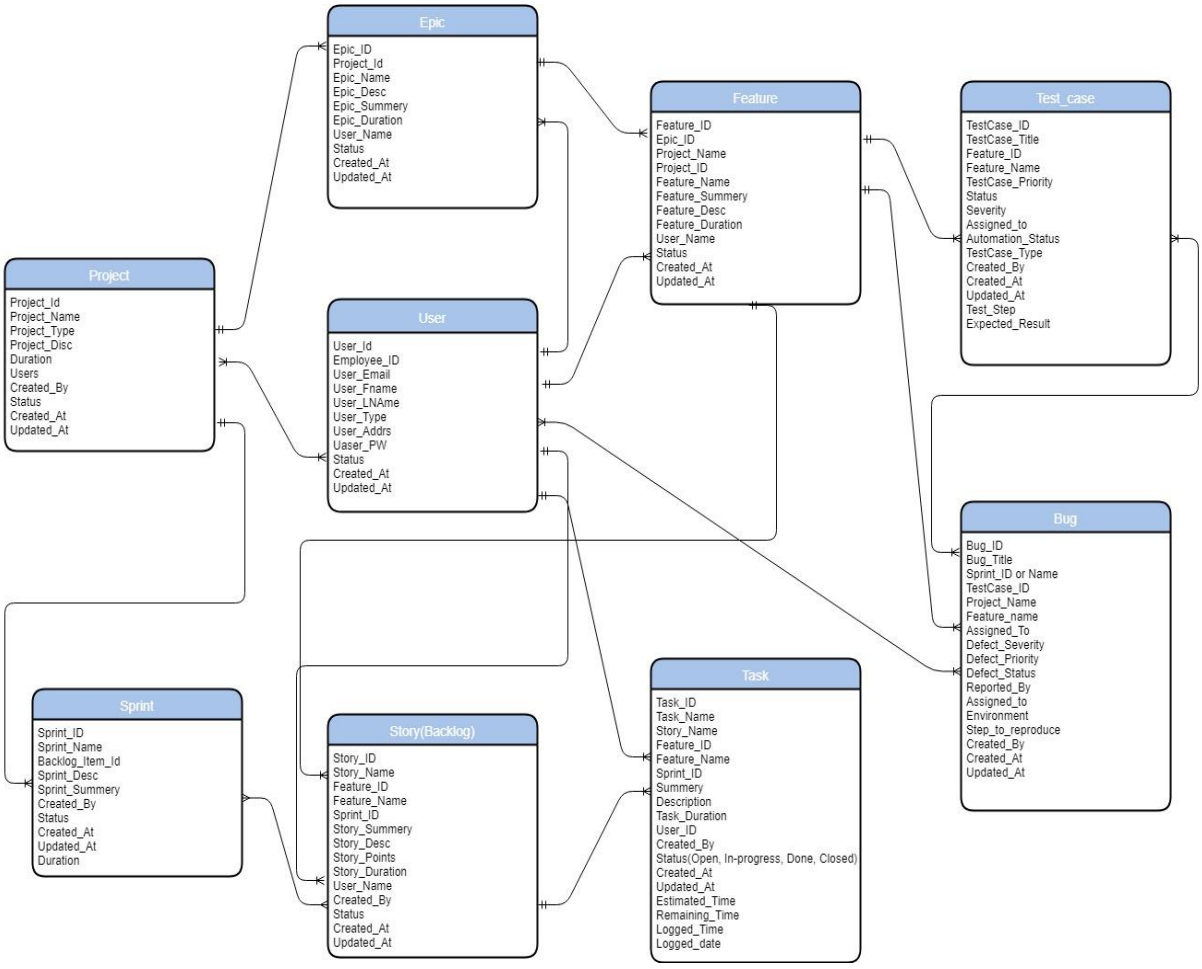

Figure 3.4 Database design

3.6 UI Design

The user of the developed system requires that the developed software should be user friendly, have security access, and ensure the privacy of the administrator and produce results in timely manner. The users are frequently exposed day-to-day tasks, so the system interface to the user must be simple and understandable. The web pages must be user-friendly and must be in an easy-to-use style. The user must be able to easily switch among various I/O screens. The product is well designed so that it can be used easily by layman and also the users who are novices to the system.

The system should be designed in such a way that only authorized users should be allowed to login to the system. The user interface should be as interactive as possible. A user-friendly interface must be provided so that the user can easily interact with the system and comprehend things in a quicker and easier way. The system must provide reliable and up-to-date information.

The application should be efficient so that the user does not spend much time in training. Consistency will increase the confidence of the user in the reliability of the application. The user must be limited with a small set of operations to achieve the result. The application should be visually and conceptually clear. The interface should accommodate user mistakes easily and fast. It should minimize the errors and should handle them peacefully. Interfaces below,

3.6.1 Login screen of the SQA web application

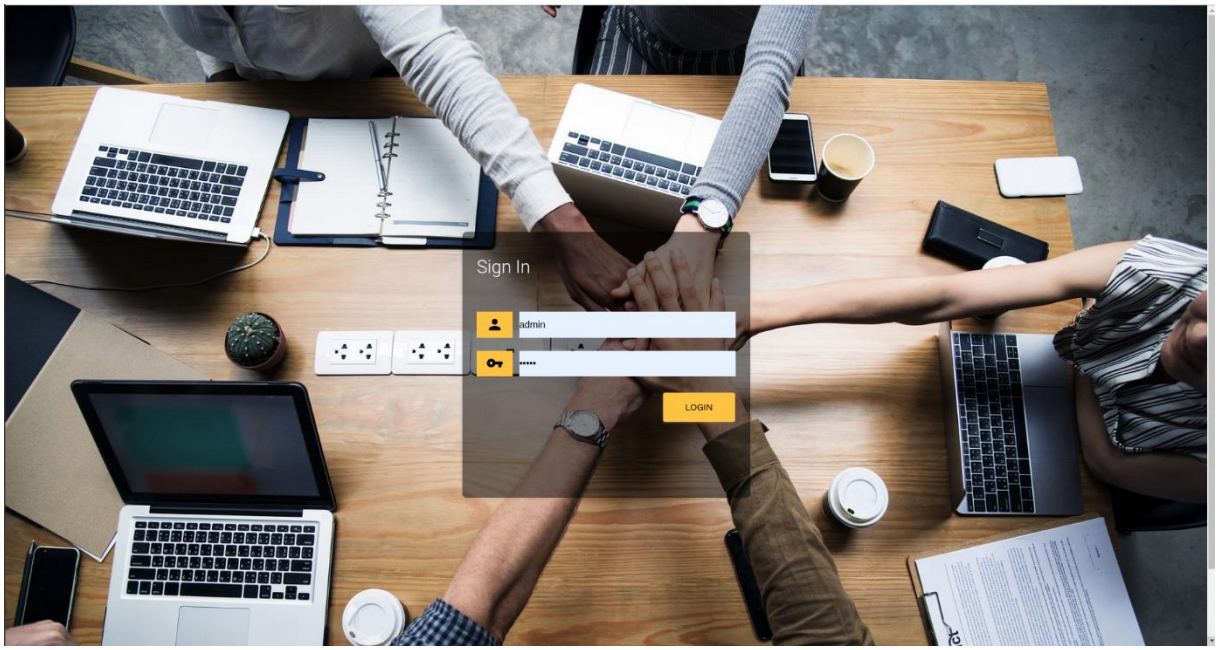

Figure 3.5 - Log in screen

3.6.2 Home page of the SQA web application

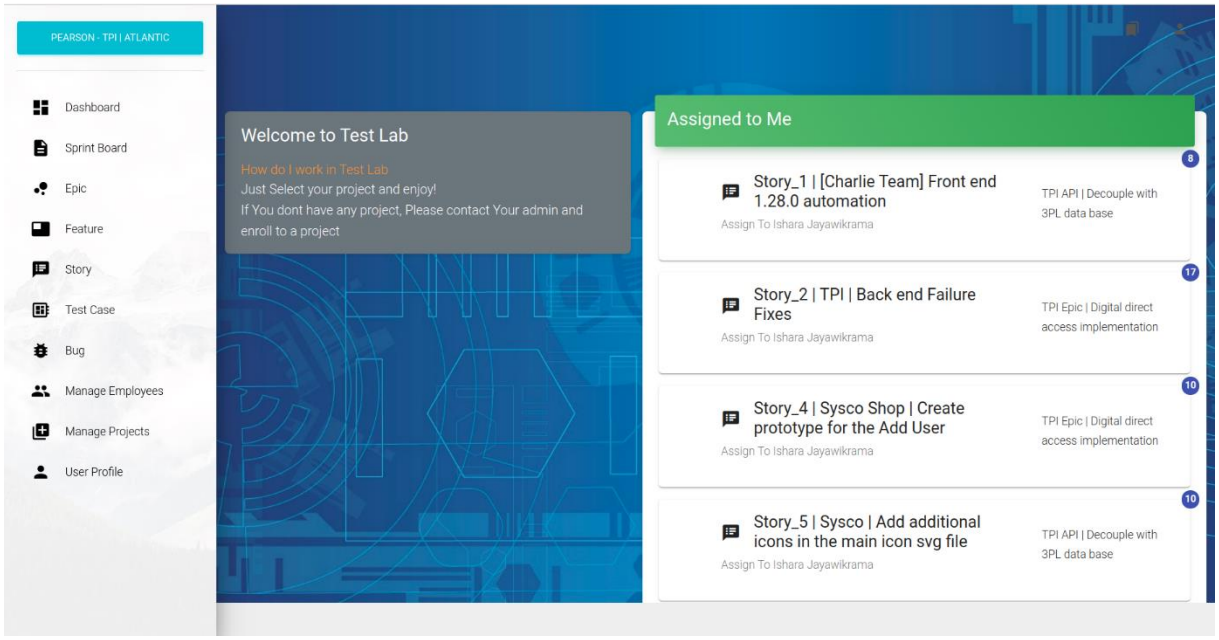

Figure 3.6 Home page of the SQA web application

3.6.3 Dashboard page of the SQA web application

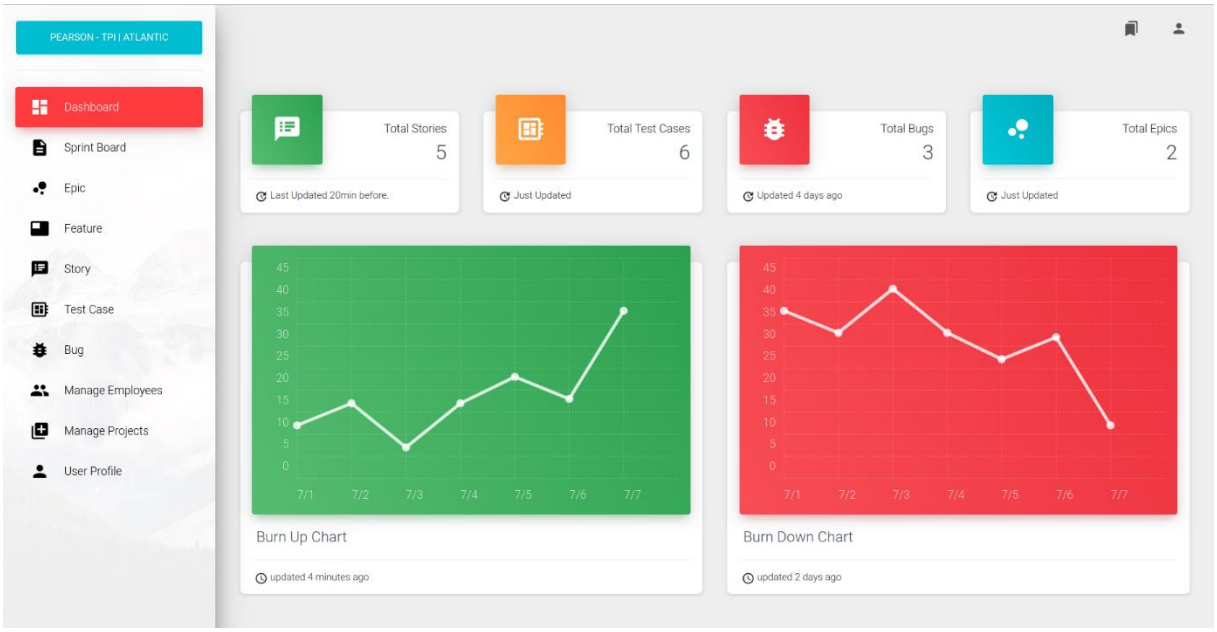

Figure 3.7 Dashboard page of the SQA web application

3.6.3 Sprint board of the SQA web application

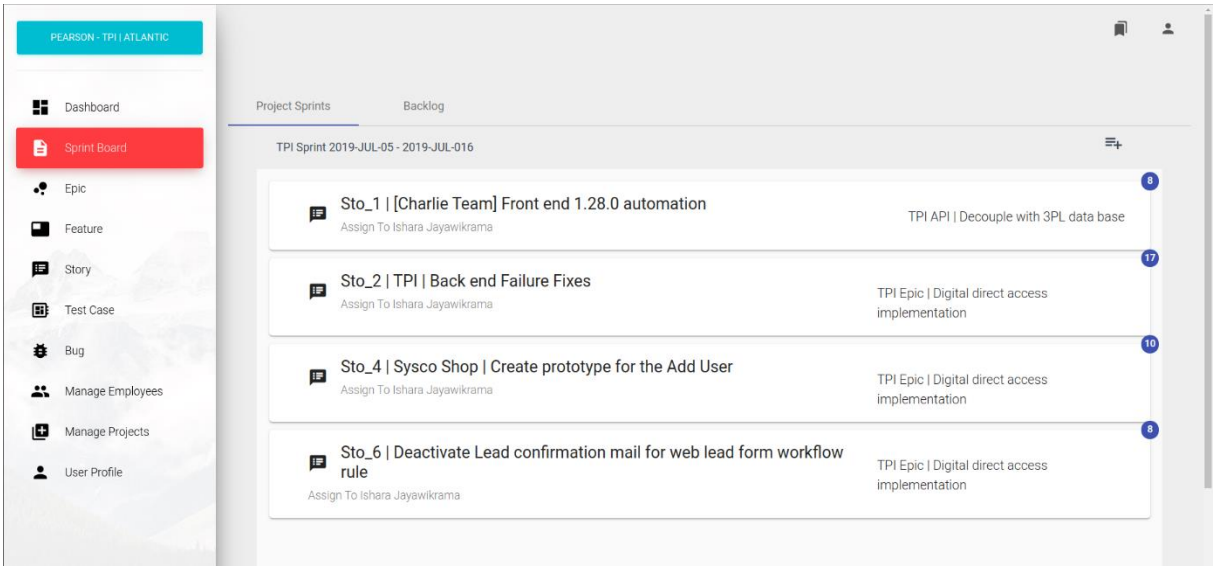

Figure 3.8 Sprint board of the SQA web application

3.6.4 User story creation page of the SQA web application

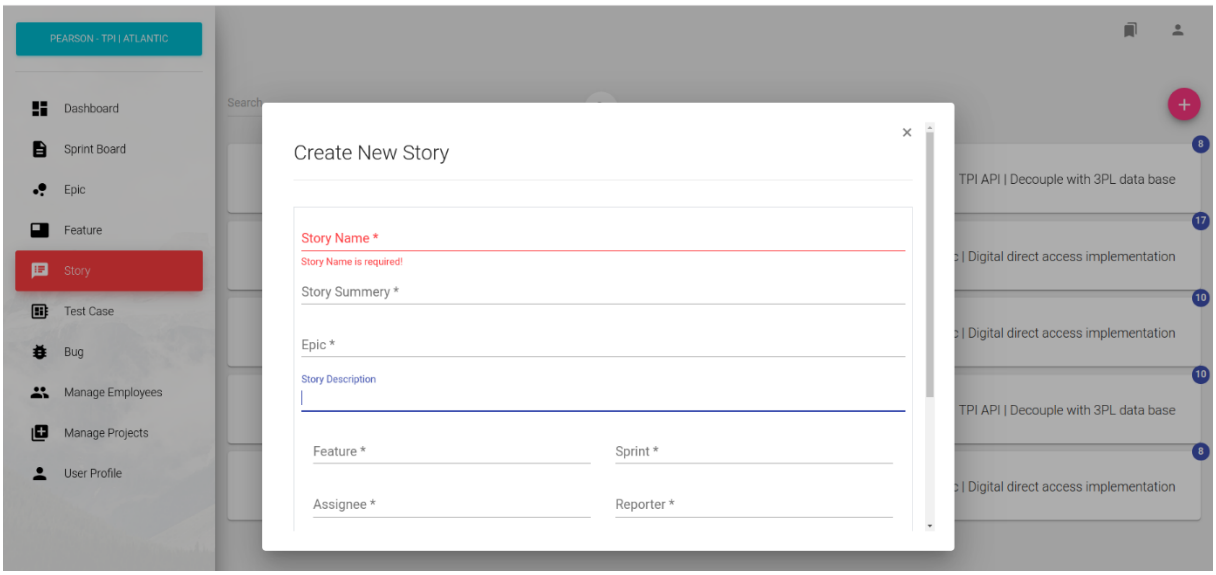

Figure 3.9 User story creation page

3.6.5 Test case view page of the SQA web application

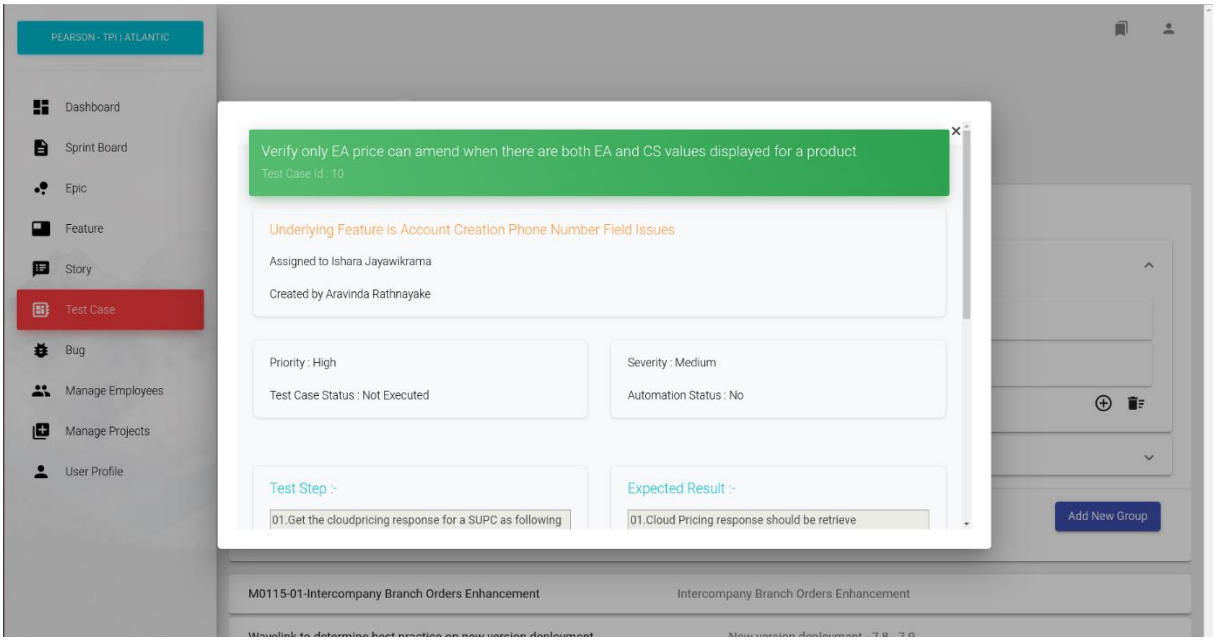

Figure 3.10 Test case view page

3.6.6 Bug creation and managing page of the SQA web application

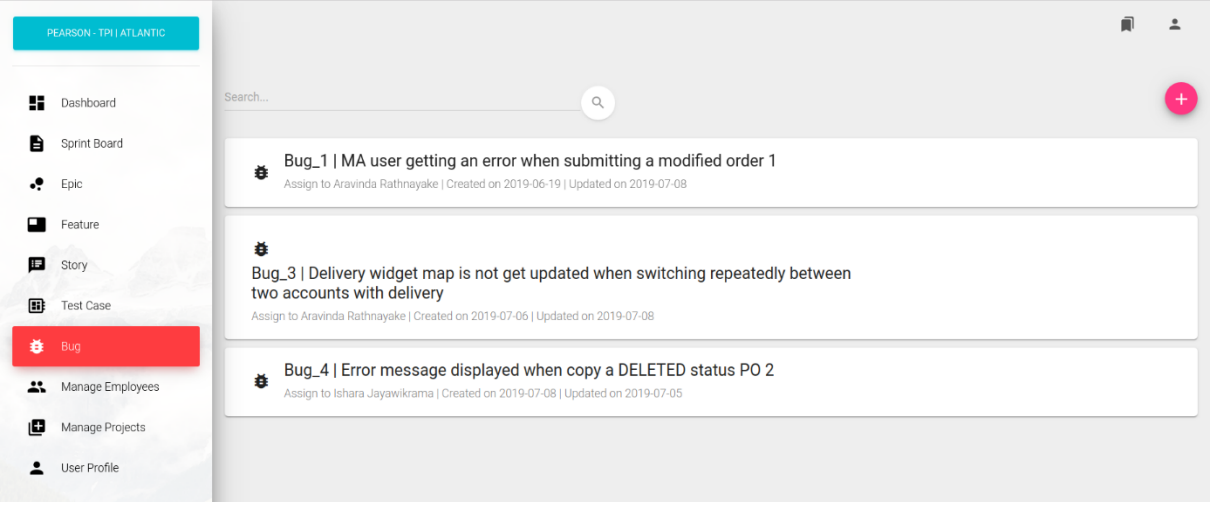

Figure 3.11 Bug creation and managing page.

3.7 Technology

- Java 1.8

Java web development platform is among the most required web development services. Java is an object-oriented language with a thoroughly worked through object model. It can be used to develop both standalone and web applications and suits both purposes equally well [15].

- STS (spring tool suit)

Spring Tools 4 is the next generation of spring tooling for your favorite coding environment. Largely rebuilt from scratch, it provides excellent support for developing Spring-based enterprise applications, whether you prefer Eclipse, Visual Studio Code, or Atom IDE [16].

- Nodejs

Node.js is an open-source, cross-platform JavaScript run-time environment that executes JavaScript code outside of a browser. Node.js lets developers use JavaScript to write command line tools and for server-side scripting—running scripts server-side to produce dynamic web page content before the page is sent to the user's web browser [17].

- Angular/ Angular CLI

The Angular CLI is a command-line interface tool that you use to initialize, develop, scaffold, and maintain Angular applications. You can use the tool directly in a command shell, or indirectly through an interactive UI such as Angular Console [18].

- Visual Studio Code

Visual Studio Code is a source-code editor developed by Microsoft for Windows, Linux and macOS. It includes support for debugging, embedded Git control and GitHub, syntax highlighting, intelligent code completion, snippets, and code refactoring. It is highly customizable, allowing users to change the theme, keyboard shortcuts, preferences, and install extensions that add additional functionality [19].

3.8 Summary

This chapter describes about the analyzing and design of the system. The system is a web application and it provides a smooth flow for Software SQA engineers in a Software Company to do their daily tasks.

Chapter 4

EVALUATION

4.1 Introduction

The main purpose of the project is to provide a better flow for Software Quality Assurance Process and project management process where both SQA engineers and Developers of the software industry such as QA Leads, Project Managers, Team members and Top-Level managers can work smoothly and accurately on a day-to-day basis.

This chapter is dedicated to discuss on the evaluation of project with respect to the objectives as well as the data that is being used as assistance to the project. In this chapter the testing strategies and test scenarios are mentioned. Also, it shows that how the developed system is accepted by the user.

Evaluation is a collection of activities and functions used to monitor and control a software project so that specific objectives are achieved with the desired level of confidence. A formal definition of software testing is “the systematic activities providing evidence of the fitness for use of the total software product”.

Agile development methodology is the process of establishing requirements, designing, building and testing a system in a series of short development cycles. Since we use Agile methodology there is a testing activity for every development activity.

4.2 Evaluation at Different Levels

The evaluation of the solution has been carefully planned and categorically assessed. It was done with the help of an evaluation plan. Evaluation plan address all the stages of the development life cycle of the project. All necessary activities were broken down and the primary decisions were taken place considering the project at very first stage by submitting written project proposal, requirements feasibility and clearness of problem domain, appropriateness of technologies and adequate resources.

- Requirement analysis

The main evaluation of the system begins at the first phase of the project with the start of gathering data that is related to the domain. After requirement gathering requirements are analyzed and the possibility of incorporating the requirements in the system to be development is also studied. Finally, a Requirement Specification document is created which serves the purpose of guideline for the next phase of the model.
Then the Test Planning was started and acceptance test design was conducted after the requirements analysis is completed.

Verification activities: Requirements reviews.
Validation activities: Creation of user acceptance test cases
Test Basis for acceptance test design: Use cases, User requirements.

- System Design

Functional test cases, component test cases and Integration test cases are developed based on the system design. Doing this at an early stage provides more time for the actual test execution later.
The component tests and integration tests are an essential part of any development process and helps eliminate the maximum faults and errors at a very early stage. Component tests can be designed at this stage based on the internal module designs.
Verification activities: Design reviews.
Validation activities: Creation and review of test cases.

- Testing

Throughout the development lifecycle of this project, a series of system tests and validations were done in order to ensure the quality of the final system. System testing is directly associated with the system design phase. System tests check the entire system functionality during the implementation of the system each module was tested individually. After completing integration of modules, the whole system was tested as a whole.

4.3 User Acceptance Testing

Acceptance testing is basically related to the business requirements testing. Here testing is done to validate that the business requirements are met in the user environment and tests are based on test cases developed in early stages.

User acceptance testing plays a major role in evaluation of a product. The user acceptance testing for this project was done mainly by providing a prototype of the product for few users and getting their feedbacks.

The implemented system targets a specific user category. It is a product for the people who are working in the Software Industry. However, the importance of this type of a system may not be understandable at once for a person with lesser technical knowledge. Therefore, for this user acceptance test, group of people with some technological knowledge background about using systems were selected. Since it's difficult to find a user base that are all Non-technical and familiar with systems, system was tested among both the people who are technically expertise as well has people who are Non-Technical.

A questionnaire was prepared in order to evaluate the implemented system. Following characteristics were targeted in the given questionnaire.

- Usability of the system - For how extent the Engineers can use the system and find it useful.

  The level of ease to learn and what kind of problems they face.
- Reliability of the system - The extent to which the system is expected to perform its intended functionality.
- Accuracy of the system - To which extent the response of the system is correct and data provided and retrieved through the system is accurate.
- Effectiveness of the system - Time taking to provide a result to the end users and how efficient and effective the system it. Less waiting time or had to wait more than they expected.
- Importance of the system to the users - The extent to which this system is useful to the society.

- Final evaluation of the system as a whole - Overall idea of users about the process and workflow of the application.

According to the results of evaluation it was possible to identify areas where further improvements are needed. Results show that the accuracy of the system and reliability has to be improved. Also, it is noticeable that many users have decided that Software Quality Assurance Process Management Application is an important concept and such a system is important to the society.

4.4 Test Scope

Employee view.

Test cases were developed based on the below use cases and scenarios.

- User can login to the system.

  Test Scenarios:
  Verify whether the Admin user and Employ can login to the system.
  Verify error messages displayed for the user who has entered incorrect credentials.
  Verify UI elements are properly aligned in login page.

- User will have a vertical navigation panel.

  Test Scenarios:
  Verify whether the user can navigate to different pages using left navigation.
  Verify only admin can navigate to the admin panel through left navigation.
  Verify UI elements are properly aligned in left navigation panel.

- Employee can create Test Plans.

  Test Scenarios:

  Verify user is able to create test plan in Test page.

- Employee can organize test cases by adding test cases to test suites.

  Test Scenarios:

  Verify whether the user is able to create test suite.

  Verify whether the user can add test cases to test suite.

- Employee can create test cases to each feature.

  Test Scenarios:

  Verify user can create test cases.

  Verify whether the mandatory fields are available when creation a test case.

  Verify whether the user can attach test cases to any feature.

- Employee can add test steps to each test case.

  Test Scenarios:

  Verify whether the user can add test steps to a test cases.

- Employee can copy test cases to different suites.

  Test Scenarios:

  Verify whether the user can add same test case to a different suite.

- Employee can execute test cases.

  Test Scenarios:

  Verify whether the user can open test case and see the test steps.

  Verify whether the user can change the status of a test case.

  Ex: Not executed to pass, Fail to Pass, etc.

- Employee can create bugs to each test cases.

  Test Scenarios:

  Verify whether the user is able to create a Bug if status of a test case is fail.

  Verify all fields are available in Bug creation page.

  Verify created bug can be opened.

  Verify Employee can update status of a defect according to defect lifecycle.

- Employee can create Epics.

  Test Scenarios:

  Verify whether the Employee can create Epics.

- Employee can create Feature list against the Epics.

  Test Scenarios:

  Verify whether the Employee can create Feature lists.

  Verify whether the user can attach Features to an Epic.

- Employee will see assigned Agile boards.

  Test Scenarios:

  Verify whether the Employee can navigate to the Agile board through left navigation.

  Verify whether the user can see assigned Agile board.

- Employee can go in to the Agile board and create backlog items.

  Test Scenarios:

  Verify whether the user can create backlog items.

  Verify all mandatory fields are available when creating a backlog.

- Employee can add backlog items to each sprint.

  Test Scenarios:

  Verify whether the user can add backlog items to a spring.

  Verify whether the user can change the sprint of a backlog item.

- Employee can create sprint to each Agile board.

  Test Scenarios:

  Verify whether the user can create a sprint.

- Employee will see burn down chart.

  Test Scenarios:

  Verify whether the user can see the burn down chart for a selected sprint.

  Verify UI elements are properly aligned.

- Admin and Employee can view dashboard.

  Test Scenarios:

  Verify whether the users have access to dashboard.

  Verify UI elements are properly aligned.

Admin view.

- Admin has an Agile board list.

  Test Scenarios:

  Verify whether the admin users can see access to dashboard.

  Verify UI elements are properly aligned.

- Admin has an Agile board list.

  Test Scenarios:

  Verify whether the admin users can Agile board list.

- Admin can assign employees to each Agile board.

  Test Scenarios:

  Verify whether the admin users can assign each Employee to an Agile board.

- Admin can create an employee.

  Test Scenarios:

  Verify whether the admin users can create employee accounts.

  Verify whether admin user can manage employee list.

  Admin can view employee list.

Above Test Scenarios are the most important sections of the application. Since these needed to be identified accurately and given the output to the users in an insignificant amount of time.

4.4.1 Risk mitigation checklist

Below table 4.1 consists of the risk mitigation checklist that done to conduct a system evaluation

Table 4.1Risk mitigation checklist

Test Scope	**Passed? (Yes/No)**
Has the system been tested with a fresh database?	Yes
Has the set up checked in a fresh machine?	Yes
Have the following been done?	
Creation of test cases	Yes
Environment compatibility testing as requested	Yes
System Integration Testing	Yes
Performance Testing	Yes
Typos and spell check	
UI	Yes
Messages	Yes
Alerts	Yes
Usability testing / Consistency testing	
UI Consistency (as per the UI guideline)	Yes
Use of mandatory fields (Mandatory field validation and control coloring	Yes
Exceptional data handling (Data inputs with special characters and symbols	Yes
Input field types and lengths validations (Check for celling limits)	Yes
Navigation (Tab Options, Unwanted user clicks)	Yes

4.5 Types of Evaluation

There are different types of evaluations methods depending on how the system is evaluated and the purpose of evaluation. A user-based evaluation is considered in evaluating the system, and to accomplish that, the progress of the workflow process of the organization is being continuously monitored.

- What is evaluated?

How accurate the data retrieval is when there are query based scenarios to retrieve data. How the manual process is improved with the proper approach and how the quality of life of the people who are employed in the QA Industry has improved with the use of the developed system.

- What is the purpose of the evaluation?

The purpose of the evaluation is ultimately to achieve the project goal and the objectives. People who are working for the QA Industry should be able to overcome the difficulties in the manual process. Therefore, it should be evaluated whether the goals are achieved properly or not.

- Who is interested in the evaluation?

Interested parties of the evaluation process are basically the people who are the end users. If the project is successfully achieving the objectives and the goal, then this project can be extended to deliver.

- How will they use the findings?

People who are working for the QA Engineering Process can use the findings of the evaluation result in identifying the process. How much of data they can retrieve and how efficient and accurate the new process is.

What questions should be answered?

- What are the processes in the project management and QA management that are automated?
- Are the results regarding test case executions accurate?
- As the process become more efficient and effective?
- What is the level of user friendliness in the system?
- What kinds of enhancements should be applied and how the approach should be taken?

4.5.1 Tested Devices

The system can be installed into Windows machines all the devices have the ability to install the developed application. Implemented system has been tested in various Environments. Tested devices shown in below table 4.2

Table 4.2 Tested devices

Tested Device	Operating System
HP – i3 – Pro book	Windows 7
HP – i3 – Pro book	Windows 8
DELL – Inspiron	Windows 7
DELL – Inspiron	Windows 10

Cross Browser Testing was conducted in the below Browser Versions:

Google Chrome - Version 71.0.3578.98 (Official Build) (64-bit)

Mozilla Firefox - Version 63.0.3 (64-bit)

Internet Explorer - Version 11.0.9600.19180

4.6 Summary

In this chapter all the evaluation criteria of the system along with the results gained by evaluations are presented. The summary of feedbacks from questionnaire is presented as a chart. Finally, it describes areas to be improved based on the results. Next chapter will be focused on conclusion of the system based on the achieved objectives, problems encountered and limitations of the system. Also, further expandability of the system is discussed.

Chapter 5

CONCLUSION

5.1 Introduction

The main purpose of the project is to provide a better flow for a software quality engineering process which includes writing test cases, executing test cases with results, creating test plans, creating relevant test suites etc. The users who intended to use this software are SQA engineers, developers, management and business analysts. This is aimed at simplifying many day to day activities of SQA engineers of a software company This project was successful in implementing a web based system to manage software quality engineering process which includes writing test cases, executing test cases with results, creating test plans, creating relevant test suites etc. The users who intended to use this software are SQA engineers, developers.

Various aspects of the project are being discussed throughout the whole dissertation. A full description about the problem and the developed solution for that problem, the final product as well as the technologies being used were discussed. The design of the project, its implementation, and the evaluation was the focus in last few chapters.

This chapter will mainly conclude the dissertation. It is dedicated to provide a description about the overall achievement of the project, the objectives achieved, and problems encountered throughout the project under the scope and time and actions taken to overcome those problems. Finally, how the project can be further developed in order to eliminate the problems encountered, the future extendibility is discussed.

5.2 Achieving the Objectives of the Project

In order to achieve the objectives a successful software project management application has to provide features to create test cases and execute test cases, check the overall status of test execution, to categorize test cases, create a bug when there is a failure in test cases, create Epics, create features, create user stories, create tasks, see the agile board, time reporting, see burn down chart and a user-friendly interface for users for the easy access is very important.

The below objectives were achieved from an engineer's point of view

- The user is able to manage, organize and track all testing efforts and keep all user stories in a central place.
- Easily accessible test artifacts with entire team and organize test suites and test plans.
- Execute tests and record results using application's modern user interface.
- Summary dashboard for projects, bugs, test plans and runs.
- Using the application, track time for each task that how much time has been spent on the particular task from start to end.
- Enable team to track tasks and visualize process so team knows how projects and tasks are progressing.

For the individual companies:

- Keep better schedules
- Increased productivity
- Better visibility

5.3 Problems Encountered in the Project

Technological Challenges faced:

Spring Boot (via Maven), and frontend code, in the project, have been very integrated. Maven is expecting a certain directory structure that has been generally incompatible with front-end code. This forced me into moving the front-end project down in **src/main/resources** and make configuration changes that could be haphazard and unreliable as compared to a native, Nodejs based front-end project.

During the time of development, the database changes were identified. It was very difficult task for me to change the database structure due to ambiguous requirements.

Due to lack of knowledge in AngularJS I had to find solutions for below questions.

How to post a json to a service?

How to route the components?

How to access the reactive form data from the type script file?

How the “Two-way” data binding in angular works?

What are lifecycle hooks and basic usage?

How to interact between Parent and Child components?

How to create a service in angular?

5.4 Limitations of the Currently Developed Solution

- The system was implemented as an assist for the employees of a Software company. A main limitation was that a mobile application could not be made.
- Need to be able to import text or CSV files for test cases, test plans, test lab, and Defects.t
- Need to be able to copy or move items from the test suite to other test suites.
- Need to be able to select multiple test cases and attach them to the defect. Currently the user can only attach one.
- Quarry based search not supported.
- Next limitation is that there is no way to interact with other systems. (APIs)

Even though there can be limitations of the currently developed system, they can be addressed and eliminated by taking necessary actions and further developing the system.

5.5 Further extendibility of the system

As the further development of the system, we mainly focus on how to overcome the limitations that are were presented in the previous section. There were two major limitations discussed there and those have to be addressed one by one to provide a better system through further development.

The following can be increased with a mobile app:

1. Easy Time tracking
2. Better visibility for higher management (Dashboard feature)
3. Better visibility for the team (Burn down chart)

It is important to provide APIs to interact with the application. Thus, users can upload test cases and transfer user stories from their existing systems.

A feature can be implemented to import and export data from the application. Eg: test cases, bugs, etc.

Furthermore a search feature could be implemented to retrieve data from the application using complex queries.

Drag and Drop feature could also be implemented to copy or move items.

References

[1] Tutorials Point India Limited, "SQA Components," 2019. [Online]. Available: https://www.tutorialspoint.com/software_quality_management/software_quality_management_sqa_components.htm.

[2] Software Testing Class, "Software Testing Life Cycle (STLC)," 2019. [Online]. Available: https://www.softwaretestingclass.com/software-testing-life-cycle-stlc/.

[3] Trust Radius, "Micro Focus Application Lifecycle Management / Quality Center (ALM/QC)," 2019. [Online]. Available: https://www.trustradius.com/compare-products/micro-focus-application-lifecycle-management-quality-center-alm-qc-vs-wrike.

[4] Wikimedia Foundation, Inc., "HP Application Lifecycle Management," 2019. [Online]. Available: https://en.wikipedia.org/wiki/HP_Application_Lifecycle_Management.

[5] Guru99 Tech Pvt Ltd, "Create Releases & Cycles in HP ALM (Quality Center)," 2019. [Online]. Available: https://www.guru99.com/hp-alm-release-specifications.html.

[6] Wikimedia Foundation, Inc., "Team Foundation Server," 2019. [Online]. Available: https://en.wikipedia.org/wiki/Team_Foundation_Server.

[7] Microsoft Corporation, "Add, update, and follow a work item," 2019. [Online]. Available: https://docs.microsoft.com/en-us/azure/devops/boards/backlogs/add-work-items.

[8] Microsoft Corporation, "Monitor sprint burndown," 2019. [Online]. Available: https://docs.microsoft.com/en-us/azure/devops/boards/sprints/sprint-burndown?view=azure-devops.

[9] Microsoft Corporation, "Scrum and sprint planning tools," 2019. [Online]. Available: https://docs.microsoft.com/en-us/azure/devops/boards/sprints/scrum-sprint-planning-tools?view=azure-devops.

[10] Microsoft Corporation, "Create manual test cases," 2019. [Online]. Available: https://docs.microsoft.com/en-us/azure/devops/test/create-test-cases?view=azure-devops.

[11] Microsoft Corporation, "Define, triage, and manage bugs," 2019. [Online]. Available: https://docs.microsoft.com/en-us/azure/devops/boards/backlogs/manage-bugs?view=azure-devops.

[12] Wikimedia Foundation, Inc., "Jira (software)," 2019. [Online]. Available: https://en.wikipedia.org/wiki/Jira_(software).

[13] Guru99 Tech Pvt Ltd, "Best 25 Test Management Tools in 2019," 2019. [Online]. Available: https://www.guru99.com/top-20-test-management-tools.html.

[14] Microsoft Corporation, "Track work with user stories, issues, bugs, features, and epics," 2019. [Online]. Available: https://docs.microsoft.com/en-us/azure/devops/boards/work-items/about-work-items?view=azure-devops.
[15] Magic web solutions, "Java as a web development platform," 2019. [Online]. Available: https://www.magicwebsolutions.co.uk/blog/java-as-a-web-development-platform.htm.
[16] Pivotal Software, Inc., "Spring Tool," 2019. [Online]. Available: https://spring.io/tools.
[17] Wikimedia Foundation, Inc., "Node.js," 2019. [Online]. Available: https://en.wikipedia.org/wiki/Node.js.
[18] Google inc., "Angular CLI Overview and Command Reference," 2019. [Online]. Available: https://angular.io/cli.
[19] Medium, "Cool Extensions to Make Programming Life Easier," 2019. [Online]. Available: https://medium.com/@micaonthego/vs-code-cool-extensions-to-make-programming-life-easier-48bb428d45c3.
[20] Amazon Web Services, Inc., "DevOps and AWS," 2019. [Online]. Available: https://aws.amazon.com/devops/.
[21] Wikimedia Foundation, Inc., "Burn down chart," 2019. [Online]. Available: https://en.wikipedia.org/wiki/Burn_down_chart.
[22] Linchpin SEO, "Agile Process and Method Overview," 2019. [Online]. Available: https://linchpinseo.com/the-agile-method/.
[23] Nulab, Inc., "ER diagrams vs. EER diagrams: what’s the difference?," 2019. [Online]. Available: https://cacoo.com/blog/er-diagrams-vs-eer-diagrams-whats-the-difference/.
[24] Wikimedia Foundation, Inc., "LoadRunner," 2019. [Online]. Available: https://en.wikipedia.org/wiki/LoadRunner.
[25] Microsoft Corporation, "Azure Test Plans | Azure DevOps Server 2019 | Run manual tests," 2019. [Online]. Available: https://docs.microsoft.com/en-us/azure/devops/test/run-manual-tests?view=azure-devops.

Glossary

Database - is an organized collection of data for one or more purposes, usually in digital form.

Graphical User Interface - is a type of user interface that allows users to interact with electronic devices with images rather than text commands.

Internet - is a global system of interconnected computer networks that use the standard

Java - Java is a programming language and computing platform

Objects Oriented Development - is a standard approach to software development based on objects and its instances

Structured Query Language - is a database computer declarative language designed for managing data in relational database management systems (RDBMS).

Unified Modeling Language (UML) - is a standardized general-purpose modeling language in the field of object-oriented engineering. This includes a set of graphic notations techniques to create visual models of object-oriented software-intensive systems.

DevOps - is the combination of cultural philosophies, practices, and tools that increases an organization's ability to deliver applications and services at high velocity: evolving and improving products at a faster pace than organizations using traditional software development and infrastructure management processes [20].

Burn down Chart - is a graphical representation of work left to do versus time. The outstanding work (or backlog) is often on the vertical axis, with time along the horizontal. That is, it is a run chart of outstanding work. It is useful for predicting when all of the work will be completed [21].

Agile – The Agile Method is a particular approach to project management that is utilized in software development. This method assists teams in responding to the unpredictability of constructing software. It uses incremental, iterative work sequences that are commonly known as sprints [22].

EER Diagram - Enhanced entity-relationship (EER) diagrams are basically an expanded upon version of ER diagrams. EER models are helpful tools for designing databases with high-level models. With their enhanced features, you can plan databases more thoroughly by delving into the properties and constraints with more precision [23].

LoadRunner - It is used to test applications, measuring system behavior and performance under load. LoadRunner can simulate thousands of users concurrently using application software, recording and later analyzing the performance of key components of the application [24].

Appendix

End user feedback form:

Job Title:

……..

Ratings:

1 – Strongly disagree 2 – Disagree 3 – Possibly

4 – Agree 5 – Strongly agree

What do you think about the following features of the system?

Rate them according the scale given above.

	Option	Rate
1.	Usability of the system is good	
2.	User friendliness of the system is high	
3.	Look and feel of the system throughout all the forms is consistent	
4.	Instructions given in the system are understandable	
5.	Error messages are informative and understandable	
6.	Menus, forms and navigation methods are uniform and consistent	
7.	Navigation through the system is intuitive and easily understandable	
8.	Entering data in to forms can be carried out easily	
9.	Finding the needed information can be done without difficulty	
10.	Reports, attendance reviews and other transactions can be done easily	
11.	Adequate information is provided in the reports	
12.	The required functionalities have been implemented in the system	
13.	There is a marked improvement in the system compared to the old manual system	
14.	The new system is beneficial to the company	