A

DISSERTATION

"OA EYE"

SUBMITTED BY

AMOL V. CHOUDHARI

GUIDED BY

PROF. (DR.) S. A. KULKARNI

PUNE INSTITUTE OF COMPUTER TECHNOLOGY PUNE

UNIVERSITY OF PUNE

❁ 2005-06 ❁

A

Dissertation on

"QA Eye"

Submitted By:

AMOL V. CHOUDHARI

Guided By:

Prof. (Dr.) S.A.Kulkarni
H.O.D., M.E. Computer Engineering,
PICT, Pune.

Pune Institute of Computer Technology, Pune
University of Pune
(Year 2005 -2006)

PUNE INSTITUTE OF COMPUTER TECHNOLOGY

(College of Engineering Affiliated to the University of Pune & Accredited by NBA)
Sr. No. 27, Satara Road, Dhankavadi, Pune - 411 043 (INDIA)
ID No. : PU/PN/Engg./054(1983)

Phone No. 24371101 / 24378063 Fax No. : (020) 24372106
Website : picsctr.edu E-mail : principal@pictsctr.edu

Ref.: PICT/

Date 29/05/2006.

CERTIFICATE

This is to certify that the project

Entitled

"QA Eye"

Is being submitted by Mr. Amol Vilasrao Choudhari in partial fulfillment for the award of the degree of Master of Engineering in Computer Engineering to University of Pune, is a record of work carried out by his under our supervision and guidance.

PROF.(Dr.) **S.A.KULKARNI**
(PROJECT GUIDE)
(HEAD OF ME COMP.ENGG.)
PUNE INSTITUTE OF COMPUTER TECHNOLOGY,
PUNE-411043

(PRINCIPAL)
P.I.C.T., PUNE-411043

Principal
Pune Institute of Computer Technology

Pune Institute of Computer Technology Pune-43

Cybage
e-volving solutions

March 24, 2006

CERTIFICATE

TO WHOMSOEVER IT MAY CONCERN

This is to certify that **Mr. Amol Vilasrao Choudhari.**, student of second year of Masters of Computer Engineering (**M.E.Computers**) from Pune Institute of Computer Technology (**PICT**), under the University of Pune, is working with us on **QA Eye** tool from April 2005 till date.

During this period, he has undergone through all SDLC process and worked with following technologies.

"QA EYE"

Application development includes following technologies:

ASP.NET using C#, MS-SQL Server 2000

For Cybage Software Pvt. Ltd.

Elston Pimenta
Manager-HRD

Sandeep Pawar
Project Manager

Cybage Software Pvt. Ltd.
25/A-1, West Avenue,
Kalyani Nagar, Pune - 411 006.

ACKNWLEDGEMENTS

"You can learn more about a road by treading on It just once than by referring to a thousand maps about it ..."

Whoever said this also must have spend some time training at Cybage Software Pvt. Ltd. I have learnt that an organization of the caliber of this one, this period is in deed a very short amount of time.

In my endeavor to successful compete this project I sincerely thank **Mr. Sandeep Pawar** (QA Project Manager) of Cybage, who is also my external Project Guide, for the constant guidance and support he has extended during the Project Development period.

The continued assistance and the guidance of **Prof.(Dr.)S.A. Kulkarni** (Internal Guide). He is the key of success in developing this project.

Finally I take this opportunity to thank my colleagues and friends for their cheerful encouragement.

Amol Choudhari,
M.E. (Computer),
P.I.C.T. Pune.

INDEX

1.Introduction

1. Introduction

1.1. Background

This QA Eye tool is developed for QA management process. This whole product is developed on Microsoft .net platform. The reason for preferring this technology is discussed below in details.

1.2. Technical Basics

1.2.1. Microsoft .net Framework

The .NET Framework is an integral Windows component that supports building and running the next generation of applications and XML Web services. The .NET Framework is designed to fulfill the following objectives:

- To provide a consistent object-oriented programming environment whether object code is stored and executed locally, executed locally but Internet-distributed, or executed remotely.
- To provide a code-execution environment that minimizes software deployment and versioning conflicts.
- To provide a code-execution environment that promotes safe execution of code, including code created by an unknown or semi-trusted third party.
- To provide a code-execution environment that eliminates the performance problems of scripted or interpreted environments.
- To make the developer experience consistent across widely varying types of applications, such as Windows-based applications and Web-based applications.

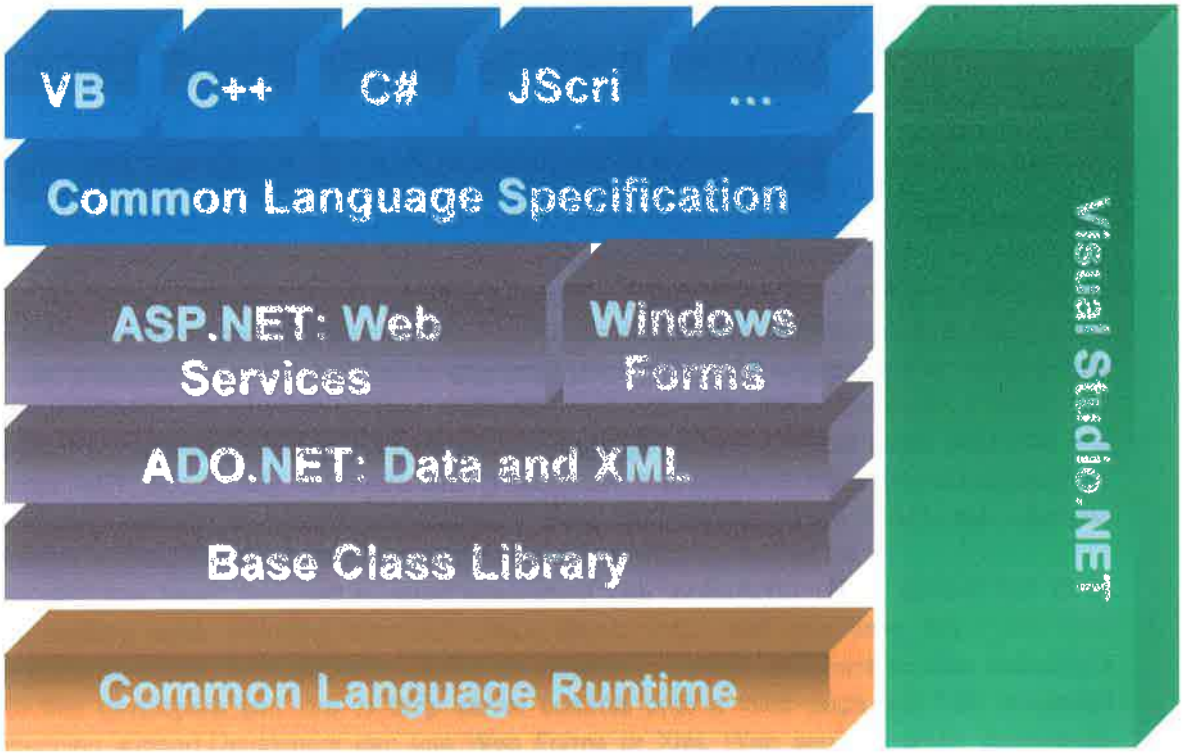


Fig 1.1: **.Net Framework**

To build all communication on industry standards to ensure that code based on the .NET Framework can integrate with any other code. The .NET Framework has two main components: the common language runtime and the .NET Framework class library. The common language runtime is the foundation of the .NET Framework. You can think of the runtime as an agent that manages code at execution time, providing core services such as memory management, thread management, and remoting, while also enforcing strict type safety and other forms of code accuracy that promote security and robustness. In fact, the concept of code management is a fundamental principle of the runtime. Code that targets the runtime is known as managed code, while code that does not target the runtime is known as unmanaged code. The class library, the other main component of the .NET Framework, is a comprehensive, object-oriented collection of reusable types that you can use to develop applications ranging from traditional command-line or graphical user interface (GUI) applications to applications based on the latest innovations provided by ASP.NET, such as Web Forms and XML Web services.

The .NET Framework can be hosted by unmanaged components that load the common language runtime into their processes and initiate the execution of managed code, thereby creating a software environment that can exploit both managed and unmanaged features. The .NET Framework not only provides several runtime hosts, but also supports the development of third-party runtime hosts.

For example, ASP.NET hosts the runtime to provide a scalable, server-side environment for managed code. ASP.NET works directly with the runtime to enable ASP.NET applications and XML Web services, both of which are discussed later in this topic.

Internet Explorer is an example of an unmanaged application that hosts the runtime (in the form of a MIME type extension). Using Internet Explorer to host the runtime enables you to embed managed components or Windows Forms controls in HTML documents. Hosting the runtime in this way makes managed mobile code (similar to Microsoft® ActiveX® controls) possible, but with significant improvements that only managed code can offer, such as semi-trusted execution and isolated file storage.

1.2.2. ASP.net

ASP.NET is more than the next version of Active Server Pages (ASP); it provides a unified Web development model that includes the services necessary for developers to build enterprise-class Web applications. While ASP.NET is largely syntax compatible with ASP, it also provides a new programming model and infrastructure for more scalable and stable applications that help provide greater protection. You can feel free to augment your existing ASP applications by incrementally adding ASP.NET functionality to them. ASP.NET is a compiled, .NET-based environment; you can author applications in any .NET compatible language, including Visual Basic .NET, C#, and JScript .NET. Additionally, the entire .NET Framework is available to any ASP.NET application. Developers can easily access the benefits of these technologies, which include the managed common language runtime environment, type safety, inheritance, and so on.

ASP.NET has been designed to work seamlessly with WYSIWYG HTML editors and other programming tools, including Microsoft Visual Studio .NET. Not only does this make Web development easier, but it also provides all the benefits that these tools have to offer, including a GUI that developers can use to drop server controls onto a Web page and fully integrated debugging support.Developers can use Web Forms or XML Web services when creating an ASP.NET application, or combine these in any way they see fit. Each is supported by the same

infrastructure that allows you to use authentication schemes, cache frequently used data, or customize your application's configuration, to name only a few possibilities.

1.2.3. C# Language

C# is an elegant and type-safe object-oriented language that enables developers to build a wide range of secure and robust applications that run on the .NET Framework. You can use C# to create traditional Windows client applications, XML Web services, distributed components, client-server applications, database applications, and much, much more. C# syntax is highly expressive, yet with less than 90 keywords, it is also simple and easy to learn. The curly-brace syntax of C# will be instantly recognizable to anyone familiar with C, C++ or Java. Developers who know any of these languages are typically able to begin working productively in C# within a very short time. C# syntax simplifies many of the complexities of C++ while providing powerful features such as null able value types, enumerations, delegates, anonymous methods and direct memory access, which are not found in Java. C# also supports generic methods and types, which provide increased type safety and performance, and iterators, which enable implementers of collection classes to define custom iteration behaviors that are simple to use by client code.
As an object-oriented language, C# supports the concepts of encapsulation, inheritance and polymorphism. All variables and methods, including the Main method, the application's entry point, are encapsulated within class definitions. A class may inherit directly from one parent class, but it may implement any number of interfaces. Methods that override virtual methods in a parent class require the override keyword as a way to avoid accidental redefinition. In C#, a struct is like a lightweight class; it is a stack-allocated type that can implement interfaces but does not support inheritance.

1.2.4. MS SQL Server

Microsoft SQL Server 2000 is a full-featured relational database management system (RDBMS) that offers a variety of administrative tools to ease the burdens of database development, maintenance and administration. In this article, we'll cover six of the more frequently used tools: Enterprise Manager, Query Analyzer, SQL Profiler, Service Manager, Data Transformation Services and Books Online. Let's take a brief look at each:
Enterprise Manager is the main administrative console for SQL Server installations. It provides you with a graphical "birds-eye" view of all of the SQL Server installations on your network. You can perform high-level administrative functions that affect one or more servers, schedule common maintenance tasks or create and modify the structure of individual databases.
Query Analyzer offers a quick and dirty method for performing queries against any of your SQL Server databases. It's a great way to quickly pull information out of a database in response to a user request, test queries before implementing them in other applications, create/modify stored procedures and execute administrative tasks
SQL Profiler provides a window into the inner workings of your database. You can monitor many different event types and observe database performance in real time. SQL Profiler allows you to capture and replay system "traces" that log various activities. It's a great tool for optimizing databases with performance issues or troubleshooting particular problems.
Service Manager is used to control the MSSQLServer (the main SQL Server process), MSDTC (Microsoft Distributed Transaction Coordinator) and SQLServerAgent processes. An icon for this service normally resides in the system tray of machines running SQL Server. You can use Service Manager to start, stop or pause any one of these services.

Data Transformation Services (DTS) provide an extremely flexible method for importing and exporting data between a Microsoft SQL Server installation and a large variety of other formats. The most commonly used DTS application is the "Import and Export Data" wizard found in the SQL Server program group.

1.2.5. AJAX, Java Script

The term Ajax is used to describe a set of technologies that allow browsers to provide users with a more natural browsing experience. Before Ajax, Web sites forced their users into the submit/wait/redisplay paradigm, where the users' actions were always synchronized with the server's "think time." Ajax provides the ability to communicate with the server asynchronously, thereby freeing the user experience from the request/response cycle. With Ajax, when a user clicks a button, you can use JavaScript and DHTML to immediately update the UI, and spawn an asynchronous request to the server to perform an update or query a database. When the request returns, you can then use JavaScript and CSS to update your UI accordingly without refreshing the entire page. Most importantly, users don't even know your code is communicating with the server: the Web site feels like it's instantly responding.
While the infrastructure needed by Ajax has been available for a while, it is only recently that the true power of asynchronous requests has been leveraged. The ability to have an extremely responsive Web site is exciting as it finally allows developers and designers to create "desktop-like" usability with the standard HTML/CSS/JavaScript stack.

By performing screen updates on the client, you have a great amount of flexibility when it comes to creating your Web site. Here are some ideas for what you can accomplish with Ajax:

- Dynamically update the totals on your shopping cart without forcing the user to click Update and wait for the server to resend the entire page.
- Increase site performance by reducing the amount of data downloaded from the server. For example, on Amazon's shopping cart page, when I update the quantity of an item in my basket, the entire page is reloaded, which forces 32K of data to be downloaded. If you use Ajax to calculate the new total, the server can respond with just the new total value, thereby reducing the required bandwidth 100 fold.
- Eliminate page refreshes every time there is user input. For example, if the user clicks Next on a paginated list, Ajax allows you to just refresh the list with the server data, instead of redrawing the entire page.
- Edit table data directly in place, without requiring the user to navigate to a new page to edit the data. With Ajax, when the user clicks Edit, you can redraw the static table into a table with editable contents. Once the user clicks Done, you can spawn an Ajax request to update the server, and redraw the table to have static, display-only data.

1.3. Project Overview

The primary goal of this document is to provide a complete and accurate list of requirements for the "Cybage QA Eye" project to be developed for Cybage Software Pvt. Ltd. The QA eye is Web-based test management tool. It helps user to organize and manage all phases of the application testing process, including specifying testing requirements, planning tests, executing tests, and generating reports.

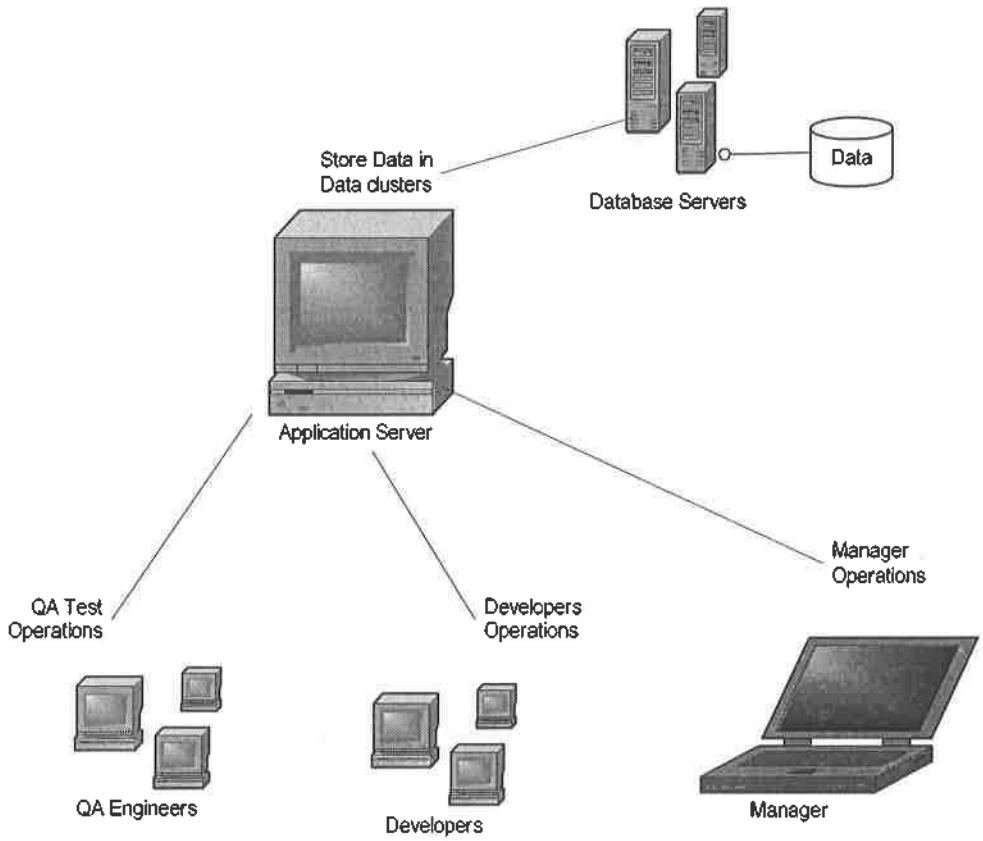


Fig 1.2: **Workflow of QA Eye**

As shown in above figure whole QA process is decentralized in an organization. Data is stored in various DB servers to reduce the complexity. QAs, Developers and Managers are given different privileges depending on their role in projects. This whole enterprise application is based on web application which is developed in Microsoft .net framework technology.

2. Project Plan and Risk Analysis

2. Project Plan and Risk Analysis

This plan will be used for managing **QA Eye** project to be developed for Client Cybage Software Pvt. Ltd.

2.1. Purpose:

2.1.1. Purpose of the Document

The purpose of this plan is to identify:

- All the project activities.
- The processes required for executing the project and tailoring required for the project, if any.
- Roles and responsibilities for the project execution and management activities.
- Manpower / hardware / software resources required.
- Parameters used for project planning and monitoring mechanisms.
- Communication channel with Client / Team Member.
- Training plan required for the project.
- Risks and plan for mitigating these risks.
- The project data, which will be used for measuring the project performance.

It is planning and tracking document that forms the basis of project execution.

2.1.2. Objectives of SPMP-D

The objectives of SPMP-D are:

1. To draw a complete project execution plan.
2. To describe the schedule not adherence.
3. To establish smooth communication channel for notifying/ informing all parties involved in executing the project.
4. To establish quantitative and statistical project objectives
5. To define sustained level of high quality input / output through reviews and audits.
6. To outline risks in executing the project and to prepare a risk mitigation strategy and plan to address those risks.
7. To define Change Management Process.

2.2. Project Objectives

Project objectives are defined below to achieve the project's established quality and process-performance objectives. These objectives are monitored quantitatively and necessary corrective actions are taken if a deviation from the objectives is observed.

2.2.1. Quantitative Project Objectives

Following are the quantitative objectives established for the QA Eye project to make the project's performance more **predictable**.

Data points which are beyond the control limit will trigger Corrective Action which may include Corrective Analysis and Resolution Process (CAR) and this will be discussed in the meeting which will be conducted immediately after Project Status Report (PSR) preparation.

Sr. No.	Parameters	Project Phase in which the parameters will be monitored/ In which they are applicable	Project/ Process Objective along with control limits	Base Measurement	Data Source	Priority and Action Plan to achieve the project objectives
1.	No. of defects logged/unit time (month)	Construction and Testing	Overall: UCL: 50 LCL: 10 Phase-wise: <Construction: UCL: 10 LCL: 3	No. of defects logged, period of construction/ testing	– Test Summary report – Project Status Report	– Priority: 1 Unit test cases should be written.
2.	Time spent in reviews/week	All phases	Overall: UCL: 4 man-hours LCL:1 man-hour Phase-wise: <Requirement: UCL: 1 LCL:1	Time spent in reviews during period	– Review Reports – Review Summary Report	– Priority: 2 Review of each operation .

2.2.2. Statistical Project Objectives

Following are the statistical objectives established for the QA Eye project to determine the **capability** to achieve the project's objective.

- Requirements volatility.
- Ratios of estimated values of the planning parameters (e.g. size, cost, and schedule).
- Coverage and efficiency.
- Effectiveness of training.
- Percentage of total defects inserted or found in the different phases of the project life cycle.
- Percentage of the total effort expended in the different phases of the project life cycle.

Sr. No.	Parameters	Project Phase in which the parameters will be monitored/ In which they are applicable	Project/ Process Objective along with control limits	Organizations Objectives (Baselines)	Base Measurement	Data Source
1.	Schedule Variance	All phases	Overall: UCL: 2.5 LCL: 0.0	**UCL: 3.0** **LCL: 0.0**	Planned schedule Vs Actual schedule	– Project Status Report – Weekly Status Report
			Phase-wise: UCL: 2.5 LCL: 0.0			
2.	Effort Variance	Construction and Testing	Overall: UCL: 7.5 LCL: 0.0	**UCL: 8.0** **LCL: 0.0**	Planned effort Vs Actual effort	– Project Status Report – Weekly Status Report
			Phase-wise: UCL: 2.5 LCL: 0.0			
3.	Defect Density	All phases except requirement phase	Overall: UCL: 0.70 LCL: 0.0	**UCL: 0.75** **LCL: 0.30** **(For screen-based defect density)** **UCL: 6.00** **LCL: 0.50** **(For KLOC-based defect density)**	Total number of defect> <Size of code in terms of number of screens, number of lines, number of function points etc.	– TTPro/ Bugzilla etc. – Project Closure Report
			Phase-wise: UCL: 2.5 LCL: 0.0			

Table 2.1: Project Objectives

Project data is collected and analyzed in Project Status Report (PSR), which is sent to the Delivery Head. Project manager in consultation with the Delivery Head will take decision for corrective action if there is significant deviation in planned Vs actual.

2.3. Processes used for the project

2.3.1. Project Phases

The project is of development type and is following Modified Waterfall model life cycle.

Various Phases identified in the project are as follows:

For each phase identify the Entry, Task, Verification / Validation, Exit (ETVX) criteria's.

- Event that triggers the process (Entry criteria).
- Task and activities performed in the phase.
- Conducting Verification activities like reviews & audits or signoff from the Client.
- Event that logically ends the process (Exit criteria).
- Responsibilities of main stakeholders for each phase executed. Record responsibilities with respect to project management and horizontal processes at the end of the table.

Sr. No.	Phase	Entry Criteria	Verification Activities	Exit Criteria	Stakeholder& Responsibility of main Stakeholders
1.	Requirements Analysis	SOW from Client	SRS Review by Client	SRS created and Signoff by Client	Client- Provide timely inputs to clarify requirements understanding. Review and approval of SRSProject Manager/ Project Leader- Obtain understanding of requirements before freezing them, Assessing feasibility and risks Obtain agreement from client as well as project team. Prepare SRS Manage requirement changes, Review requirements traceability **Design Group-** understand requirements before developing design
2.	Design	SRS/ PRD	HLD/ LLD Review by Client	HLD/ LLD created and signoff by Client	**Client-** review and approve HLD and LLD **Project Manager-** Define design methodology and standards before commencing design Review & approve the design documents

Sr. No.	Phase	Entry Criteria	Verification Activities	Exit Criteria	Stakeholder& Responsibility of main Stakeholders
					Design Group- develop & document high-level design and low-level design **Developer-** understand design before developing code.
3.	Construction	Design Document	Implemented Code Review by Client	Code and Database schema created and signoff by Client	**Project Leader-** set up development environment **Developer-** develop and verify code as per design **Build Engineer-** Conduct Smoke testing before creating build
4.	Testing	SRS/Test Plan	Test cases Review by Client	Test cases created and signoff by Client	**QA Manager-** Set up test environment, Validate and accept build before testing **QA Engineer-** Perform testing, report defects **Developer-** Resolve defects Project Leader- List known defects.
5.	Release	TCD/TSR	Integration of system Review by client	System is created and signoff by client.	**Project Leader-** Define sequence of integration, Create user documents **Build Engineer-** Create and verify build **Client-** Test and accept the release

Sr. No.	Phase	Entry Criteria	Verification Activities	Exit Criteria	Stakeholder& Responsibility of main Stakeholders
6.	All phases	Project Management			**Project Manager/ Project Leader-** Initiate project, establish estimates, define project scope, define process applicability and tailoring, assign responsibilities, establish project plans, identify and track risks, track actual progress against planned, resolve deviations & issues, revise project plan, schedule, objectives appropriately, conduct status review periodically and at milestone closures, close project.
	All phases	Resource Management			**Project Manager-** facilitate smooth project execution when there is resource change in Project team
	All phases	Configuration Management			**Project Manager-** Identify CCB to analyze and approve change requests **Configuration Controller-** establish and manage configuration system for project
	All phases	Metrics & Measurement			**Project manager-** establish project-level improvement goals & baselines, use organization –level models, analyze actual data against goals, take corrective action to correct deviations, if any **SQA-** Collection of project-level data **Metrics Analyst-** compiles project-level data to arrive at organization level baselines and models.

Sr. No.	Phase	Entry Criteria	Verification Activities	Exit Criteria	Stakeholder& Responsibility of main Stakeholders
	All phases	Reviews			**Quality Lead**- Select work products for review, define review criteria, plan reviews **Reviewer**- Conduct reviews, communicate results **Project Leader**- plan and track corrective actions **Reviewee**- Take corrective actions

Table 2.2: **Phases of Projects**

2.3.2. Processes used

The following table lists out various processes used in the projects.

Sr. No.	Process and corresponding Templates to be Used	Applicability Yes/ No If 'No' Why?	Tailoring Required/ Alternative practice (If any) and Why?
	Project Management Activities		
1.	**Project Kick Off Process**	Yes	
	Project Kick Off Document	Yes	
	Project By Net - Request Form	Yes	
	Statement of Work	Yes	
2.	**Estimation Process**	Yes	
	Task Based Estimation	Yes	
	Function Point Estimation	Yes	
	Detailed Estimation	Yes	
	Project Closure Estimation Analysis	No	Client supplied format will be used.
3.	**Risk Management Process**	Yes	
	Risk Monitoring Document- PBN	Yes	
4.	**Project Planning & Monitoring Process**	Yes	
	Software Project Management Plan-Development	Yes	
	Project Status Report	Yes	
	Weekly Status Report	Yes	
	Disaster Recovery Plan	No	Not specified by Client
	SPC Document	Yes	
	Weekly QA Status Report	Yes	
	Client Feedback Form	Yes	
5.	**Change Management Process**	Yes	
	Change Request Log-PBN	Yes	Remedy tracker will be used for tracking change requests
	Impact Analysis Form-PBN	Yes	
6.	**Causal Analysis and Resolution Process**		
	Causal Analysis Form	Yes	
7.	**Project Closure Process**	Yes	
	Project Closure Report	Yes	
	Project Sign Off Report	Yes	

SDLC Activities			
1.	**Requirements Management Process**	Yes	
	Requirement Understanding Document	Yes	
	System Requirement Specification	Yes	This will be done through e-mails since it's a small project
	Requirement Traceability Matrix	Yes	We will be using Client supplied format, as per Client's request
2.	**Design Process**	Yes	
	High Level Design	Yes	
	Low Level Design/ Design Document	Yes	Client Supplied Format will be used
3.	**Construction Process**	Yes	
	Unit Test Case Document	Yes	
4.	**Testing Process**	Yes	
	Software Test Plan	Yes	
	Test Case	Yes	
	Build Acceptance Report	Yes	
	Bug Bash Defect Report	No	Project is not supposed to go under Bug Bash Cycle.
	Test Summary Report	Yes	
5.	**Build and Release Process**	Yes	
	Release Note	Yes	
	Read Me	Yes	
	Build Notification Mail Template	Yes	
6.	**Deployment Process**	Yes	
	Deployment Schedule	Yes	
	Rollout Plan	No	No Specification from Client
Horizontal Activities			
1.	**SQA-QL Process**	Yes	
	Quality Planning and Monitoring Document (QPMD)	No	Client supplied information is used.
	QL Status Report	Yes	
2.	**Internal Audit Process**	Yes	
	Audit Report	Yes	
3.	**Resource Management Process**	Yes	
	Handover Takeover Report	Yes	
	Ramp Up Plan	Yes	
4.	**SCM Process**	Yes	
	SCM Plan	Yes	

5.	**Review Process**	Yes	
	Review Report	Yes	Review Report will be prepared for all the – Technical documents (like SRS, HLD, LLD, Test Case etc. – Planning documents (SPMP, SCMP, QPMD etc.) – Code Review – Defect Review
6.	**Decision Analysis and Resolution Process**	Yes	
	Decision Analysis and Resolution Form	Yes	
7.	Minutes of Meeting	Yes	
8.	Minutes of Chat	No	MOM is already used.
9.	Measurement & Metrics	Yes	
10.	Inter-group Coordination	Yes	

Table 2.3: **Process Used**

2.3.3. Standards, Checklists, and Guidelines

Following standards, checklists, and guidelines will be followed during the life cycle of the project.

Sr. No.	Processes	Standards	Checklists	Guidelines
		Project Management		
1.	**Project Kick Off Process**		Project Kick Off Checklist	
2.	**Estimation Process**			Function Point Estimation Guidelines
				Estimation Guidelines
				Task Based Estimation Guidelines
				Task Based Estimation Checklists
3.	**Risk Management Process**		Risk Management Checklist	
4.	**Project Planning & Monitoring Process**		Microsoft Project Plan Checklist	QMS Tailoring Guideline

Sr. No.	Processes	Standards	Checklists	Guidelines
			Project Planning Checklist	Microsoft Project Plan Guidelines
			Status Review Checklist	Project Team Meeting Agenda Guidelines
5.	**Change Management Process**		Change Request Checklist	
6.	**Project Closure Process**		Project Closure Checklist	
			SDLC	
1.	**Requirements Management Process**		Requirements Understanding Document Checklist	Requirements feasibility Guidelines
			SRS Review Checklist	
			Technical Feasibility Checklist	
			Requirements Management Checklist	
2.	**Design Process**		Design Checklist	
3.	**Construction Process**	Database Coding Standards/ .net Coding Standards	Construction Checklist	Code Review Guidelines
4.	**Testing Process**	WinRunner Coding Standards/ other testing tool related coding standards		Window Installation Testing Checklist/ Other installation Checklist
				TTPro Guidelines/ other bug tracker tool related guidelines
				Linux Installation Testing Guidelines/ other installation guidelines
5.	**Build and Release Process**			Build and Patch Versioning Guidelines
6.	**Deployment Process**		Deployment Checklist	User Guide.doc
				User Guide.frm
			Horizontal	
1.	**SQA-QL Process**			Quality Leader Guidelines
2.	**Resource Management Process**		Resource Management Checklist	
			New Joinee Project Induction Checklist	

Sr. No.	Processes	Standards	Checklists	Guidelines
			Resource Release Checklist	
3.	**SCM Process**	Access Rights Standards	Configuration Audit Checklist	SCM Guidelines
				Check in Checkout Guidelines
				Review Checklist
4.	**Review Process**		Review Documents	
5.	**Measurement & Metrics Process**			Baseline Guidelines
6.	**Intergroup Coordination Process**			Communications Guidelines
7.	**Others**			Meeting Guidelines
				E-Mail Communication Guidelines
				Conference Call Guidelines

Table 2.4: **Checklists Details**

2.4. Resource Plan

Resources required by the project to perform development, project management (estimation, planning, monitoring, risk management, Causal Analysis) as well as other activities (such as reviews/ audits, measurements and configuration) are planned here.
Resources include human resource, hardware, software, tools and components.

2.4.1. Human Resource

The team structure to execute this project is as follows:

Sr. No.	Role	Number of resources required	Responsibilities	Required Skill Set
1.	Project Manager	1	– Project Planning, scheduling and monitoring, quantitative & statistical process control – Client communication, communication with all the departments for project needs – Risk management and mitigation, evoking Decision Analysis and resolution	Year of experience and expertise in following areas: - **PM Activities**: 1-yr. Of exp - **Design/.net technology** : 2 – yrs of exp - **QA Process** – 3-yrs of exp

Sr. No.	Role	Number of resources required	Responsibilities	Required Skill Set
2.	Project Leader	1	- Monitoring the Development process - Communication with PM - Review of Code - Sending WSR to PM	- **Design/.net technology** : 3 – yrs of exp - **QA Process** – 1-yrs of exp
3.	Developers	3	- Understanding the Requirements - Design the modules - Writing the Codes to implement the system - Unit testing	- **.net technology** : 1-2 – yrs of exp
4.	QA Lead	1	- Creating Test plan - Assigning various modules to QA - Review the Test cases - Sending WSR to PM	- **.net technology** : 1 – yrs of exp - **Black Box testing**: 3 yrs exp.
5.	QA Engineers	1	- Understanding the Requirements - Writing the test cases for it - Executing the test cases - Updating the Results - Sending BAR and TSR.	- **.net technology** : 0.5 – yrs of exp - **Black Box testing**: 1 yrs exp.
6.	Developer + Quality Leader (QL)	1	- QPMD preparation - Conducting reviews and audit on the project. - Checking process compliance in the project. - Providing guidance to the project team for QMS implementation.	- **SQA Process** : 0.5 yrs exp
7.	Developer Configuration + Controller (CC)	1	- Monitoring the Process - Verifying the Document's structures.	- **SQA Process** : 0.5 yrs exp
8.	Build Engineer + Developer	1	- Creating the Build - Deploying it on testing server - Sending notification to QA	- **Deployment exp** : 1 yrs
9.	SQA	1	- Process Management	- **SQA Process** : 2 yrs exp

2.4.2. Development Environment

This section refers to development environment and Client supplied Hardware / Software.

2.4.2.1. Hardware

This section states all the hardware resources required for the project.

Sr. No.	Purpose	Configuration	Quantity	Critical (Yes / No)
1.	Client	Desktop Pentium III m/c, 256 MB RAM, 40 GB Hard Disc	4	
2.	Web server	P IV 2.4 GHz, 512 MB RAM, 2 x 40 GB Hard Disk	1	
3.	Database Server	1 x P IV 2.4GHz, 1 GB RAM, 2 x 40GB IDE HDD	1	
4.	Application Server	Win2k Server, NET, source offsite, Visual Source Safe, IIS	1	

Table 2.5: **Hardware details**

2.4.2.2. Software

This section states all the software and specific languages needed for the project.

Sr. No.	Name of the software	Version/release details	No. of licenses	Critical (Yes / No)
1.	Visual Studio .Net	2003	3	Yes
2.	SQL Server	2000	1	Yes
3.	Rational Rose	2000	1	Yes
4.	Windows XP/NT	2001	1	Yes
5.	PBN	2002	1	Yes
6.	AJAX	2005	1	No
7.	Microsoft Visio	2003	1	No
8.	Microsoft Office	2003	1	No

Table 2.6: **Software Used**

2.5. Test environment

All the Test Environment is detailed in Software Test Plan.

2.6. Components reused

2.6.1. Reusable Components used

Following are the reusable components, which will be used in this project:

1. PBN Database
2. LDAP Authentication
3. Administrator

2.6.2. Components Can be reused

Following components of this project can be used in other projects/ applications:

1. Administrator database
2. Test Lab's Result
3. Requirements

2.7. Project management tools used

The following section refers to the tools used for project management activities.

Sr. No.	Purpose	Tools	Critical (Yes / No)
1.	To generate Timesheet	PBN	No
2.	For Project planning and tracking	MS Project	Yes

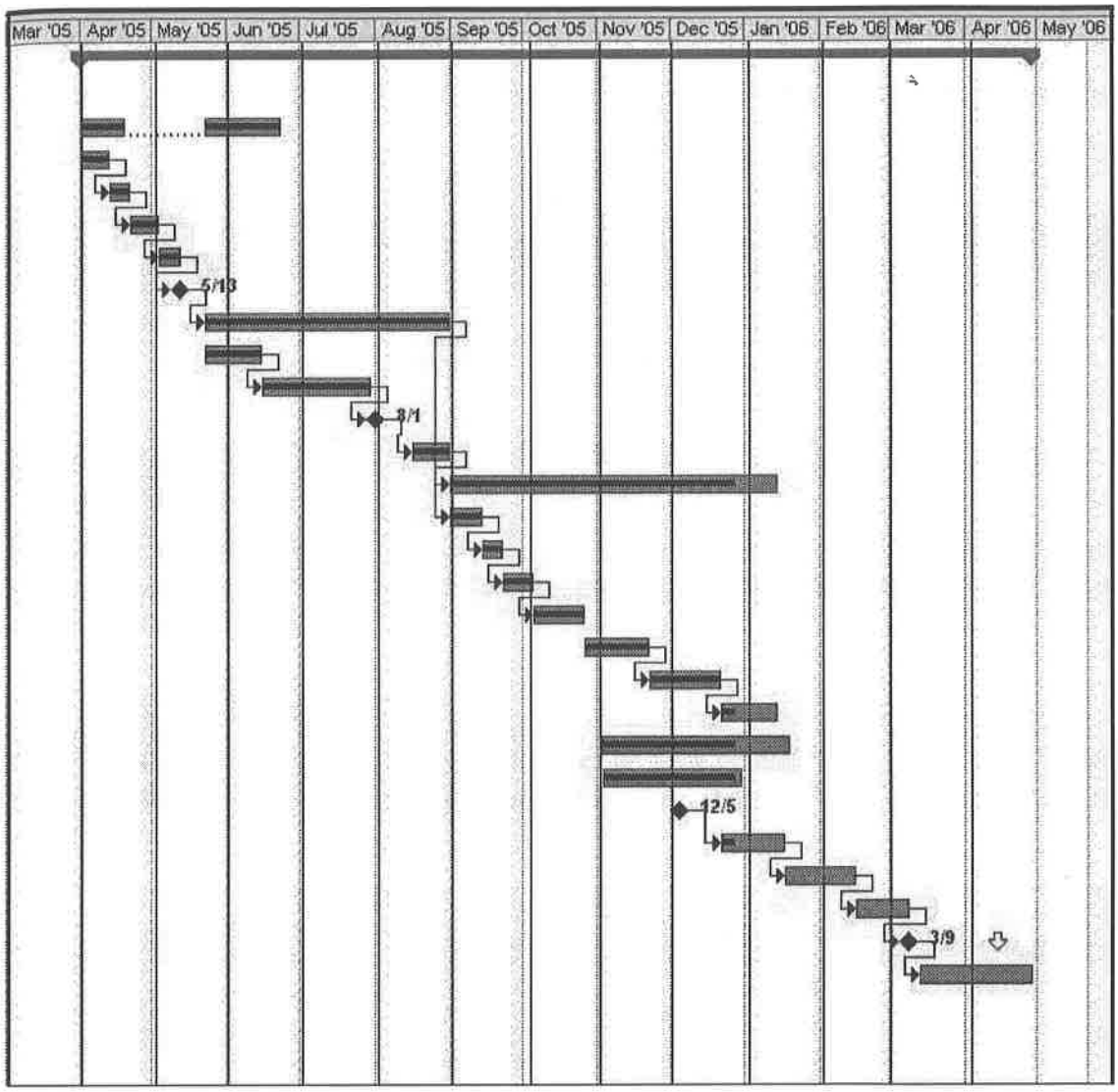
Table 2.7: **Project Management Tools**

2.7.1. Gantt Chat of QA Eye

This chart is used to manage the projects SDLC cycle. In this Project schedule is defined. For QA Eye project following Gantt chart is represented.

	Task Name	Duration	Start	Finish	Predecessors
0	QA Eye	281 days?	Fri 4/1/05	Fri 4/28/06	
1	0. Project Kickoff	1 day?	Fri 4/1/05	Fri 4/1/05	
2	1. Requirement Analysis	36 days	Fri 4/1/05	Wed 6/22/05	
3	1.1 Administrator	8 days	Fri 4/1/05	Tue 4/12/05	
4	1.2 Requirements	7 days	Wed 4/13/05	Thu 4/21/05	3
5	1.3 Test Plan	8 days	Fri 4/22/05	Tue 5/3/05	4
6	1.4 Test Lab	7 days	Wed 5/4/05	Thu 5/12/05	5
7	1.5 Reports	6 days	Fri 5/13/05	Fri 5/20/05	6
8	2. Project System Design	72 days	Mon 5/23/05	Tue 8/30/05	7
9	2.1 Hign Level Design	17 days	Mon 5/23/05	Tue 6/14/05	
10	2.2 Low Level Design	33 days	Wed 6/15/05	Fri 7/29/05	9
11	2.3 Database Design	11 days	Mon 8/1/05	Mon 8/15/05	10
12	2.4 UI Design	11 days	Tue 8/16/05	Tue 8/30/05	11
13	3.Coding	98 days	Wed 8/31/05	Fri 1/13/06	12
14	3.1 UI Implementation	10 days	Wed 8/31/05	Tue 9/13/05	8
15	3.2 Authentication	6 days	Wed 9/14/05	Wed 9/21/05	14
16	3.3 User Controls	9 days	Thu 9/22/05	Tue 10/4/05	15
17	3.4 Requirements Page	15 days	Wed 10/5/05	Tue 10/25/05	16
18	3.5 Test Plan page	19 days	Wed 10/26/05	Mon 11/21/05	
19	3.6 Test Cases page	22 days	Tue 11/22/05	Wed 12/21/05	18
20	3.7 Report Generation	17 days	Thu 12/22/05	Fri 1/13/06	19
21	4.Testing	56 days	Wed 11/2/05	Wed 1/18/06	
22	4.1 Unit Testing	41 days	Thu 11/3/05	Thu 12/29/05	
23	4.2 Functional Testing	13 days	Mon 12/5/05	Wed 12/21/05	
24	4.3 Integration Testing	18 days	Thu 12/22/05	Mon 1/16/06	23
25	4.4 Regression Testing	21 days	Tue 1/17/06	Tue 2/14/06	24
26	4.5 Automation Testing	16 days	Wed 2/15/06	Wed 3/8/06	25
27	5. Deployment	2 days	Thu 3/9/06	Fri 3/10/06	26
28	6. Maintainence	35 days	Mon 3/13/06	Fri 4/28/06	27

Fig 2.1: **Gantt Chart (Software Design, Coding)**



Project: QA Eye Date: Tue 5/23/06	Task	Rolled Up Task	External Tasks
	Progress	Rolled Up Milestone	Project Summary
	Milestone	Rolled Up Progress	Group By Summary
	Summary	Split	Deadline
Page 1			

Fig 2.2: **Gantt Chat (Scope, Analysis, Study)**

2.8. Service Level Agreements

The following section caters to the Service Level Agreements between the different affected Groups.

Sr. No	Required service area	Department / Group providing the service	Expected service level for the project	Agreed service level
1.	Resource	HR	2 days	2 days
2.	Network Connection	IS Department	3 days	3 days

Table 2.8: **Agreements**

2.9. Training

This section addresses the training related details like Identification of training, training plan, and training waiver details. This also addresses the product/ project domain training needs or the application training needs.

2.9.1. Training Plan

On completion of every training, except ramp-up or on-job, records like Attendance sheet, Evaluation Forms, Feedback Forms, and details of trainer are to be submitted to the Training Manager.

Sr. No.	Identified area of training	Number of resources	Planned Date	Suggested Method	Training imparted on
1.	OOAD and UML	3	13-Apr-2005	Computer based, interactive training.	13-Apr-2005
2.	Role Base training for Project Manager/ Configuration Controller/ Quality Lead	4	18-Apr-2005	Training Room	18-Apr-2005
3.	QMS Awareness Training	All	2-May-2005	Training Room	5-May-2005
4.	Ramp up Training	--	As & when a new resource is added	Interactive	--

Table 2.9: **Training Plan**

2.9.2. Training Waiver Details

This section refers to the training waiver details for the team members who do not require training if they have already undergone it or they are experienced.

Sr. No.	Person Name	Training Waived	Reason for Waiver
1.	Amol Choudhari	Ramp up Training	Experienced/ Already Undergone
2.	Sandeep Pawar	OOAD and UML/ Role Base training	Experienced

Table 2.10: **Waiver Details**

2.10. Task allocation and tracking mechanism

This section refers to the task allocation and the tracking mechanism to be used for the project.

The Project's **Work Breakdown Structure** (WBS).

- Requirement and Design : Sandeep, Amol
- Construction : Amol
- Testing : Amol

- PBN is used for tracking the mechanism.

2.11. Critical dependencies

This section refers to the critical dependencies in the project.

Identify and mention the critical dependencies on the various groups (like design group, estimation group, coding group, testing group etc.) and affected group the project is dependent on the VPN connectivity throughout the course of project execution.
Document the actions to be taken if it is found in advance that a critical path will exceed its schedule.
E.g. conducting status review meeting:

- To analyze the impact
- To finalize the actions
- To control the schedule
- By allocating additional buffers or using the schedule slack, if available

For small projects, this section is optional

2.12. Milestones, Deliverables and Acceptance Criteria

This section refers to the schedule for key project milestones, and the project deliverables.

- Deliverables including Client's responsibility e.g. Requirements document, software licenses from Client etc.).
- Schedules may be expressed in absolute calendar time. Also provide schedule at Clients site.

Sr. No.	Deliverables	Milestones	Responsibility	Frequency	Planned Schedule	Acceptance Criteria	Remarks*
1.	Project Planning, Software Test Plan, Test Case's, Integration Checklist	SPMP	Project Manager/QA	Monthly	1-Apr-05	Review and sign-off from Client	
2.	Demo build	Product demonstration	Project Manger/Developer	Once	13-Apr-06	Review and sign-off from Client	
3.	Code Base of build/ patch, release note, read me, user documentation	Client Release	Project Manager/Developer	As planned	13-Apr-06	Review and sign-off from Client	

Table 2.11: **Milestones**

2.13. Project Organization

The following flowchart depicts the hierarchy of the project organization/ Roles established to execute the project. The structure also displays the reporting and escalation mechanism of the project.

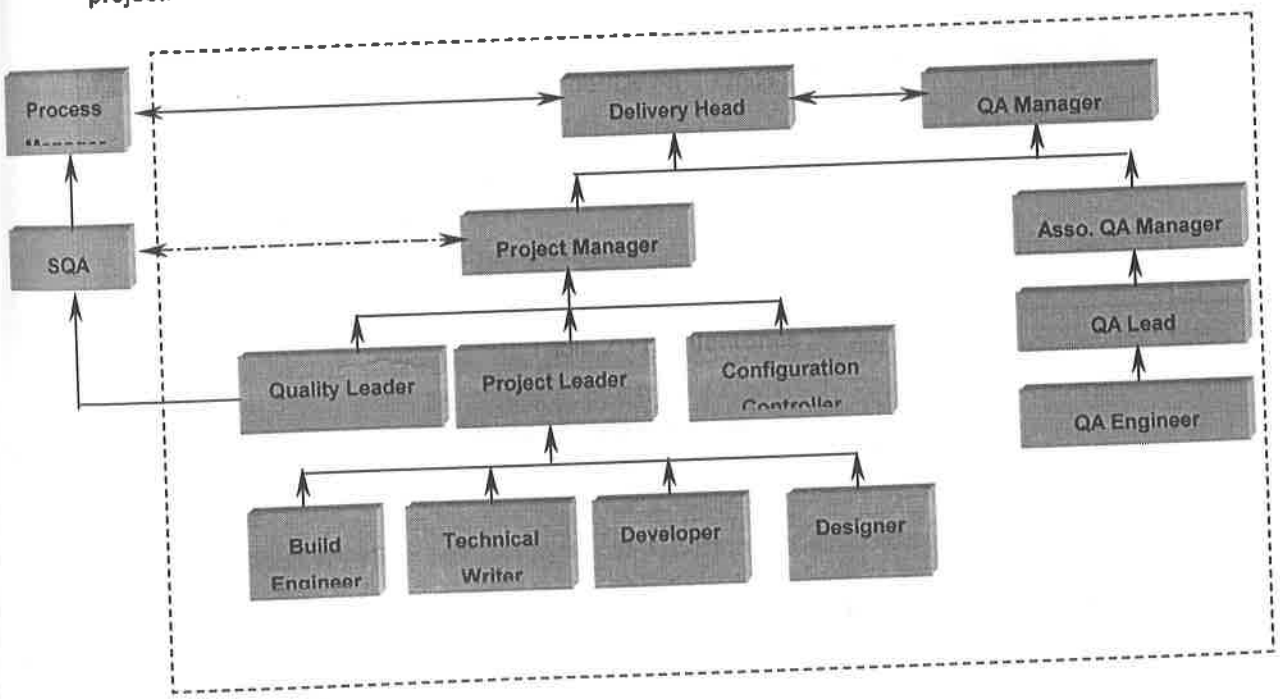


Fig: 2.3: **Hierarchy of Organization**

2.14. Monitoring and Controlling Mechanisms

For this project **PBN** will be used for monitoring and controlling the project.
Review meetings will be held **weekly** for tracking the schedule.
Project Team member will submit their weekly Timesheet.

- Mechanism for observation and record changes.
- The Project Manager/ Project Leader will update the schedule in the following situations:
  - Any activity is completed.
  - There is a change in the scope of the project / process.
  - Where there is any major changes requested by the Client.
- The Project Manager/ Project Leader will update the Delivery Head about the status of the project after the status review meeting
- State escalation mechanisms for referring issues / problems to Senior Management.

2.14.1. Status Reporting

Status reporting will be through E-mail / MOM and conducted on Weekly and all the status report will go to PM.

2.14.2. Milestone Reviews

Status Report (PSR) of every milestone completed will be reported through email and reviewed at subsequent status review meeting.
Following milestones will be followed up by formal Causal Analysis and Resolution (CAR):

- Design review
- Intercommunications
- Meeting Deadlines
- Defect density
- Change in construction phase.

2.14.3. Escalation

The following section addresses the escalation mechanism for the project whenever any issues or queries are raised within the project or outside the team.
An issue will be escalated if it affects any one of the following:

- Scope
- Schedule
- Agreed requirements
- Quality

An issue will also be escalated if it is unresolved or out of control of the concerned parties.

The issues may arise due to:

- Disagreement between the parties
- Unavailability of resources
- Technological hurdles
- Lack of decisions in time

- Quality-related issues
- Lack of response from either party

The SQA and QL will raise the process-related issues to the Process Manager and project-related issues to the Project Manager. The SQA will appraise the Senior Management on quality-related issues, if any, on a monthly basis.

For any unresolved issues or issues that threaten the schedules or relationships, the concerned managers would make immediate attempts to resolve the issues amicably. If a solution for the issue is not mutually agreed, the issue will be escalated.
The project status meetings and project reviews will be conducted using Status reports and ad-hoc communication. This will help in preventing the escalation of issues.

2.14.4. This will be illustrated in the table below.

Threshold Period	Escalate to	Designation
Issues raised by the Client		
3 days	Amol Choudhari	Project Leader
7 days	Sandeep Pawar	Project Manager
Issues raised by Cybage Software Pvt. Ltd.		
3 days	Amol Choudhari	Project Leader
7 days	Sandeep Pawar	Project Manager
Above 7 days	Gurvinder Chattwal	Senior Management

Table 2.12: **QA Eye Escalation**

2.15. Estimation

This section refers to the project estimates carried on the project. Here Function Point Estimation used in this project.

2.15.1. Function Points

Count the number of function points in each category by categorizing the individual requirements from QA Eye's requirements document. In this table the functions are in bold and the complexity weighting factors are in italics:

Category	Simple	Average	Complex	Function Points
Outputs	**8** x *4* = 32	**9** x *5* = 45	**3** x *7* = 21	32 + 45 + 21 = 98
Inputs	**7** x *3* = 21	**9** x *4* = 36	**9** x *6* = 54	21 + 36 + 54 = 111
Inquiry Outputs	**5** x *4* = 20	**7** x *5* = 35	**3** x *7* = 21	20 + 35 + 21 = 76
Inquiry Inputs	**5** x *3* = 15	**6** x *4* = 24	**4** x *6* = 24	15 + 24 + 24 = 63
Files	**12** x *7* = 84	**3** x *10* = 30	**4** x *15* = 30	84 + 30 + 60 = 174
Interfaces	**9** x *5* = 45	**6** x *7* = 42	**8** x *10* = 80	45 + 42 + 80 = 167
			Total Raw Function Points	689

Table 2.13: **Function Points**

- ***Value Adjustment Factor (VAF)***

This is based on General System Characteristics. VAF rates the general functionality of the application being counted. Each characteristic has an associated description that helps determine the degree of influence of the characteristic. The degree of influence varies from 0 to 5 for each characteristic.

- *Value Adjustment Factor (VAF)* = ***Sum of 14 Degrees of Influence *.01 + 0.65***
  = ***TDI *.01 + 0.65***

There are 14 General System Characteristics. For each of the following "environmental factor" assign a rating of 1-5 for this project, where 5 is the highest. Then sum these to create an "environmental influence factor".

➢ ***General System Characteristics (GSCs)***

Environmental Factor	**Rating (0, 1, 2, 3, 4, 5)**
Data Communications	5
Distributed Computing	4
Performance Requirements	3
Constrained Configuration	0
Transaction Rate	5
Online Inquiry	4
End-User Efficiency	2
Online Update	2
Complex Processing	5
Reusability	2
Ease of Conversion / Install	3
Ease of Operation	4
Used at Multiple Sites	2
Potential for Function Change	2
Total (N)	43

Table 2.14: **GSCs**

➢ ***Calculate Complexity Adjustment Factor (CAF) using N from step B above.***

CAF = 0.65 + (0.01 x N) = 0.65 + (0.01 x 43) = **1.08**

➢ ***Compute Adjusted Function Points (AFP)***

AFP = FP (Raw FP from step A) x CAF = 689 x 1.08 = **744.12**

➢ ***Convert to LOC (Optional).***

The 48 below is came from a 'standards' table that said 48 lines of .net code per function point.

LOC for the .net language = AFP x LOC / AFP = 744.12 x 48 = **35,717** LOC

2.16. Risks Management Plan

This serves as a planning mechanism of identifying potential problems in advance. This proactively focuses on preventing problems.

2.16.1. Risk Management Strategy

Risk Management Strategy of the project would be as follows:

2.16.2. Risk projection

Risk projection also called risk estimation, attempts to rate risk in two ways-the like hood or probability that the risk, should it occur.

- ***Risk Table:***

A risk table provides a project manager with a simple technique for risk projection. We began by listing risk in the first column of the database. Risk table is given below

Risk	Probability	Impact
Size estimation may be significantly low	**60%**	**Critical**
End user hesitate to use the system	**30%**	**Marginal**
Delivery deadlines will be tightened	**50%**	**Critical**
Customer will change requirement	**80%**	**Critical**
Technology will not meet expectations	**30%**	**Very Critical**
Funding will not lost	**30%**	**Critical**
Improper Understanding of Project definition	**10%**	**Critical**

Table 2.15: **Risk Table**

Points that may be included in the projects risk management strategy include:

Initial identification of risks:

1. Identification of new risks during the course of project: (how new risks would be identified and acted on. New risks may be identified during team meeting, status reviews, metrics analysis etc.)
2. Frequency of periodic evaluation of identified risks and their impact/ probability: (the period/ event on which the project risks will be evaluated and updated, if required. The period could be a fortnight or a month, while the event could be phase end or milestone closure. In such evaluation, probability and impact of existing risks will also be re-assessed. These parameters are likely to change depending on phase of the project.)
3. Method of continuous risk monitoring (monitoring early warning signals, monitoring mitigation actions for risks materialized)

3. Software requirement specification

3. Software requirement specification

3.1. Introduction

The primary goal of this document is to provide a complete and accurate list of requirements for the "Cybage QA Eye" project to be developed for Cybage Software Pvt. Ltd.
The QA eye is Web-based test management tool. It helps user to organize and manage all phases of the application testing process, including specifying testing requirements, planning tests, executing tests, and generating reports.

3.1.1. Purpose of the Document

This document will be used for capturing QA tool's requirements to be developed for client Cybage.
The purpose of this plan is to:

- Verify that all the requirements for the project are defined and understood.
- To establish & maintain an agreement on the requirements with the Client.

3.1.2. Objectives of SRS

The objectives of SRS are: -

- A centralized repository for test cases across all the projects associated with their clients.
- Administrator should responsible for managing clients, projects, users and groups.
- To ensure that requirements for the projects are defined and clearly understood.
- To track the requirements with the user acceptance test cases.
- Tool should provide standard structure to define Test Plan for any projects
- Centralized test data should be available to the group having proper permissions.
- To ensure that all functional requirements required for the project are captured.
- To track the test cases results to the requirements defined.
- Separate Test Lab for executing the test cases defined in Test Plan.
- The tool should be such that user should be able to generate reports.

3.1.3. Scope of SRS

Scope of SRS is to develop and maintain a centralized repository for Quality Stuffs like Test Requirements, Test Plan, Test Lab across all the projects of particular Clients.
Standard structure is defined for the Test Plan and associating the defined test cases to test labs for manual execution. Finally maintaining reports of all the centralized test data to overview and analyze the test results and testing process.

3.2. Requirements

3.2.1. Scope of the Project

Scope of Project is to have a centralized repository for testing related data across all the projects. It is QA Eye for Clients. There are four major sections via Requirements, Test Plan, Test Lab and Reports.

3.2.2. Functional Requirements

This sub-section describe Client requirements related to core functionality of the "QA Eye Tool"

3.2.2.1. LOGIN (Welcome Page)

Requirement ID	1	
Requirement Category (Technical / Non technical)	Technical	
Requirement Type (Functional)	Functional	
Requirement Importance (Mandatory / Required / Desirable)	Mandatory	
Requirement description	**Login Page.** This page will be a Welcome page of this tool. There will be Username and Password fields for authenticating user across defined project and client. The password and username should be authenticated from LDAP. 1) **Username** field requirements a) It will not accept any special characters. b) It will accept max. 20 characters. 2) **Password** field requirement a) It will accept data in encrypted form. b) It will accept max. 20 characters. c) It will not contain any special characters. d) It will not contain any blanks and spaces. On login page, There will be 2 links provided for 1) Site Administrator : For Administrator Login 2) Help : For Online help to users	
Dependant Requirements (Parent and Child Requirements)	**Parents Requirement**	0
	Child Requirement	2

3.2.2.2. Site Administrator (Link)

Requirement ID	2	
Requirement Category (Technical / Non technical)	**Technical**	
Requirement Type (Functional)	**Functional**	
Requirement Importance (Mandatory / Required / Desirable)	**Mandatory**	
Requirement description	**Site Administrator (Link)** After Clicking on Site Administrator link, application will pop up dialog box for authenticating Administrator. The box will contain Project, Username and Password fields. 1) **Project** Field is combo box containing the registered projects in the application which will be listed alphabetical order.(For Site Administrator **Default** is the project) 2) **Username** Field requirement. (It is alpha. combo box) a) It will by default contain "Administrator" in this field. b) It will not accept any special characters. c) It will accept max. 20 characters. 3) **Password** Field Requirement a) It will accept data in encrypted form. b) It will accept max. 20 characters. c) It will not contain any special characters. d) It will not contain any blanks and spaces. 4) There will be **OK** button, to login into the application. It will have either of 2 actions i.e. a) If Username, Password matches it will allow user to login into the application b) If Username, Password does not matches then it will throw an error stating "Access Denied" 5) **Cancel** button will cancel the operation of administrator login process	
Dependant Requirements (Parent and Child Requirements)	**Parent Requirement**	1
	Child Requirements	3

3.2.2.3. QA Eye Administrator

Requirement ID	**3**	
Requirement Category (Technical / Non technical)	**Technical**	
Requirement Type (Functional)	**Functional**	
Requirement Importance (Mandatory / Required / Desirable)	**Mandatory**	
Requirement description	**QA Eye Administrator** After Successful login of Administrator, he/she will redirect to the main page of Administrator. The page contains 4 tabs. 1)**Clients** : Under Clients Tab, admin can able to add, modify and delete the clients from the QA Eye application. **Refer Requirement 4** 2)**Projects** : Under Projects Tab, admin can able to add projects, update projects and delete projects associated with particular clients. Using this Tab he can also associate different projects to particular clients. **Refer Requirement 5** 3)**Users** : Using this Tab, admin can add users, update user's description and delete users. He also assign users to projects. **Refer Requirement 6** 4)**Groups** : Using this Tab, admin can add new groups, modify existing group, setting permission to groups. He can assign particular group to projects. **Refer Requirement 7**	
Dependant Requirements (Parent and Child Requirements)	**Parent Requirement**	2
	Child Requirements	4,5,6,7

3.2.2.4. Clients

Requirement ID	**4**
Requirement Category (Technical / Non technical)	**Technical**
Requirement Type (Functional)	**Functional**
Requirement Importance (Mandatory / Required / Desirable)	**Mandatory**
Requirement description	**Clients Page :** After Clicking on Clients Tab, admin will redirect to Clients page. 1) This page will have **New** button to create new clients. 2) On clicking this button, New Client form will be displayed with following fields. a) **Client Name** : This will contain alphanumeric characters maximum capacity of 25 characters. b) **Description** : This will contains alphanumeric characters max. of 100 characters. c) **No. of Projects**: This is numeric field. And it will contains maximum of 4 characters. d) **Active** : This will be a check box. Containing the active status of client. 3) On this page footer It will have 3 buttons a) **Save** : For Saving the defined clients into the database b) **Save and New** : for saving current defined client and asking admin to add one more new client. c) **Cancel** : Cancel the Add client Operation. 4) The Clients page will also contain list of registered clients into the application. This list will be displayed into the tabular form or grid. 5) **Update** Operation: a) In Clients page after selecting edit button of particular client to update, it will redirect to the update client page. b) after changing the required fields, Save button will be used to save the changes of existing clients. 6) **Activating** And **Deactivating** Clients: a) The form will have check box namely **Active**. b) Selecting this box will activate the client c) And not selecting this box will deactivate the clients. d) Active clients can be deactivated using update operation 7) **Delete** Clients: While update operation using edit button, the new page will have Delete button option to delete the selected Client.

	a) Once you select Delete option, the application will ask admin to confirm delete operation.	
Dependant Requirements (Parent and Child Requirements)	**Parent Requirement**	3
	Child Requirements	

3.2.2.5. Projects

Requirement ID	**5**
Requirement Category (Technical / Non technical)	**Technical**
Requirement Type (Functional)	**Functional**
Requirement Importance (Mandatory / Required / Desirable)	**Mandatory**
Requirement description	**Projects Page :** After Clicking on Projects Tab, admin will redirect to Projects page. 1) This page will have **New** button to create new Projects. 2) On clicking this button, New Project form will be displayed with following fields. a) **Client Name** :This will contain be a combo box field containing all the registered list of clients. Admin will select the Client under whom the New project is to be added. (The list of client will be retrieved in alphabetical order. b) **Project Name**: This will contain alphanumeric characters maximum capacity of 25 characters. c) **Description** : This will contains alphanumeric characters max. of 100 characters. d) **Active** : This will be a check box. Containing the active status of client. 3) On this Add Project page footer, it will have 3 buttons d) **Save** : For Saving the defined project into the database e) **Save and New:** for saving current defined project and asking admin to add one more new project. f) **Cancel**: Cancel the Add project Operation. 4) The Projects page will also contain list of registered Projects into the application. This list will be displayed into the tabular form or grid.

	5) **Update** Operation: a)On Projects page after selecting edit button of particular project to update, it will redirect to the update project page. b) after changing the required fields, Save button will be used to save the changes of existing projects. 6) **Activating** And **Deactivating** Projects: Using ACTIVE check box project will be activated or deactivated. 7) **Renaming** Projects: **Update**, Admin is able to change the name of Projects. 9) **Delete** Clients: While update operation using edit button, the new page will have Delete button option to delete the selected Project. a) Once you select Delete option, the application will ask admin to confirm delete operation. **10) Removing Project** from Project List: Once if admin delete the project using Delete operation then that project should be automatically removed from the Project list.	
Dependant Requirements (Parent and Child Requirements)	**Parent Requirement**	3
	Child Requirements	

3.2.2.6. Users

Requirement ID	**6**
Requirement Category (Technical / Non technical)	**Technical**
Requirement Type (Functional)	**Functional**
Requirement Importance (Mandatory / Required / Desirable)	**Mandatory**
Requirement description	**Users Page :** After Clicking on Users Tab ,admin will redirect to Users page. 1) This page will have **New** button to create new Users. 2) On clicking this button, New User form will be displayed with following fields. a) **Project's Name** :This will be a combo box field containing all the registered list of Projects. Admin will select the Project under whom the New user is to be added. b) **Client's Name** : Once the Project is selected from Combo

box the associated Client's name will be retrieved on this page in front of Client's name label.

c) **User's Full Name**: This will contain alphanumeric characters maximum capacity of 40 characters.

d) **User Name** : It will contain max. 20 alphanumeric characters. It will not contain any spaces between 2 characters.

e) **Email** : This will contain alphanumeric max. of 40 characters.

f) **Phone No** : This will contain maximum of 10 numeric Characters.

g) **Description** : This will contains alphanumeric characters max. of 50 characters.

3) On this Add User page footer, it will have 3 buttons

g) **Save** : For Saving the defined User into the database

h) **Save and New:** for saving current defined User and asking admin to add one more new User.

i) **Cancel**: Cancel the Add User Operation.

4) The Users page will also contain list of registered Users into the application. This list will be displayed into the tabular form or grid.

5) **Update** Operation:

a)On Users page after selecting edit button of particular user to update, it will redirect to the update User page

b) after changing the required fields, Save button will be used to save the changes of existing user

6) **Renaming** Users:

Update, Admin is able to change the name of Users.

7) **Importing** a New User :

This button will be present on the page where no. of users list associated to particular project is displayed.

There will be two operations to add new user to the selected project.

a) **New** button: Using New button admin can add new user

b) **Import** button: After clicking on this button admin is asked to enter 3 fields to define which existing user he needs to add in selected project.

8) **Changing Password** :

At the time of adding new user, each user has given default password. i.e. it's Username itself.

But admin can change the user's password using Change

	Password link present on Users details page. 9) **Delete** Users: While update operation using edit button, the new page will have Delete button option to delete the selected Users. a) Once you select Delete option, the application will ask admin to confirm delete operation. 10) **Removing User** from User Group: Once if admin delete the User using Delete operation then that User should be automatically removed from the User group list.

3.2.2.7. Groups

Requirement ID	**7**
Requirement Category (Technical / Non technical)	**Technical**
Requirement Type (Functional)	**Functional**
Requirement Importance (Mandatory / Required / Desirable)	**Mandatory**
Requirement description	**Groups Page :** After Clicking on Groups Tab, admin will redirect to Groups page. 1) This page will have **New** button to create new Group. 2) On clicking this button, New group form will be displayed with following fields. a) **Group's Name** : This will be a alphanumeric field It will contain maximum of 20 characters. b) **Create As** : This is Combo box. It will contain the list of available groups present into the system. This group will have already assigned privileges and permission. c) **Description** : This will contains alphanumeric characters max. Of 50 characters. 3) On this Add Group page footer, it will have 3 buttons j) **Save** : For Saving the defined Group into the database k) **Save and New**: for saving current defined Group and asking admin to add one more new group. l) **Cancel**: Cancel the Add Group Operation. 4) The Group page will also contain list of registered groups into the application. This list will be displayed into the tabular form or grid. 5) **Default Groups**: To enable each team member to do his/her job and protect a project from unauthorized access, this tool lets admin assign each

member to a specific user group. This includes predefined user groups with default privileges. These groups and privilege are defined as follows :

a) **QAEyeAdmin:** Group members have full privileges.

b) **Project Manager :** Group members have full privileges in Requirements, Test Plan, Test Lab, and Reports modules. The group also has some administration privileges.

c) **QA Engineer :** Group members have full privileges in the Requirements, Test Plan, and Test Lab Modules. In the Reports module they have view Permission. The group also has some administration privileges.

d) **Developer :** Group members privileges are limited to modifying attachments in the following modules: Requirements, Test Plan, and Test Lab. The group also has some administration privileges.

e) **Viewer :** Group members have read-only privileges in a application

6) Setting User Group **Permissions :**

After creating new group, admin is able to change the permission of the group. For this following operation is carried out.

a) Select the group from the groups table by clicking on its name.

b) There will be associated users In that group

c) On the same page there will be 3 buttons available.

i) **View** : This button will display existing permissions assigned to group

ii) **Change**: This button will be used to change the permission of group. On clicking this button new page will be displayed to set the permission **Refer the Requirement 8.**

iii) **Set As** : This button will be used to assign the permission same like the existing group. On clicking this button the combo box will appear which will display the registered group. By selecting particular group and saying ok will change the permission of the group to the defined group. **Refer the Requirement 8.**

7) **Update** Operation:

a)On Group page after selecting edit button of particular group to update, it will redirect to the update group page

b) After changing the required fields, Save button will be used to save the changes of existing group.

8) **Renaming** Group:

Update, Admin is able to change the name of Group.

	9) **Assigning Existing Sets of Permissions to User Groups** 10) **Delete** Groups: While update operation using edit button, the new page will have Delete button option to delete the selected Group. 11) Adding Users to User Group: Please **refer the Requirement 8**	
Dependant Requirements (Parent and Child Requirements)	**Parent Requirement**	3
	Child Requirements	8

3.2.2.8. Setting up Permissions

Requirement ID	**8**
Requirement Category (Technical / Non technical)	**Technical**
Requirement Type (Functional)	**Functional**
Requirement Importance (Mandatory / Required / Desirable)	**Mandatory**
Requirement description	**Setting up the Permissions Page :** 1) **Adding Users** to User Group: a) In Groups page, On clicking Group name, the list of associated users is displayed in a grid form. b) On this page there will 4 buttons i.e. **Add User, View, Change, Set As.** c) On clicking **Add User** button it will allow you to add new users to the existing Group. 2) Setting User Group **Permissions :** On the same page there will be 3 buttons are used for following operations a) **View** : This button will display existing permissions assigned to group b) **Change**: This button will be used to change the permission of group. On clicking this button new page will be displayed to set the permission i) This page will contains 3 tabs(Operation Set) **Requirements, Test Plan, Test Lab** ii) Each Tab will have multiple check boxes to associate the permissions. iii) After Checking or selecting box the particular permission is said to be associated with that tab or Operation. 3) **Assigning Existing Sets of Permissions to User Groups** a) **Set As** : This button will be used to assign the permission

	same like the existing group. b) Once admin selected the group from combo box, **OK** button is used to complete the operation of assigning Existing Sets of Permissions to User Groups c) **Cancel** : This button is used to cancel the operation of assigning Existing Sets of Permissions to User Groups

3.2.2.9. Main QA Eye Page

Requirement ID	**9**	
Requirement Category (Technical / Non technical)	**Technical**	
Requirement Type (Functional)	**Functional**	
Requirement Importance (Mandatory / Required / Desirable)	**Mandatory**	
Requirement description	<u>QA Eye Main Page</u> After Successful login of User, he/she will redirect to the main page of QAEye tool. The page contains 4 tabs. 1)**Requirements** : Under Requirements Tab, user can able to add, modify and delete the Requirements from the QA Eye application. **Refer Requirement 10** 2)**Test Plan** : Under Test Plan Tab, user can able to add Test cases, update test cases and delete test cases associated with particular clients. Using This tab he can plan the testing activities **Refer Requirement 12** 3)**Test Lab** : Using this Tab, users can execute test cases, update Test results. **Refer Requirement 16** 4)**Reports** : Using this Tab, users can generate, view, or send a reports bases on the activities defined by them **Refer Requirement 20**	
Dependant Requirements (Parent and Child Requirements)	**Parent Requirement**	1
	Child Requirements	10,

3.2.2.10. Requirement Outline

Requirement ID	**10**
Requirement Category (Technical / Non technical)	**Technical**
Requirement Type (Functional)	**Functional**
Requirement Importance (Mandatory / Required / Desirable)	**Mandatory**
Requirement description	**Testing Requirements Outline page:** 1) **The Requirement module** : In this module requirements will be displayed in the Document View, Coverage View, or Coverage Analysis View. a) **Document View** will display Requirement tree. b) **Coverage View** will display the associated test cases with particular requirement. 2) The Requirements **Menu Bar :** a) The **Requirements** menu will contains commands that enable user to add and modify requirements in the tree, generate a test from a requirement, and mail a requirement. b) The **View** menu will contains commands that enable user to set the display of the requirements tree, find a requirement, and view tests coverage, associated defects, and attachments. 3) The **Requirements Toolbar :** On this tool bar there will be following buttons to be used for operating on requirements. a) **New Requirement** : This button will be used to add a new requirement to the requirements tree. The requirement is added below the currently selected requirement at the same hierarchical level. **Refer Requirement 11** b) **New Child Requirement** : This will be used to add a new requirement to the requirements tree. The child requirement is added below the currently selected requirement at a lower hierarchical level. **Refer Requirement 11** c) **Find** : This button will be used to open the Find Requirement dialog box, enabling you to search for a requirement in the tree. **Refer Requirement 11** d) **Attachments** : This button will open the Attachments dialog box, enabling user to add an attachment to the selected requirement e) **Mail Requirement** : This button will open the Send Mail dialog box, enabling you to send a requirement e-mail to

recipients selected from a list, or to any other e-mail address. **Refer Requirement 11**

4) The **Requirements Tree**
In the **Document View**, requirements data will be organized and displayed in the requirements tree. Each line in the tree displays a separate requirement record. In **Coverage View**, users can view requirements data by clicking the **Details** tab.

The following requirements details will be displayed in the requirements tree:

a) **Req ID :** A unique alphanumeric ID for the requirement that is assigned automatically. Note that the Req ID cannot be modified.
b) **Requirement Name :** The requirement name.
c) **Author :** The user name of the person who created the requirement. By default, this is the login user name.
d) **Creation Date and Time :** The date and time on which the requirement was created. By default, the creation date is set to the current database server date. It will also allow users to click the down arrow to display a calendar and select a different creation date.
e) **Direct Cover Status :** The current status of the requirement. By default, the status is **Not Covered**.
A requirement status can be one of the following:
   I. **Not Covered**: The requirement has not been linked to a test.
   II. **Failed**: One or more tests covered by the requirement have an execution status of "Failed".
   III. **Not Completed**: One or more tests covered by the
   IV. **Passed**: All the tests covered by the requirement have an execution status of "Passed".
   V. **No Run**: All the tests covered by the requirement have an execution status of "No Run".
   VI. **N/A:** The current status of the requirement is not applicable.
f) **Priority :** The priority of the requirement, ranging from low priority (level 1) to urgent priority (level 5).
g) **Product :** The component of the application on which the requirement is based.
h) **Type :** The type of requirement. By default, can be **hardware** or **software**.

Dependant Requirements (Parent and Child Requirements)	**Parent Requirement**	9
	Child Requirements	11, **20,21**

3.2.2.11. *Defining Requirements*

Requirement ID	**11**
Requirement Category (Technical / Non technical)	**Technical**
Requirement Type (Functional)	**Functional**
Requirement Importance (Mandatory / Required / Desirable)	**Mandatory**
Requirement description	**Defining Requirements page:** 1) Creating a **Requirements Tree** a) After clicking on New Requirement, The New Requirement Dialog box will open . It will consists of following fields : • Requirement Name ,Priority, Product , Type And there will be 2 buttons, OK, CANCEL. I. **OK** : This will create a new requirement after specifying the fields in a New Requirements dialog box. New requirement will be added in the requirement tree II. **Cancel** : It will cancel the operation. b) After **OK** click, application will add Requirement to the Requirement tree. c) To add the Attachments to this new requirement, **Attachments** button will be used. d) To add one more requirement in a tree , **New Requirement** button will be used e) To add next requirement below the previous defined requirement, **New Child Requirement** button will be used. 2) **Finding Requirements** in the Tree a) After clicking on **Find** tab, it will open Find dialog box b) There will be 2 fields on a dialog box • Optional field : **Parent Requirement** • **Requirement to find** : It will accept requirement name which to search. It will not case sensitive c) On clicking **Find** button , the Requirements module attempts to locate the requirement with the specified value. If the search is successful, the Search Results dialog box opens and displays a list of possible matches. 3) **Viewing the** Requirements Tree : a) **Expand** and **Collapse** the tree: • To expand a branch in the tree, click the Expand sign (+) to the left of the branch name. • To collapse a branch in the tree, click the Collapse sign (-)

	to the left of the branch name. 4) **Mailing** Requirements : a) After clicking on **Mail Test**, it will open new message window and the mail will be automatically send to respective or selected person in TO,CC, or BCC list. 5) **Modifying** the Requirements Tree a) **Renaming Requirement** : For renaming requirement, user will click on requirement . By renaming requirement he will save the changes. b) for modifying a requirement there will be edit provided to each requirement to modify the requirement. 6) **Deleting** Requirement : User can delete a requirement from the requirements tree. Deleting a requirement topic also deletes any children of the requirement topic.	
Dependant Requirements (Parent and Child Requirements)	**Parent Requirement**	10
	Child Requirements	**20,21**

3.2.2.12. Test Plan Module

Requirement ID	**10**
Requirement Category (Technical / Non technical)	**Technical**
Requirement Type (Functional)	**Functional**
Requirement Importance (Mandatory / Required / Desirable)	**Mandatory**
Requirement description	**Test Plan module Page:** 1) **The Test Plan module** : On clicking Test plan tab this page will be opened. This page will contains a) **Test Plan toolbar** : With these buttons of commands commonly used when creating and modifying the test plan tree. It will contain following buttons I. **New Folder**: Adds a new subject folder to the test plan tree. II. **New Test**: Adds a new test to the test plan tree. III. **Delete**: Deletes the selected folder or test from the tree. **refer requirement 13** IV. **Find Folder/Test**: Opens the Find Folder/Test dialog box, enabling you to search for a folder or test in the

	tree. **refer requirement 13** V. **Sort Folders**: Opens the Sort Folders in Test Plan Tree dialog box, enabling you to custom sort the folders in the test plan tree. By default, folders are sorted by name. **refer requirement 13** b) **Test Plan Tree**, a graphical representation of users test plan. c) **Design Steps tab**, a list of test steps explaining how to execute the test currently selected on the test plan tree. The tab includes an icon if the selected test has design steps. d) **Attachments tab**, a list of attachments that provide additional information for the test currently selected on the test plan tree. The tab includes an icon if the selected test has any attachments. e) **Requirements Coverage tab**, a list of the requirements that the test currently selected on the test plan tree meets. The tab includes an icon if the selected test has requirements coverage	
Dependant Requirements (Parent and Child Requirements)	**Parent Requirement**	9
	Child Requirements	13,14,15, **20,21**

3.2.2.13. Developing Test Plan Tree

Requirement ID	**13**
Requirement Category (Technical / Non technical)	**Technical**
Requirement Type (Functional)	**Functional**
Requirement Importance (Mandatory / Required / Desirable)	**Mandatory**
Requirement description	**Defining Requirements page:** 1) **Creating a Test Plan Tree** a) After clicking on New Test Module, The New Folder Dialog box will open . It will consists of following field : • Folder Name (Module Name) And there will be 2 buttons, OK, CANCEL. I. **OK** : This will create a new folder after specifying the fields in a New Folder/Module dialog box. New module/Folder will be added in the Test Plan tree II. **Cancel** : It will cancel the operation. b) After **OK** click, application will add new folder/module to the Test plan tree.

f) To add the Attachments to this new module, **Attachments** button will be used.

g) To add one more module in a tree , **New Module** button will be used

2) **Adding Tests** to a Test Plan Tree

a) To add the test to the New module, user will select the module first and will click on **New Test** button which is present on Toolbar.

b) On clicking New Test button, Create new Test dialog box will be opened with 2 fields.

- **Test Type**: By default it will contain Manual.
- **Test Name**: It will not contains special characters.

c) There will two buttons on the dialog box. **OK** and **CANCEL** For creating the new test cases under defined module.

d) **Attachment** button will be used to attach new documents related the Test case.

e) **Details** tab will be used to write the details about test case.

f) **Adding Test steps** to test case : **refer requirement 14**

3) Viewing the Test Plan Tree

For viewing Test Plan tree following methods will be used.

a) **Expand** and **Collapse** the tree:

- To expand a branch in the tree, click the Expand sign (+) to the left of the branch name.
- To collapse a branch in the tree, click the Collapse sign (-)to the left of the branch name

4) **Finding Tests** in the Tree

a) After clicking on **Find** tab, it will open Find dialog box

b) There will be 2 fields on a dialog box

- **Find In :** In this field user is able to give the module name of the test. This will have combo box containing all the modules present in the current project on which user is allocated
- **Test name to find** : It will accept Test name which to search. It will not case sensitive.

c) On clicking **Find** button , the Test Plan module attempts to locate the folder or test.

5) **Mailing** Tests :

a) After clicking on **Mail Test**, it will open new message window (Refer Req11.4)

6) **Modifying** the Test Plan Tree

a) **Renaming Test Plan**: For renaming Test plan, user will click on Test plan or test module. By renaming test module he will save the changes.

	7) **Deleting** Test Module or Test cases : User can delete a Test Module or Test cases from the Test Plan tree. Deleting a Test plan module will also deletes any test cases of the this module.	
Dependant Requirements (Parent and Child Requirements)	**Parent Requirement**	12
	Child Requirements	**14 , 20,21**

3.2.2.14. Building Tests

Requirement ID	**14**	
Requirement Category (Technical / Non technical)	**Technical**	
Requirement Type (Functional)	**Functional**	
Requirement Importance (Mandatory / Required / Desirable)	**Mandatory**	
Requirement description	1. Designing Test Steps 2. Calling a Manual Test with Parameters 3. Managing Test Steps	
Dependant Requirements (Parent and Child Requirements)	**Parent Requirement**	13
	Child Requirements	**20,21**

3.2.2.15. Linking Tests to Requirements

Requirement ID	**15**	
Requirement Category (Technical / Non technical)	**Technical**	
Requirement Type (Functional)	**Functional**	
Requirement Importance (Mandatory / Required / Desirable)	**Mandatory**	
Requirement description	1. Linking Requirements to a Test 2. Linking Tests to a Requirement 3. Analyzing Coverage	
Dependant Requirements (Parent and Child Requirements)	**Parent Requirement**	14
	Child Requirements	**20,21**

3.2.2.16. *Test Lab Module*

Requirement ID	**16**	
Requirement Category (Technical / Non technical)	**Technical**	
Requirement Type (Functional)	**Functional**	
Requirement Importance (Mandatory / Required / Desirable)	**Mandatory**	
Requirement description	1. The Test Lab Menu Bar 2. The Test Lab Toolbars 3. The Execution Flow	
Dependant Requirements (Parent and Child Requirements)	**Parent Requirement**	9
	Child Requirements	**17,18,19, 20,21**

3.2.2.17. *Creating Test Sets Tree*

Requirement ID	**17**
Requirement Category (Technical / Non technical)	**Technical**
Requirement Type (Functional)	**Functional**
Requirement Importance (Mandatory / Required / Desirable)	**Mandatory**
Requirement description	1. Adding a Test Set 2. Adding Tests to a Test Set 3. Setting the Test Set Notifications 4. Setting the Test Set On Failure Rules 5. Managing Test Sets

3.2.2.18. *Scheduling a Test Run*

Requirement ID	**18**	
Requirement Category (Technical / Non technical)	**Technical**	
Requirement Type (Functional)	**Functional**	
Requirement Importance (Mandatory / Required / Desirable)	**Mandatory**	
Requirement description	1. Modifying a Test Run Schedule 2. Arranging Tests Sequentially 3. Finding Tests 4. Viewing the Execution Flow Diagram	
Dependant Requirements (Parent and Child Requirements)	**Parent Requirement**	16
	Child Requirements	**20,21**

3.2.2.19. *Running a Test Manually*

Requirement ID	**18**	
Requirement Category (Technical / Non technical)	**Technical**	
Requirement Type (Functional)	**Functional**	
Requirement Importance (Mandatory / Required / Desirable)	**Mandatory**	
Requirement description	1. Editing the Test Steps 2. Updating Results	
Dependant Requirements (Parent and Child Requirements)	**Parent Requirement**	16
	Child Requirements	**20,21**

3.2.2.20. Reports

Requirement ID	20
Requirement Category (Technical / Non technical)	**Technical**
Requirement Type (Functional)	**Functional**
Requirement Importance (Mandatory / Required / Desirable)	**Mandatory**
Reference (Client Document / Interview Findings)	
Requirement description	Following kind of reports are targeted: - 1> Number of Test cases executed in a particular project/build as per engineers. 2> Date wise execution of test cases by engineer. 3> Test cases never executed across all the build in the projects. 4> Test failures as per builds. 5> Test cases blocked as per builds Summary report of all the builds in a project

3.2.2.21. Graphs

Requirement ID	21
Requirement Category (Technical / Non technical)	**Technical**
Requirement Type (Functional)	**Functional**
Requirement Importance (Mandatory / Required / Desirable)	**Mandatory**
Reference (Client Document / Interview Findings)	
Requirement description	Following kind of Graphs are targeted: - 1> Test suites execution Vs Build numbers. 2> Bugs Vs build 3> Test cases status(pass/failed/blocked/untested) Vs Builds.

3.3. Activities by Group

Admin

Manage Client
Manage Groups
Manage projects
Manage Users

QA

Manage projects
View Reports
Manage Requirements
View Requirements
Manage Test Plans
View Test Plans
Manage Test Cases
View Test Cases
Manage Test Suites
View Test Suites
Execute Test Cases

Managers

View Requirements
View Test Plan
View Test Cases
View Reports
Some Administrative Rights

Fig 3.1: **Privileges of Groups**

3.4. Interface Requirements

This section covers user, hardware, and software interface requirements.

3.4.1. User Interface

- Web based interface with online help.

3.4.2. Software Interface

- Server side will be developed in ASP.Net front-end. IIS web server.
- On the client user just needs explorer (internet or Netscape)

3.5. Performance Requirements

The performance requirements of the "QA Tool" system are

- Response time of the web pages shouldn't be more than 1-5 secs.

3.6. Database Requirements

- Database requirement of the "QA Tool" system is SQL server 2000.

3.7. Other requirements (Non Functional)

This section covers the non-functional requirements like security, reliability, maintainability, availability, supportability, usability, scalability, and testability including their constraints.

3.7.1. Security

- Definition of users / user groups
- Access rights
- Levels of authentication

Constraint: Security constraints imposed due to criticality of the data.

3.7.2. Reliability

These are the requirements of the following nature:

- Graceful exit on abnormal inputs / conditions
- Maintaining consistency of system data
- Causing minimal damage to the environment in case of abnormal Conditions.
- Ease of recovery

3.7.3. Maintainability

- By providing user and admin manuals.

  Constraint: N/A

3.7.4. Availability

- It should be available for use all 24 hours.

  Constraint: Down time of machine hosting the server.

3.7.5. Supportability

- Should be able to support every project.
- Should support every kind of client (linux/win32/mac etc) i.e. browsers netscape/IE/safari/mozilla/firefox etc.

Constraint: N/A

3.7.6. Usability

Requirement that affect usability -:

- Training time required for all kind of users to become productive at particular operations.
  Constraint: N/A

3.7.7. Scalability

- It depends upon the way this tool is going to be hosted if it is going to be used per project then it should support 50-80 users.
- If the tool is going to be hosted company wide then it should support for 1000-1200 users
- Constraint: Cannot handle more than 1200 clients.

3.7.8. Testability

- The system should be tested in such a way that it should verify that all the stated, implicit, present, future requirements are met.
- Various documents, templates, tools should be used for finding, tracking of bugs present in the system.
- Proper test plan, test cases should be prepared.
  Constraint: N/A

3.8. Specific Considerations

3.8.1. Assumptions

Under this section all the assumption in the areas of technical, functional and / or support from client when needed will be listed

3.8.2. Technologies

This section lists all the technologies for the project: -

- Front end: - ASP.Net, HTML. And XML
- Backend: - IIS 6.0 web server *and* SQL server 2000.

3.8.3. Software

Software to be used during development Phase

- Microsoft Visual Studio .NET
- SQL Server
- IIS 6.0
- Windows 2003 server or Windows XP

Software used during testing phase: -

- SQL Server
- IIS 6.0
- Windows 2003 server

3.8.4. Hardware

- A windows 2003 server machine with 1GB RAM and P4 or Xeon 2.8+ GHz.
- HDD 80 GB

3.9. Acceptance Criteria

- All the requirements should be implemented.
- Tool should at least be able to support 1000 people before calling it RC.
- Developers interaction should be possible
- Generating various reports
- Maintaining whole test cases database reliably
- Tool should support interaction from various branches
- It should provide highly secure QA mechanism
- Tool should be reliable and easy to learn.

4. System Design

4. System Design

4.1. Introduction

This document will be used for managing the Design architecture of **QA Eye** project to be developed for the client Cybage Software Pvt. Ltd.

4.1.1. Purpose of the Document

The purpose of this document is to:

- Identify various design approaches.
- Identify core modules of the system.
- Identify the best suitable technology.
- Identify database design (physical, logical etc).
- Identify data structures and algorithm.
- Identify and finalizing UI design in terms of technical implementation.

4.1.2. Objective of DD

The objectives for DD are to describe the detailed design of the tool which is to be developed for Managing the Testing process.

4.1.3. Scope of DD

The scope of DD is to implement the detailed structure of a product in decentralized manner. Using this System Design developer will easily implement the System. This will consists of detailed description of each modules in projects along with database design. Various techniques are mentioned here to define the architecture of this QA product.

4.2. System Overview

Software systems used in the QA Eye project are:

QA Eye helps you maintain a project database of tests that cover all aspects of your application's functionality. Every test in your project is designed to fulfill a specified testing requirement of your application. To meet the various goals of a project, you organize the tests in your project into unique groups. QA Eye provides an intuitive and efficient method for scheduling and executing test sets, collecting test results, and analyzing the data.
The QA Eye is Web-based test management tool. It helps testers organize and manage all phases of the application testing process, including specifying testing requirements, planning tests, executing tests, and analyzing QA process

4.3. Architecture

4.3.1. System Architecture

This section provides an overview of the system architecture. It also describes how the functionality and responsibilities of the system were partitioned and then assigned to sub-systems or components.

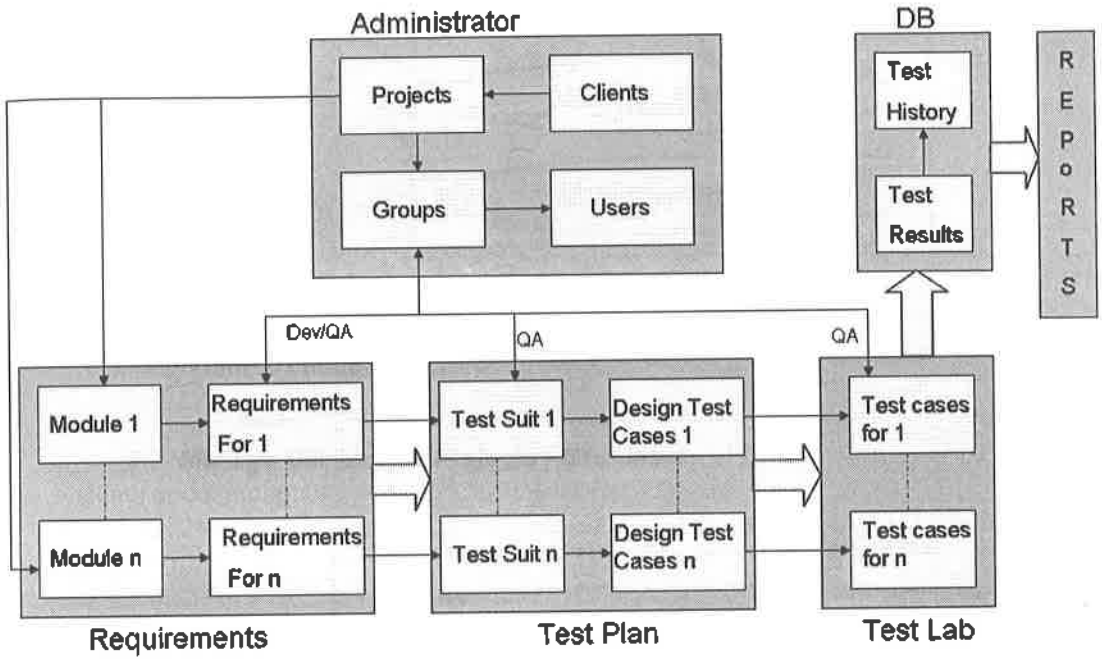


Fig 4.1: **QA Eye Architecture**

This system is partitioned in to four main sub-system as follows :

1) Administrator
2) Requirements
3) Test Plan
4) Test Lab

4.3.1.1. Administrator

Administrator is responsible for following operations :

Manage Clients : Adding clients, updating clients and deleting clients

Manage Projects : Adding projects under particular client, maintaining project list, Activating and deactivating projects, removing projects

Manage Users : Adding users to particular projects , describe user properties, assigning existing group to users and delete users

Manage Groups : Defining groups, assigning permissions to them, gathering users to form a

particular group, assigning existing group permission to new group.

- ***Administrator responsibility (Use Case)***

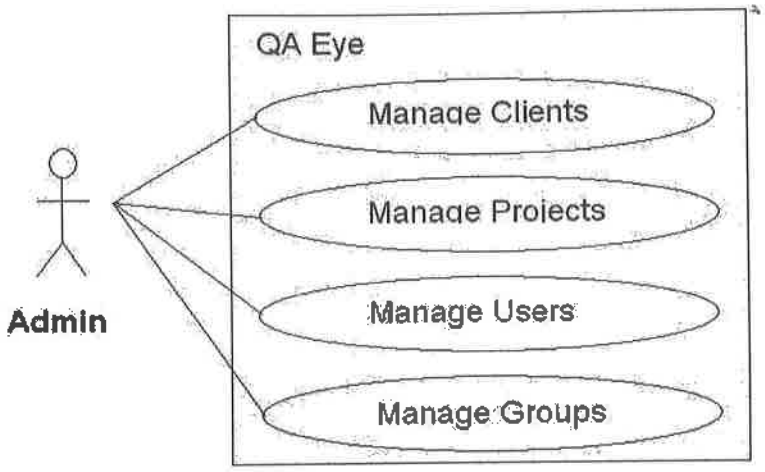


Fig 4.2: **Use case of Administrator.**

- ***Manage Clients and Projects (Use Case)***

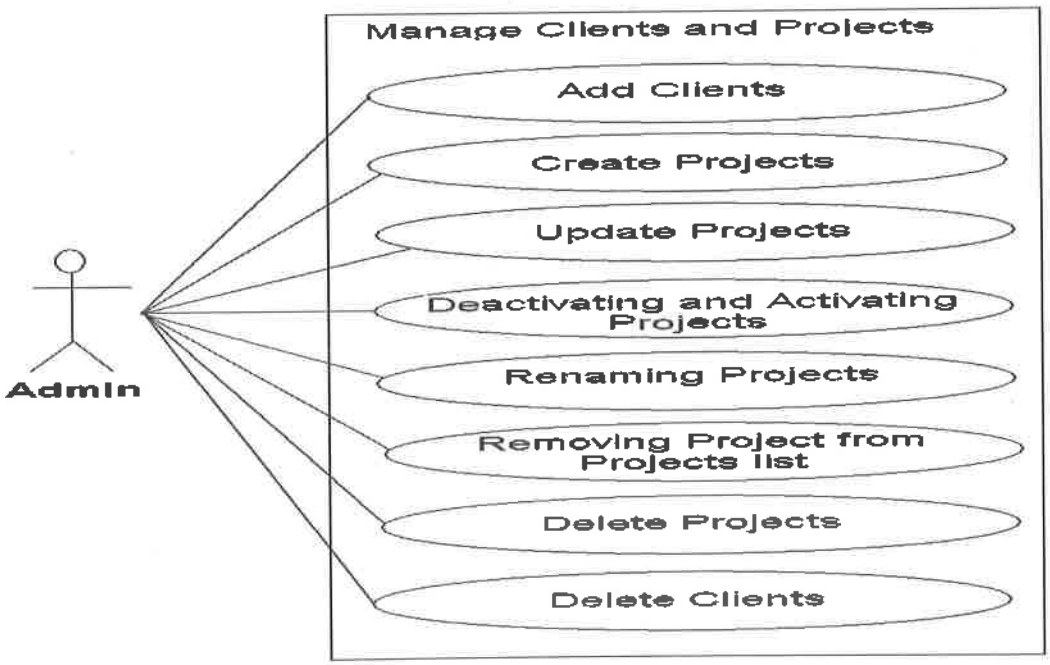


Fig 4.3: **Use case of Manage Client.**

➢ ***Manage Users and Groups (Use Case)***

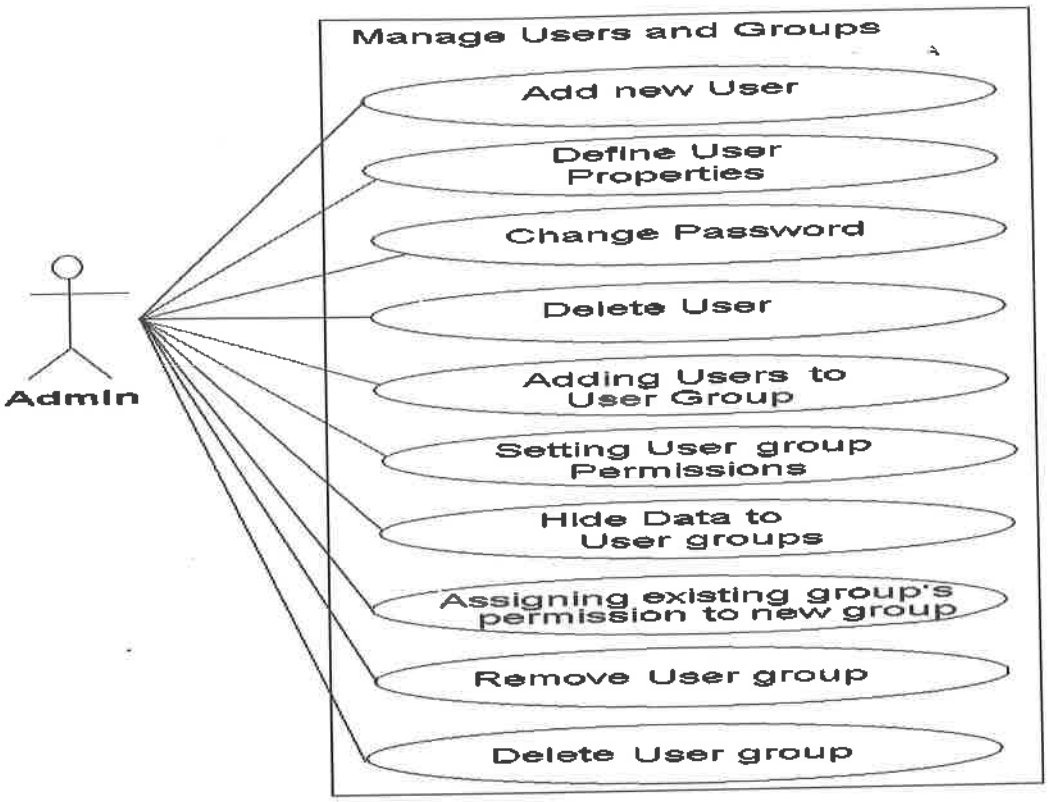


Fig 4.4: **Use case of Manage Users/Groups.**

4.3.1.2. Requirements

In this module the requirements gathering and storing is covered. After defining the particular project user will redirect to this module to define the requirements for that project, which is going to be under QA Cycle. User can add, modify, update and delete the requirements from the projects depending up on the privileges assigned to him. Following operations is to be implemented in this module.

➢ ***Defining Requirements:***

Creating a Requirements Tree
Finding Requirements in the Tree
Viewing the Requirements Tree
Viewing Requirement History
Mailing Requirements
Viewing Associated Defects
Modifying the Requirements Tree
Creating Tests from Requirements

4.3.1.3. Requirement tree

For each requirement topic, a QA tester creates a list of detailed testing requirements in the requirements tree. In this tree Project name is represented as parent node. And under this modules is included which are again parents node of detailed requirements.

1.0 Project Name
1.1 Module1
1.2 Module2
1.3 Module3
1.3.1 Requirement1
1.3.2 Requirement2
1.3.3 .....

Each requirement in the tree is described in detail and can include any relevant attachments. The QA tester assigns the requirement a priority level which is taken into consideration when the test team creates the test plan.

4.3.1.4. Test Plan and Test Lab

- ***Developing the Test Plan /Test Lab***
  - Creating a Test Plan Tree
  - Adding Tests to a Test Plan Tree
  - Viewing the Test Plan Tree
  - Associating Defects with a Test
  - Mailing Tests
  - Finding Tests in the Tree
  - Sorting a Test Plan Tree
  - Modifying a Test Plan Tree

- ***Linking Tests to Requirements***
  - Linking Requirements to a Test
  - Linking Tests to a Requirement
  - Analyzing Coverage

- ***Building Tests***
  - Designing Test Steps
  - Calling a Manual Test with Parameters
  - Managing Test Steps
  - Reports And Graphs

4.3.2. Description of components

4.3.2.1. Login

➢ ***Purpose***

Login Page will enable various types of users to log in to the system and perform various task depending on their roles in projects.
This page will validate the user (tester/developer/manager/admin) across their associated Clients, Projects and groups.

➢ **Technology being used**

ASP.net using c# and MS-SQL server, LDAP authentication, Active Directory etc

➢ **Object Model and Interface**

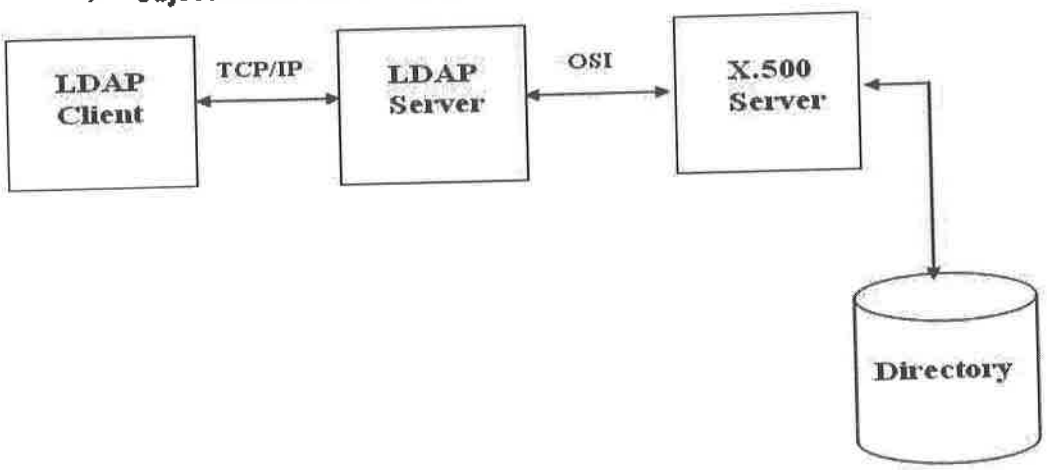
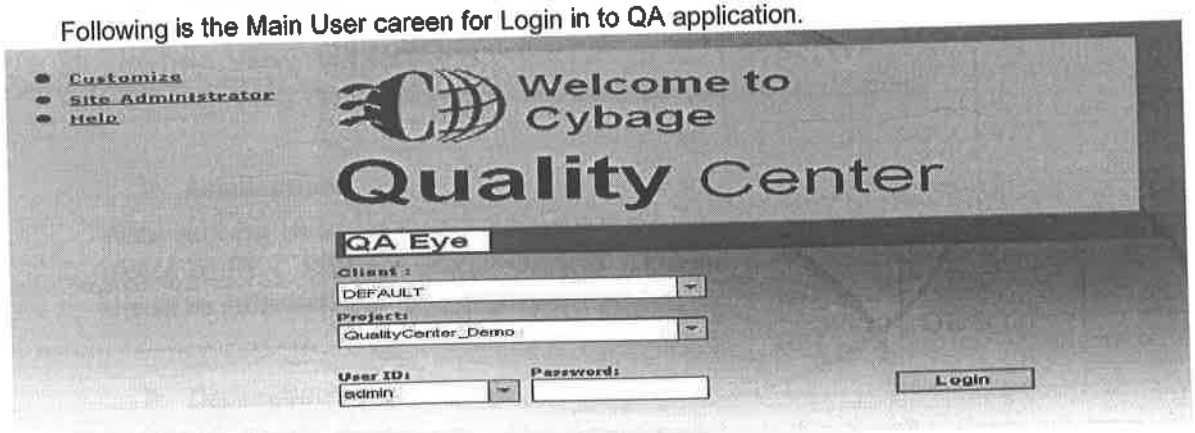


Fig 4.5: **LDAP Authentication**

➢ **User Interface**

Following is the Main User careen for Login in to QA application.



➢ **Data Requirements**

In this component we need four basic tables via *Clients*, *Projects*, *Groups, and Users*. *User_Projects* table is used to maintain the relationships between Users and Projects. Similarly *User_Groups* table is used to contain the association between a groups and users. For setting various privileges to the user we will use *Privileges* table. And *Groups_Privileges* table will store the set of privileges rules between users and groups.

We will also use PBN database for verification. The data present in PBN tables will be synchronized with the tables designed for QA Eye application.

➢ ***Design Diagram***

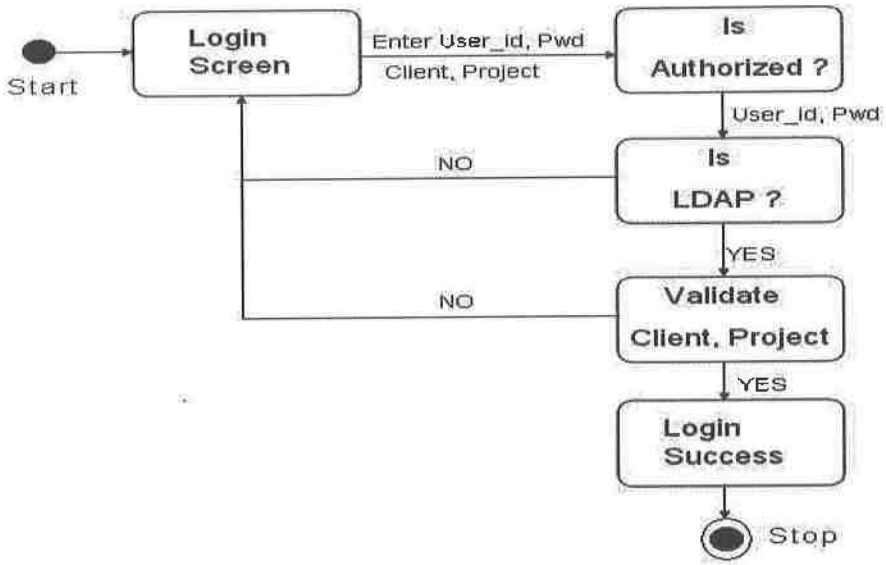


Fig 4.6: **State Diagram for Login**

➢ **Test Guidelines**

- Validate Username, Password text boxes.
- Validate drop down menu for Clients and Projects.
- Verify the Security.
- Verify the Relations between Clients/Projects/Users/Groups

➢ **Assumptions**

While verifying Users we have assumed that the data present in QA eye Database will be replica of PBN database. If any changes happens in PBN tables then that changes should be reflected in QA Eye database.

➢ ***Dependencies on other components***

All the four major components are dependent on Login functionality. Users will have access to all components if and only if they considered as authenticated users through Login component.

4.3.2.2. Administrator

- **Purpose**

Administrator is responsible for adding, modifying clients, projects, users and groups. He/She also set privileges to the User groups depending on their role in a project.

- **Technology being used**

ASP.net using c# and SQL Server, java script

- **Object Model and Interface**

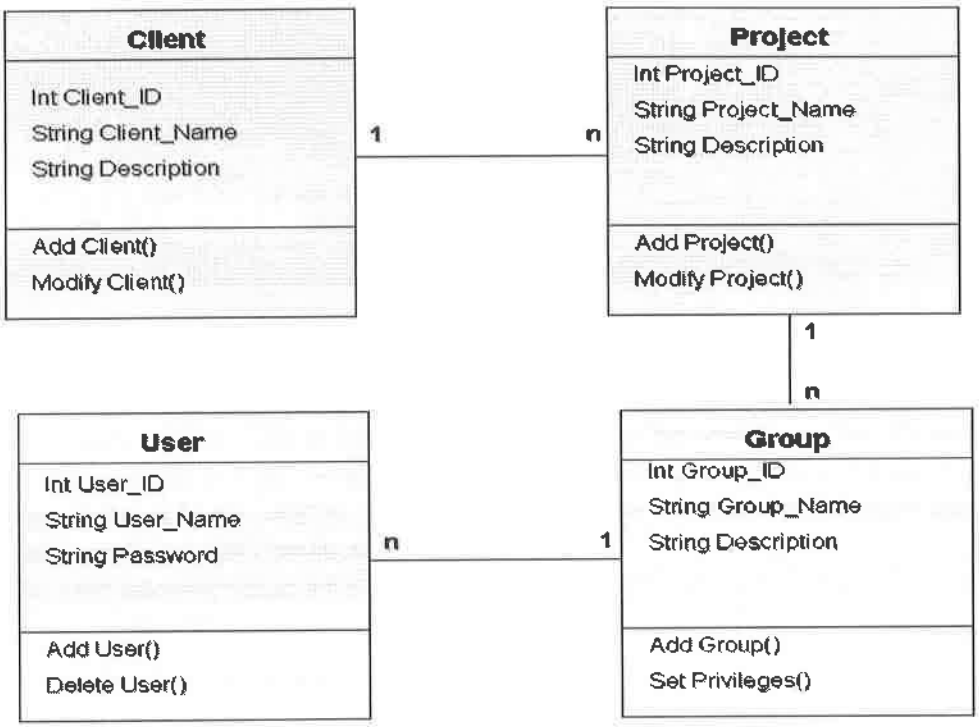


Fig 4.7: **Class diagram of Administrator**

As shown in above fig. Client is the Customer of the IT organization. This client can have multiple projects. Each Project is associated with number of groups depending up on the roles. A group consists of number of user having defined privilege to that users.

- ***User Interface***

Following is the main UI for Admin user. Here four links are provided to redirect for the respective page such as Manage Users, Manage Groups, Manage Clients, Manage Projects.

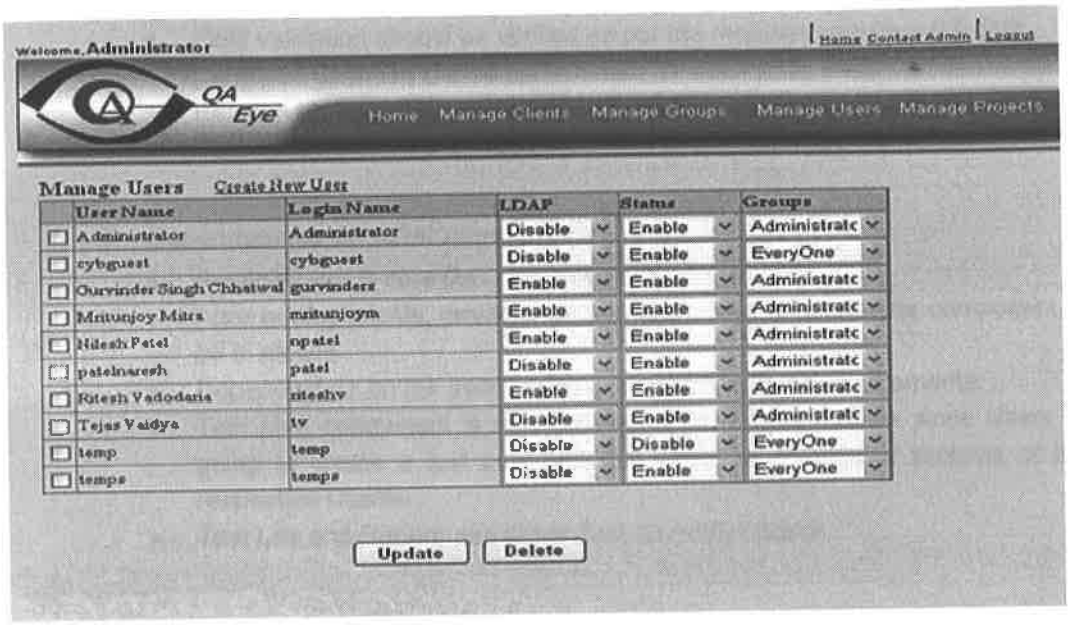
On same page there will be links associated for following operations:

Home : for redirecting to home page

Contact Admin : Writing mail to admin@cybage.com

Log Out : Logging out from the application.

➢ ***Administrator Welcome page***



➢ ***Data Requirements***

In this component we need four basic tables via *Clients*, *Projects*, *Groups, and Users*. *User_Projects* table is used to maintain the relationships between Users and Projects. Similarly *User_Groups* table is used to contain the association between a groups and users. For setting various privileges to the user we will use *Privileges* table. And *Groups_Privileges* table will store the set of privileges rules between users and groups.
Please refer following schema for more details:

➢ ***Design Diagram***

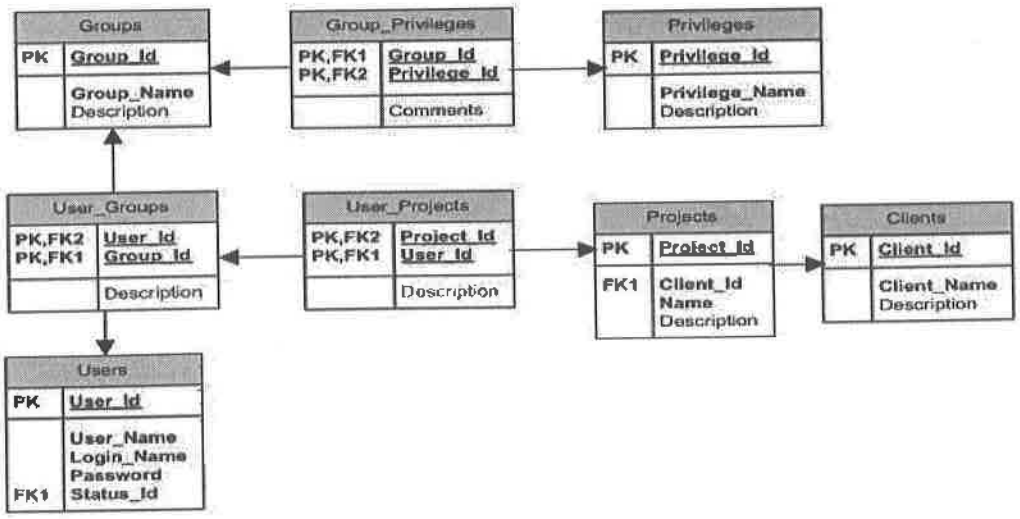


Fig 4.8: **Database of Administrator**

Test Guidelines

- Testing of Clients, Projects, Users and Groups web pages.
- Field validation should be verified as per the requirements for each field.
- Database updation should be validated for each table present in this module.
- Relation between Clients and Projects or Users and Groups should be tested.
- Testing the group's privileges according to the definition.

➢ ***Dependencies on other components***

- Administrator is core part of this product.
- After adding clients, projects, groups and users requirements component will be in picture.
- Depending up on the user's privileges he can add the requirements.
- Test plan component is also dependant on Admin module since Users are going to create a test plan and test cases in associated projects of their respective Clients.
- Test Lab and Reports are dependant on Administrator.

4.3.3. Requirements:

➢ ***Purpose***

By defining requirements, user can plan and manage tests that are more focused on business needs. Requirements are then linked to tests to provide complete traceability and aid the decision-making process.

➢ ***Technology being used***

ASP.net with c# and MS-SQL server, AJAX controls, java script etc.

➢ ***Object Model and Interface***

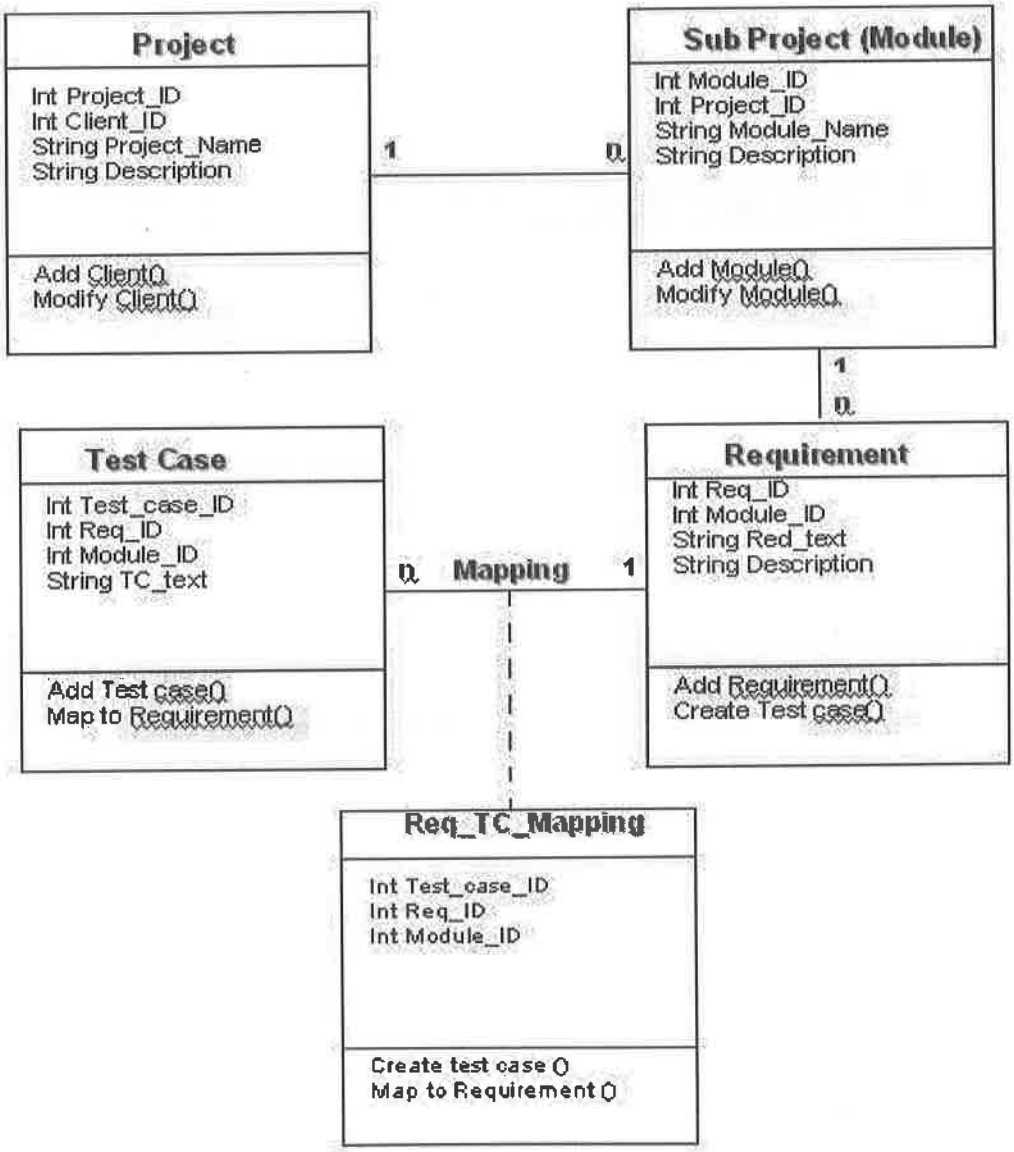


Fig 4.9: **Class Diagram of Requirements**

➢ ***User Interface***

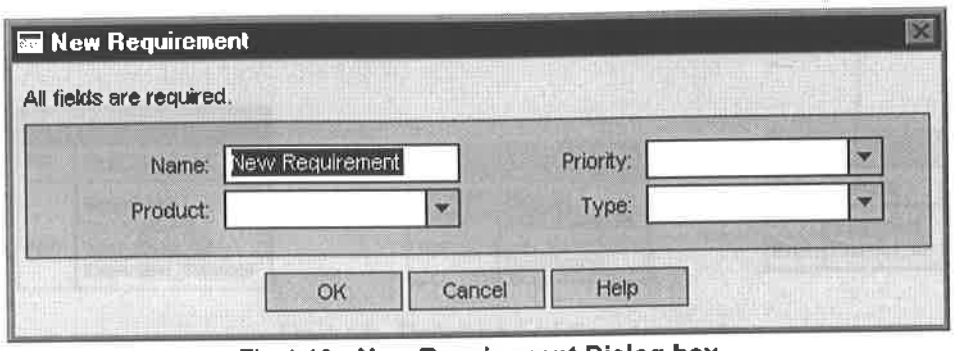


Fig 4.10: **New Requirement Dialog box**

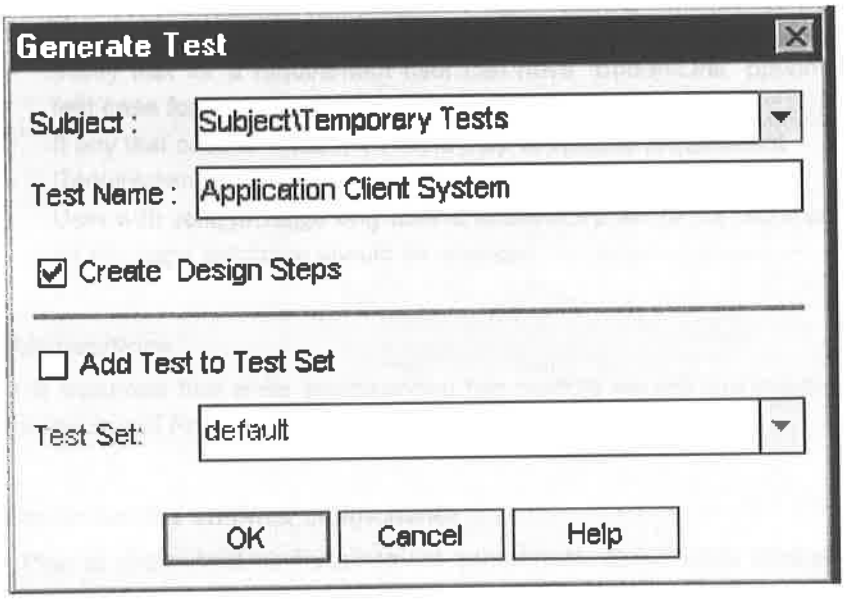


Fig 4.11: **Generating a Test from Requirements**

➢ ***Data Requirements***

Here we are mainly using *Requirements, Test cases, Req_TC_Mapping* tables. *Requirements* table is used to held the information about all the requirements which are associated to specific projects. For this project there is separate Test plan under which we will add test cases in *Test cases* table depending on defined requirements for this projects. Finally *Req_TC_Mapping* table will hold the linking between a requirement and associated test case.

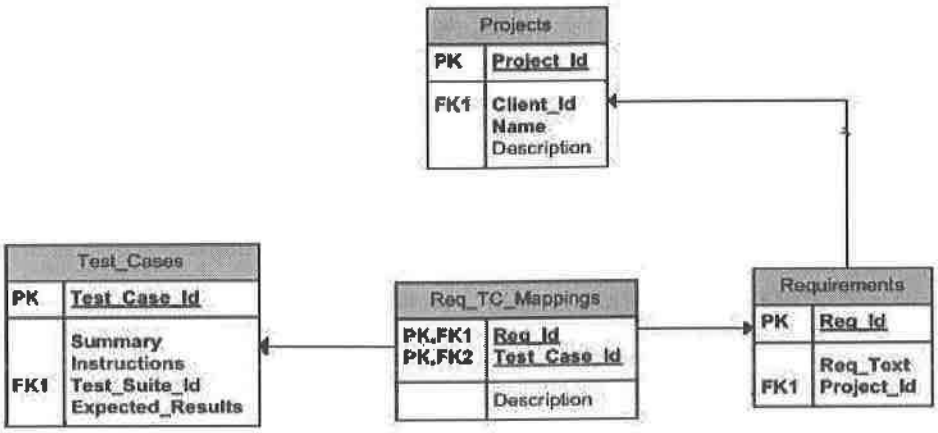


Fig 4.12: **Database of Requirements**

➢ ***Test Guidelines***

- Verify that all the requirements are associated with the defined projects
- Verify that for a requirement user can have "Button/Link" option to create a test case for it.
- If any test case is written, it should map to specific requirement
- Requirement should be present in tree structure
- User with valid privilege only able to add/modify/delete the requirements.
- All the page validation should be checked.

➢ ***Assumptions***

It is assumed that while implementing this module we will use existing database for the use of Projects, Modules definition.

➢ ***Dependencies on other components***

Test Plan is dependent on Requirement component. Since while creating test plan testers should be aware of the QA project requirements. Using the data deposited from this components tester can create test cases. Those test cases can also be linked to the requirements.

4.3.4. Test Plan and Test Lab

➢ ***Purpose***

In this module the creation of test cases will occur and these test cases will be pulled in Test Lab for execution. The results will be updated in database.

➢ ***Object Model and Interface***

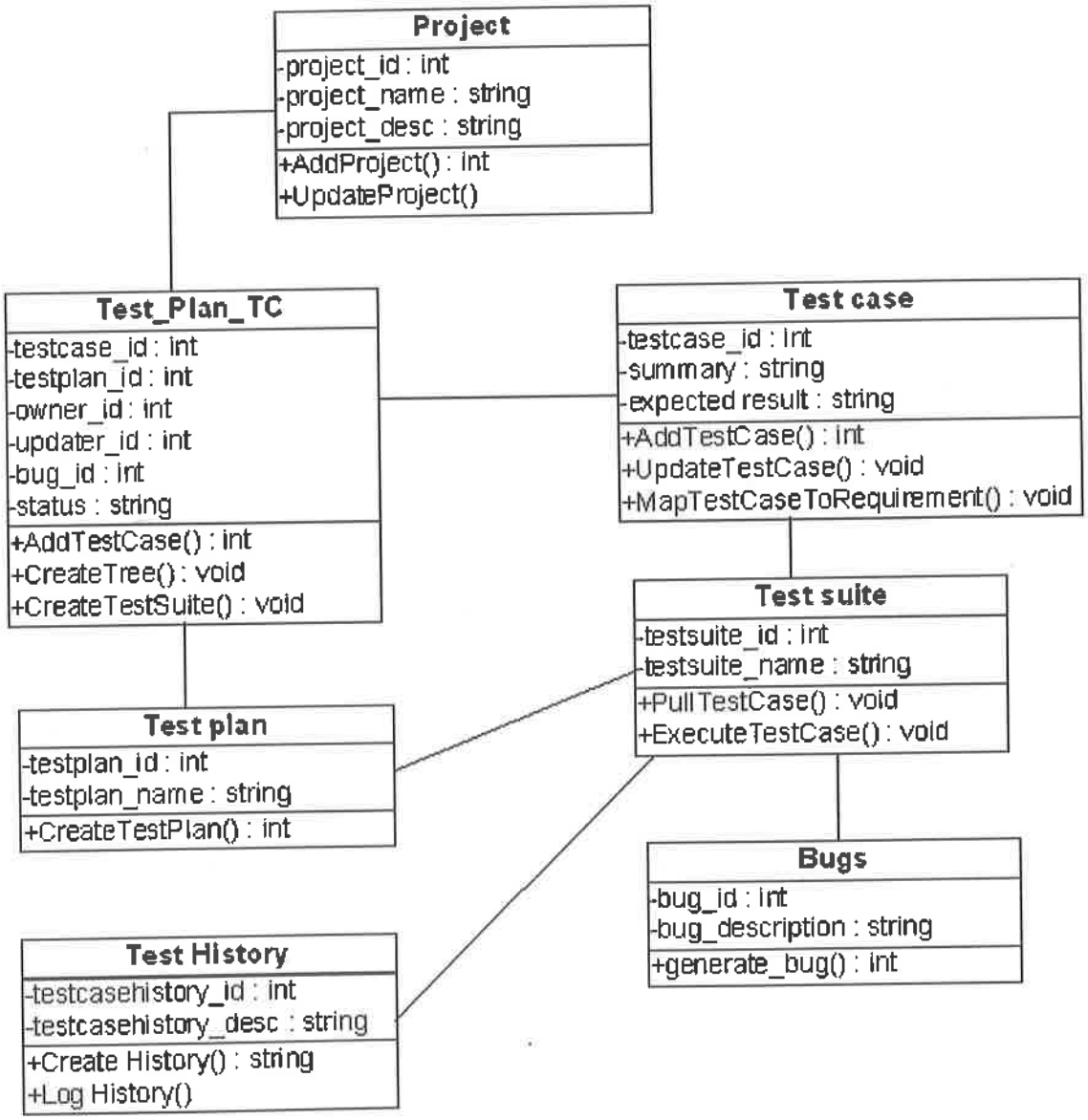


Fig 4.13: **Class diagram of Test Lab and Test Plan**

➢ ***Use Case Diagram (Manager- Operations)***

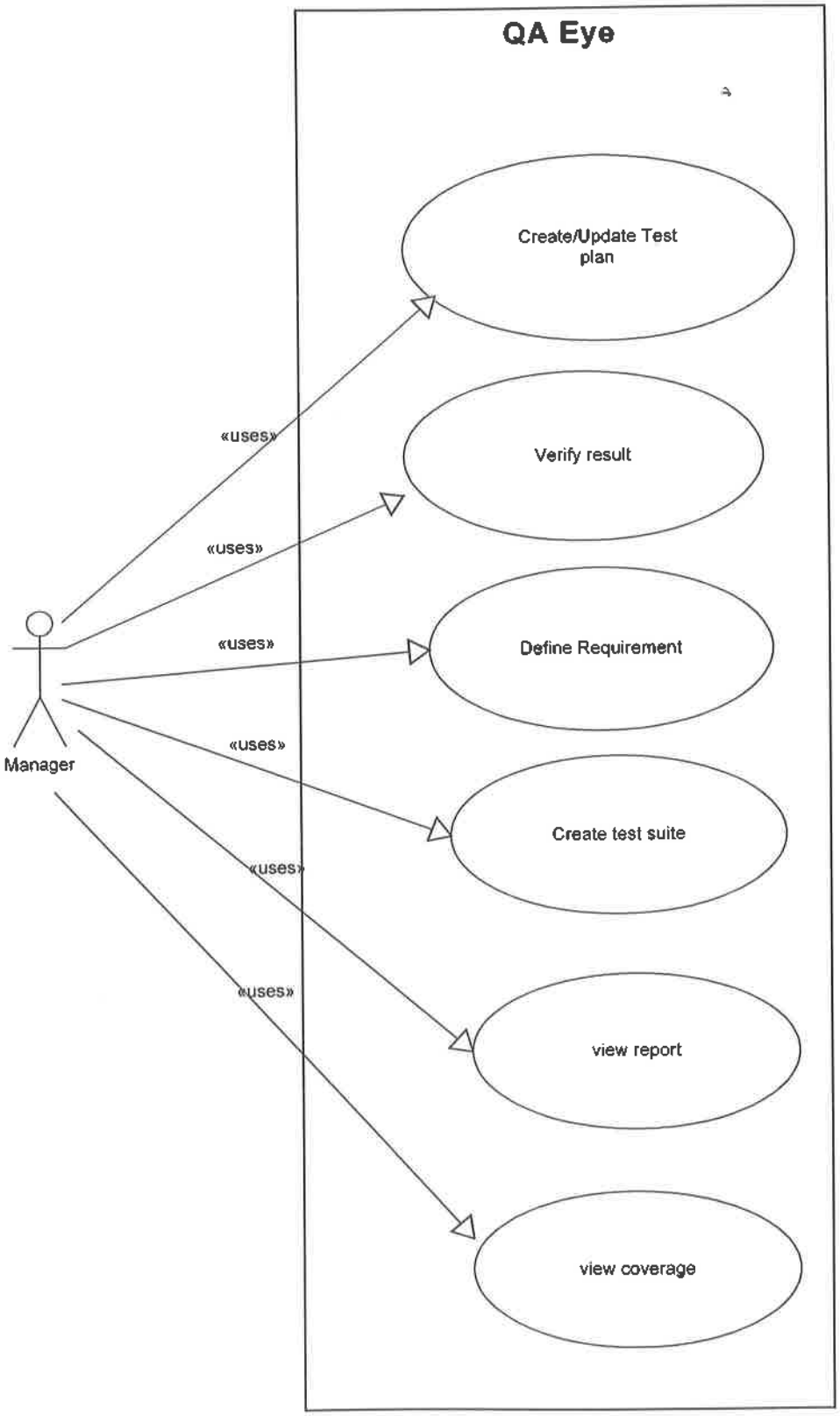


Fig 4.14: **Use Case diagram of Manager**

➢ ***Sequence Diagram : (Requirement/ Test Plan/ Test Lab)***

Following is the Sequence diagram of Test plan and Test lab module. These modules are depends on Requirements and Administrator. The data flow between these components are shown below.

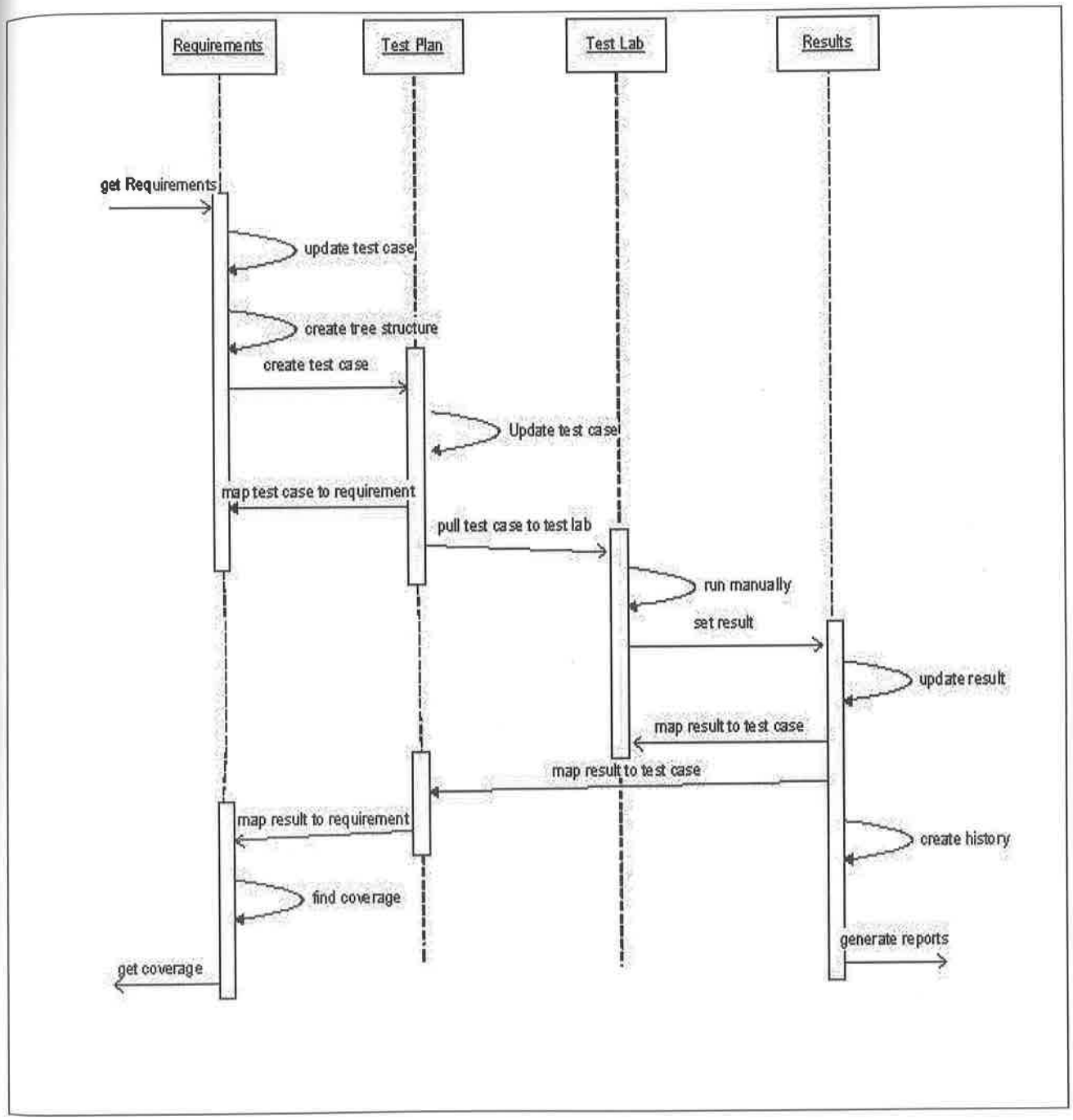


Fig 4.15: **Sequence diagram of** (Requirement/ Test Plan/ Test Lab)

➢ ***Sequence Diagram : (Projects/Users/Groups)***

As shown in this sequence diagram users privileges are defined by Admin user. Checking with this permissions user is allowed to carry Test plan and Test lab operations. This operations includes create/update test plan, create/update test lab, update results, view reports etc.

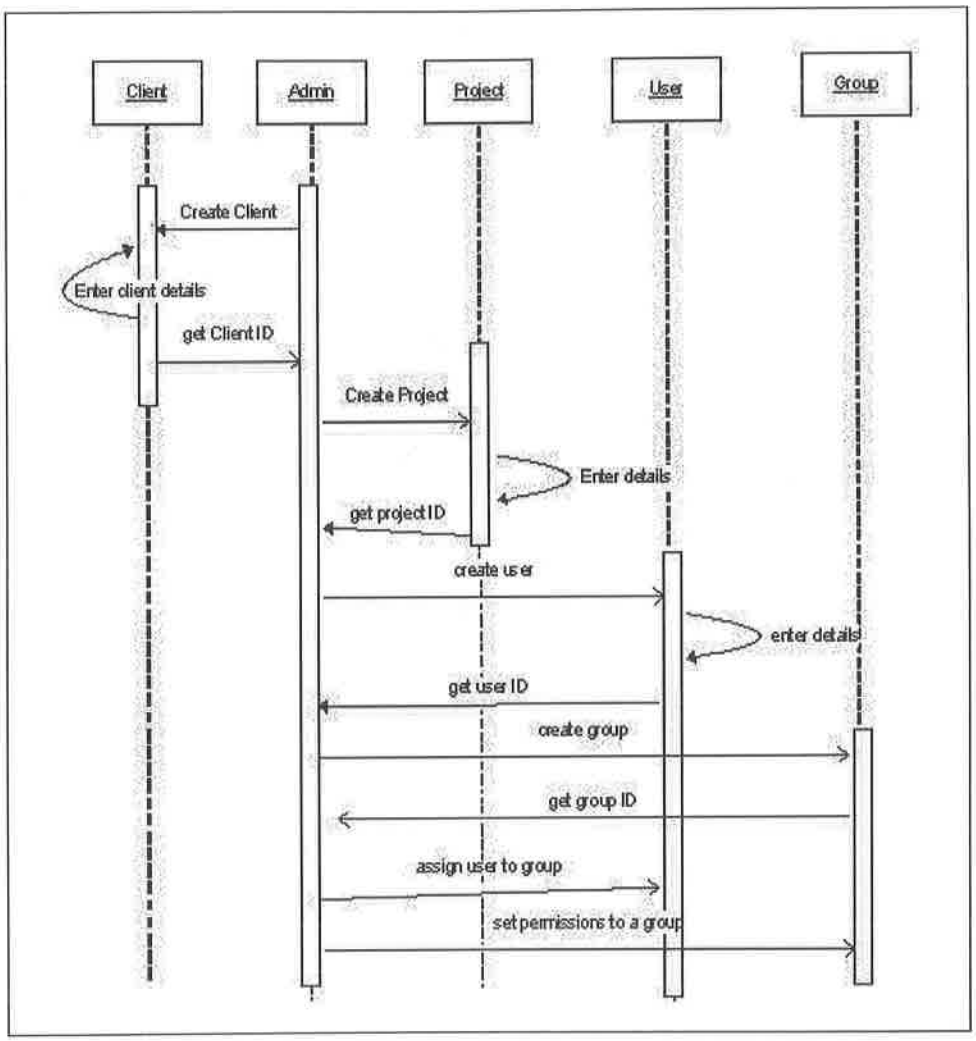


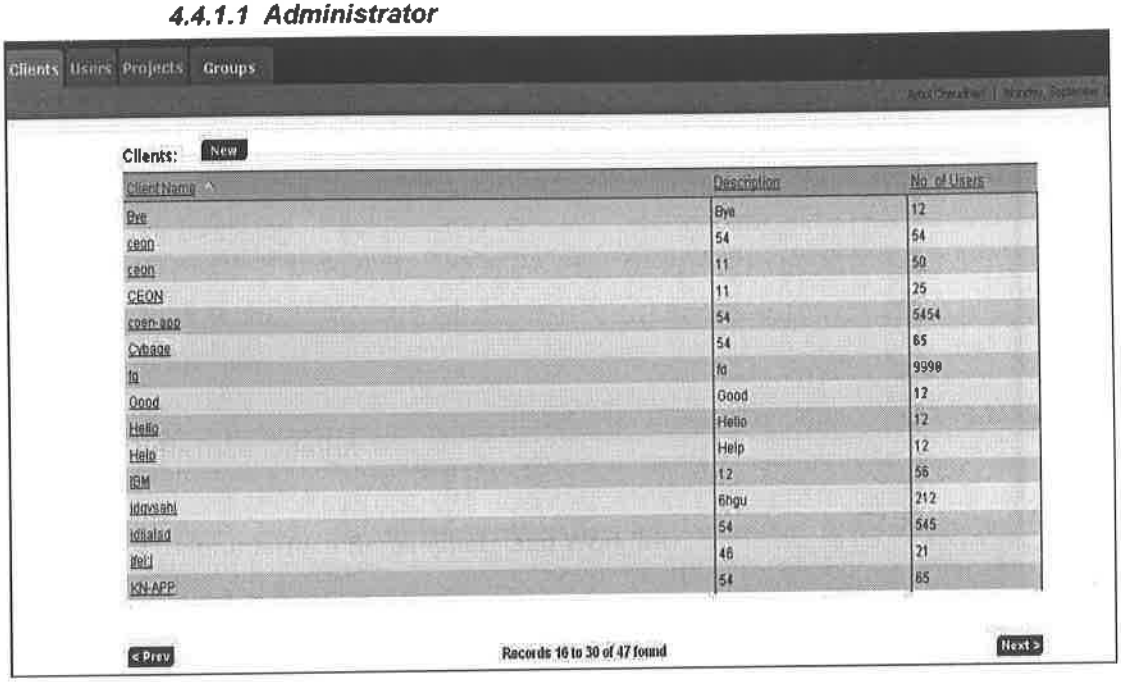
Fig 4.15: **Sequence diagram of** (Projects/Users/Groups)

Client_ID, User_ID, Group_ID, Privilege_ID are the keys which are used to establish the relationships between dependant components. After this operation respective user is authorized to his/her operations.

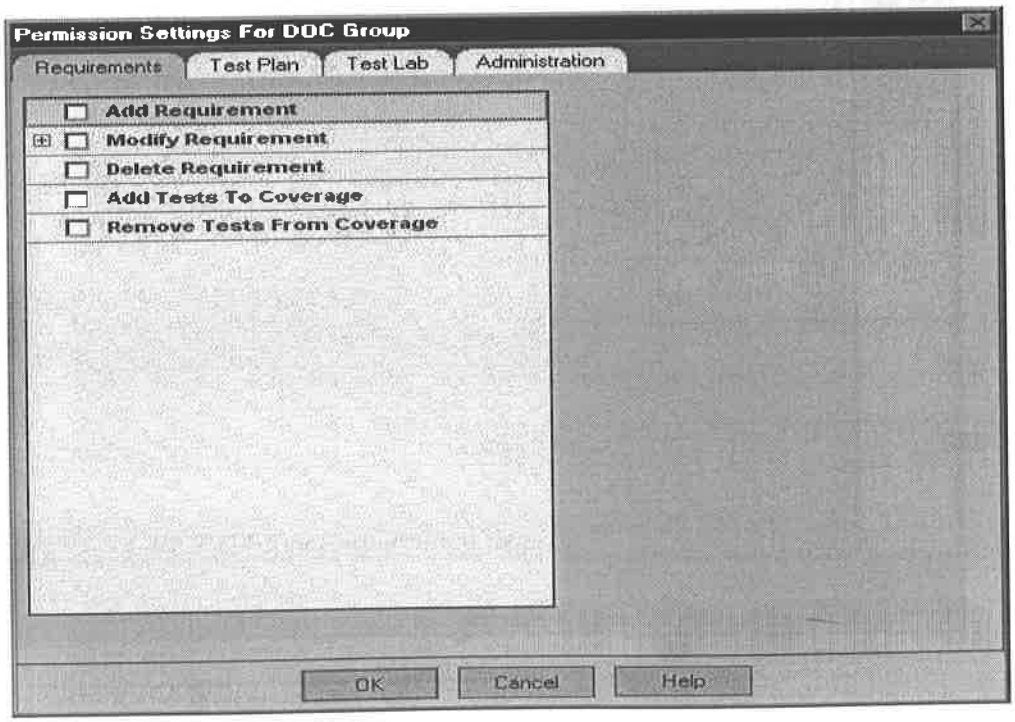
4.4. User Interface Design

4.4.1. User Interface Screenshots and Workflow

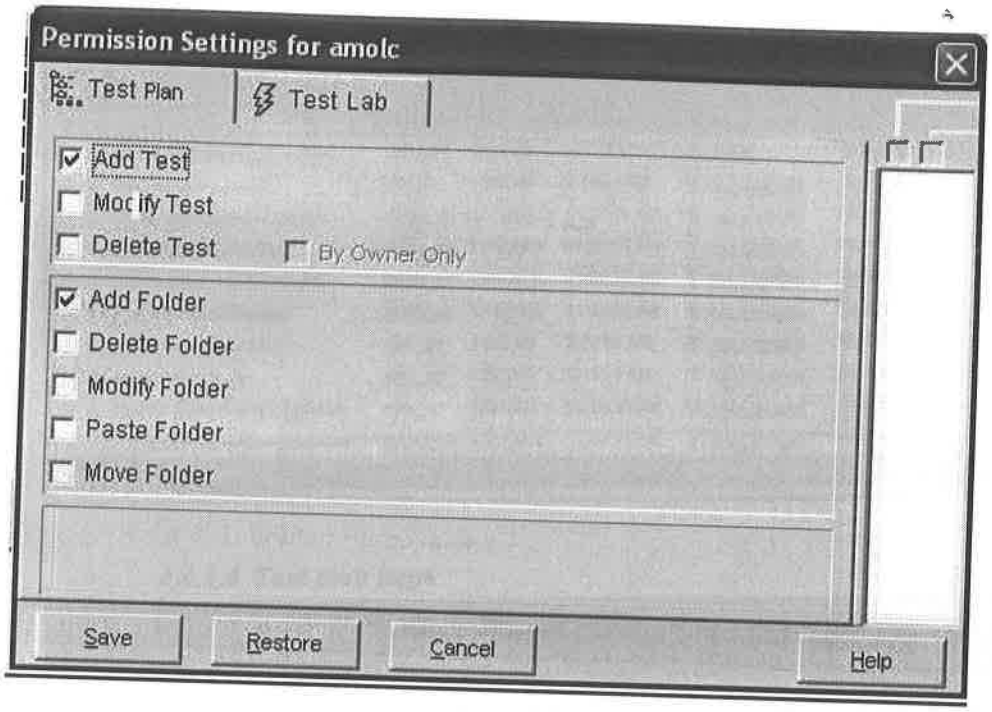
4.4.1.1 Administrator



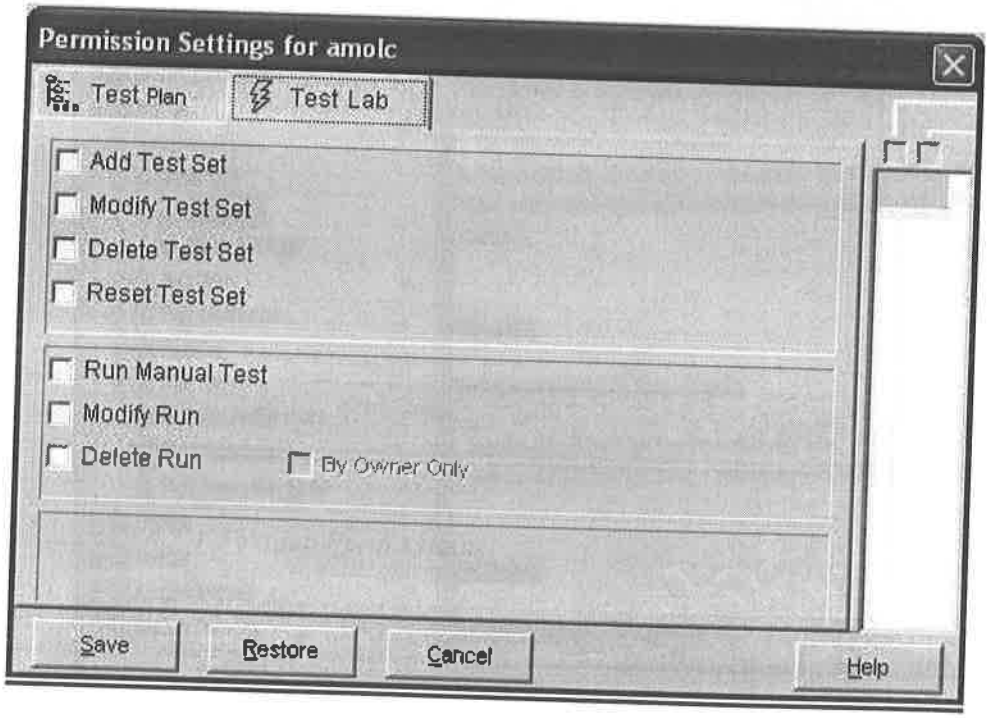
4.4.1.2 QA Eye Requirement page



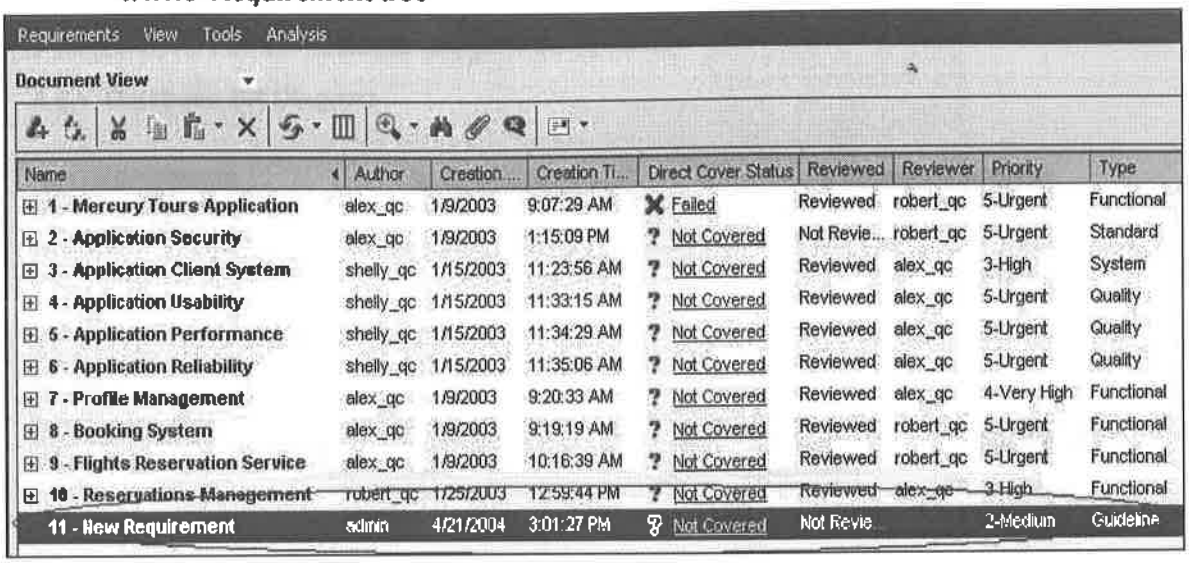
4.4.1.3 Test plan Permissions



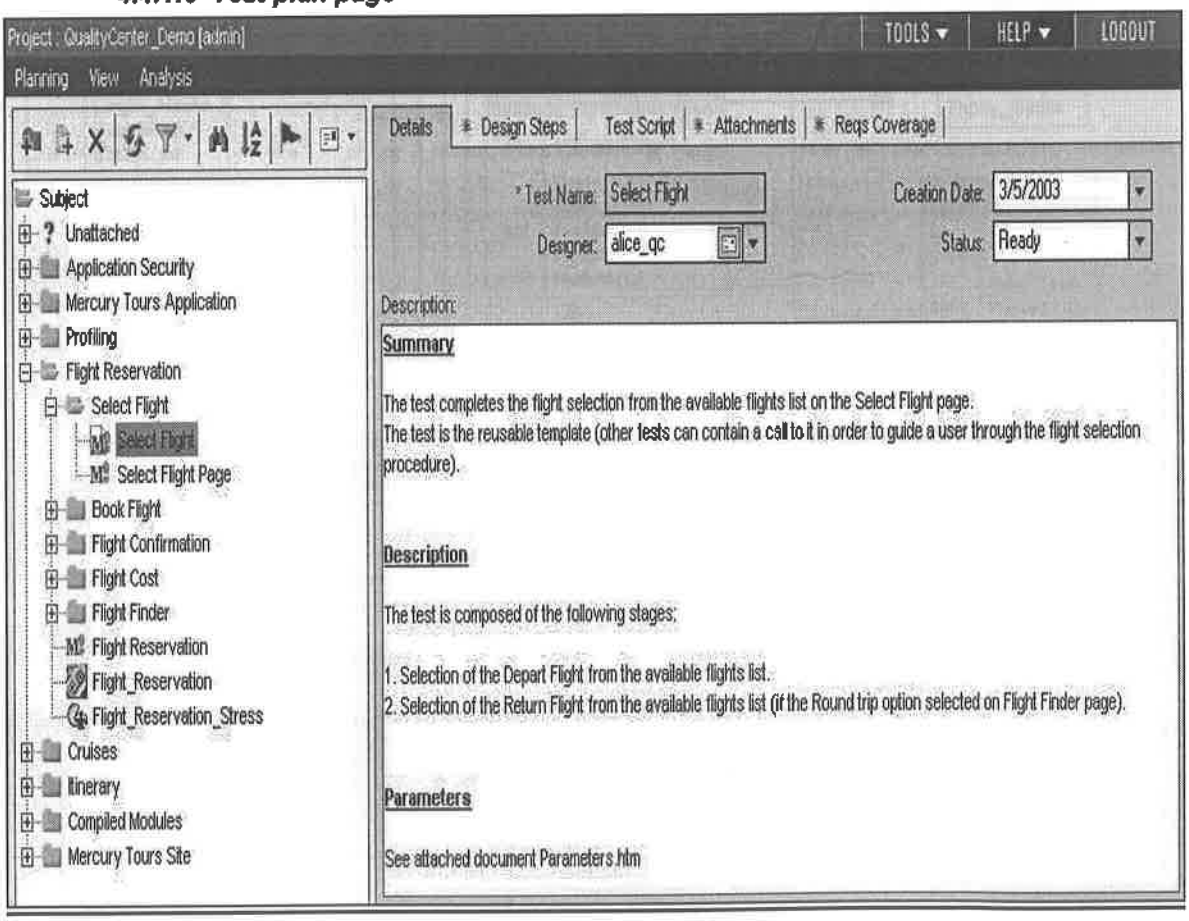
4.4.1.4 Test lab Permissions



4.4.1.5 Requirement tree



4.4.1.6 Test plan page



4.5. Database Design

This section covers all the database design of the system.

4.5.1. Database ER Diagram

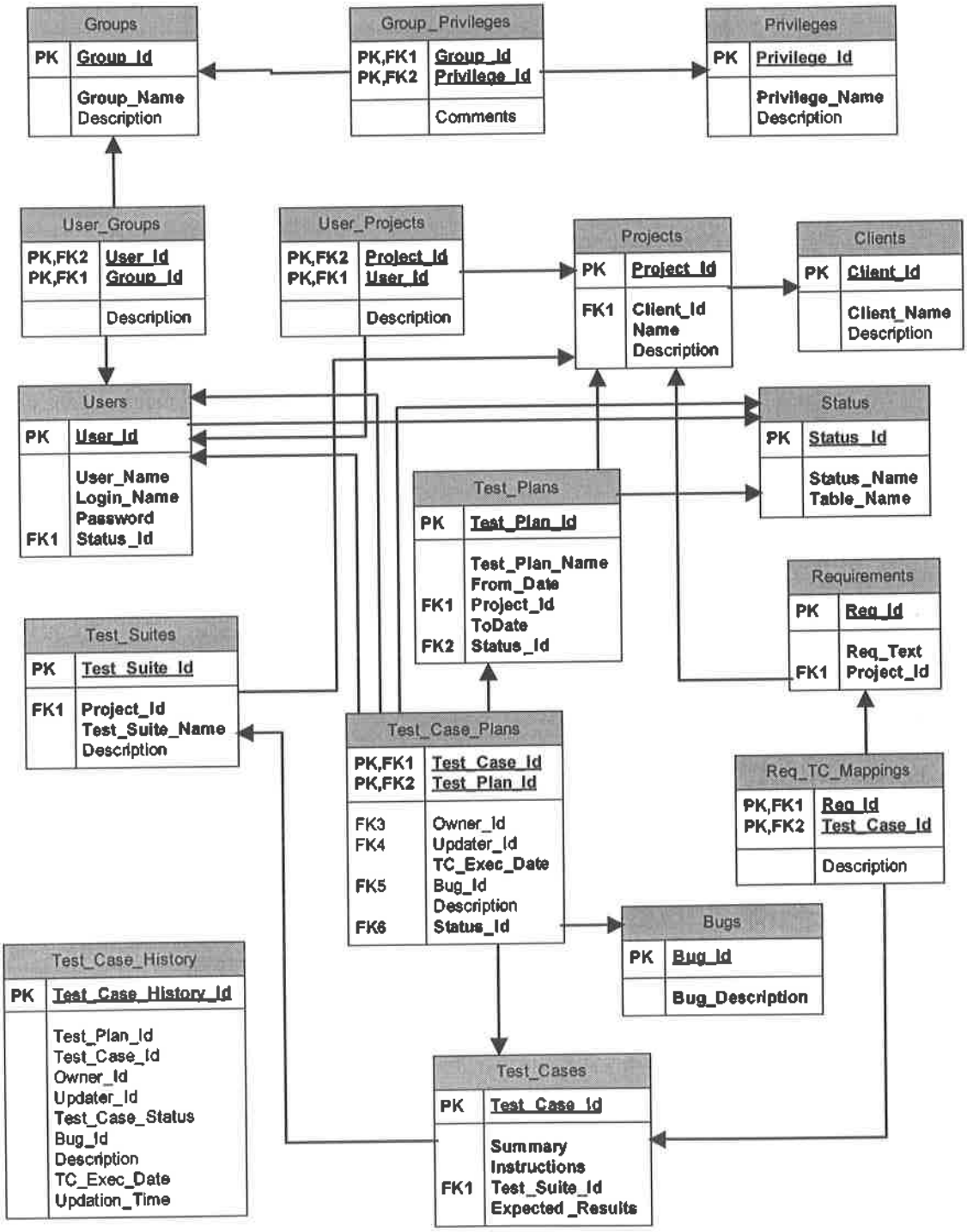


Fig 4.16: **Database Design of QAEye**

4.5.2. Tables Descriptions

- ***Clients:***

Description:

	Column Name	Data Type	Length	Allow Nulls
🔑	Client_Id	int	4	
	Client_Name	varchar	100	
	Description	varchar	250	✓

- ***Projects:***

Description:

	Column Name	Data Type	Length	Allow Nulls
▶🔑	Project_Id	int	4	
	Name	varchar	100	
	Defect_Base_URL	varchar	250	
	Client_Id	int	4	
	Description	varchar	250	✓

- ***Users:***

Description:

	Column Name	Data Type	Length	Allow Nulls
▶🔑	User_Id	int	4	
	User_Name	varchar	100	
	Login_Name	varchar	20	
	Password	varchar	20	
	Email_id	varchar	100	
	LDAP	int	4	
	Status_Id	int	4	

- ***Groups:***

Description:

	Column Name	Data Type	Length	Allow Nulls
▶🔑	Group_Id	int	4	
	Group_Name	varchar	100	
	Description	varchar	250	✓

- ***Requirements:***

Description:

	Column Name	Data Type	Length	Allow Nulls
🔑	Req_Id	int	4	
	Req_Text	varchar	4000	
	Project_Id	int	4	

- ***Test_Cases:***

Description:

	Column Name	Data Type	Length	Allow Nulls
🔑	Test_Case_Id	int	4	
	Summary	varchar	1000	
	Owner_Id	int	4	
	Test_Case_Updatetim	int	4	✓
	Test_Suite_Id	int	4	

- ***Test_Suites:***

Description:

	Column Name	Data Type	Length	Allow Nulls
🔑	Test_Suite_Id	int	4	
	Test_Suite_Name	varchar	100	
	Project_Id	int	4	
	Description	varchar	250	✓

- ***Test_Plans:***

Description:

	Column Name	Data Type	Length	Allow Nulls
🔑	Test_Plan_Id	int	4	
	Test_Plan_Name	varchar	100	
	FromDatetime	datetime	8	
	ToDatetime	datetime	8	
	Status_Id	int	4	
	Project_Id	int	4	

- ***User_Groups:***

Description:

	Column Name	Data Type	Length	Allow Nulls
🔑	Group_Id	int	4	
🔑	User_Id	int	4	
	Description	varchar	250	✓

- ***User_Projects:***

Description:

	Column Name	Data Type	Length	Allow Nulls
🔑	User_Id	int	4	
🔑	Project_Id	int	4	
	Description	varchar	250	✓

- ***Test_Results:***

Description:

	Column Name	Data Type	Length	Allow Nulls
▶	Test_Plan_id	numeric	9	✓
	Test_Case_id	numeric	9	✓
	Result	char	10	✓
	Comment	char	10	✓
	Bug_Id	numeric	9	✓
	Result_Date	datetime	8	✓

- ***Test_Case_History:***

Description:

	Column Name	Data Type	Length	Allow Nulls
🔑	Test_Case_History_Id	int	4	
	Test_Plan_Id	int	4	✓
	Test_Case_Id	int	4	✓
	Updatetimer_Id	int	4	✓
	Test_Case_Status_Id	int	4	✓
	Bug_Id	int	4	✓
	TC_Exec_Datetime	datetime	8	✓
	Updation_Time	datetime	8	✓
	Test_Case_Step_Id	int	4	✓
	Action_Id	int	4	

- ***Bugs:***

Description:

	Column Name	Data Type	Length	Allow Nulls
🔑	Bug_Id	int	4	
	Bug_Description	varchar	250	✓
	Bug_No	decimal	9	

- ***Test_Case_Plans:***

Description:

	Column Name	Data Type	Length	Allow Nulls
🔑	Test_Case_Id	int	4	
🔑	Test_Plan_Id	int	4	
	TC_Exec_Datetime	datetime	8	
	Status_Id	int	4	
	TC_Status_Updatetim	int	4	
	Bug_Id	int	4	✓
	Failed_Test_Step_Id	int	4	✓
	Comment	varchar	250	✓

- ***Test_Case_Steps:***

Description:

	Column Name	Data Type	Length	Allow Nulls
🔑	Test_Case_Id	int	4	
🔑	Test_Case_Step_Id	int	4	
	Instructions	varchar	4000	
	Expected_Results	varchar	4000	

- ***Group_Rights:***

Description:

	Column Name	Data Type	Length	Allow Nulls
🔑	Group_Id	int	4	
🔑	Right_Id	int	4	
	Comments	varchar	250	✓

- ***Req_TC_Mappings:***

Description:

	Column Name	Data Type	Length	Allow Nulls
🔑	Req_Id	int	4	
🔑	Test_Case_Id	int	4	
	Description	varchar	250	✓

- ***Rights:***

Description:

	Column Name	Data Type	Length	Allow Nulls
🔑	Right_Id	int	4	
	Right_Name	varchar	100	
	Description	varchar	250	✓

- ***Status:***

Description:

	Column Name	Data Type	Length	Allow Nulls
🔑	Status_Id	int	4	
	Status_Name	varchar	100	
	Table_Name	varchar	30	

4.5.3. Database Object Description – Stored Procedures, Triggers etc

Following are the stored procedures used are building the application.

- ***Creating new Client:***

```
CREATE PROCEDURE new_client
@name varchar(100) , @description varchar(250)
 AS
declare @clientID int
begin
      select  @clientID =   ISNULL ((max(client_ID) +1 ),1) from clients
                  insert into clients values (@clientID,@name,@description)
end
GO
```

- ***Add Requirements:***

```
CREATE PROCEDURE addrequirements
@text varchar(400) , @prID int
AS
declare @cnt int
begin
select @cnt= ISNULL(  (max(req_id) +1) ,1)  from requirements
insert into requirements values (@cnt,@text,@prID)
end
GO
```

- ***Browse testsuite and delete test case:***

```
CREATE PROCEDURE browse_testsuite_deletetestcase
@tc int
AS
DECLARE @reqid int
begin
    if not  exists (select test_case_id from test_case_plans where test_case_id =
@tc)
begin
            print 'deteting'
            delete from req_tc_mappings where test_case_id = @tc
    end
            delete from test_cases where test_case_id = @tc
    print @@error
    if (@@error <> 0)
            delete from test_case_steps where test_case_id = @tc
    end
GO
```

- ***Duplicate test case:***

```
CREATE PROCEDURE duplicate_testcase  @tc int
AS
declare @cnt as int ,  @summary varchar(400), @owner int , @updater int , @ts int ,
@stepid int , @step varchar(4000) , @Exp_result varchar(4000)
begin
select                                  @summary=summary,@owner=owner_id,
@updater=test_case_updatetimer_id,  @ts=test_suite_id  from  test_cases  where
test_case_id= @tc
select @cnt=(max(test_case_id) +1) from test_cases
insert into  test_cases values (@cnt,@summary,@owner,@updater,@ts)
    declare  cur_copysteps  cursor for
    select Test_Case_Step_Id,Instructions, Expected_Results from test_case_steps
where test_case_id = @tc
                open cur_copysteps
                while(1=1)
begin
                        fetch     next     from     cur_copysteps     into
@stepid,@step,@exp_result
                        if @@fetch_status <> 0
                            break;
                        insert      into      test_case_steps      values
(@cnt,@stepid,@step,@exp_result)
                        end
close cur_copysteps
end
GO
```

- ***Map Test case with requirement:***

```
CREATE proc mapTestcase_with_requirement
@reqid int ,
@testsuiteID int  ,
@owner int ,
@reqtext varchar (250)
As
declare @tcID int

select @tcId = isnull(max(test_case_id)+1 ,1) from test_cases

insert into test_cases values (@tcid , @reqtext  , @owner , @owner , @testsuiteID )
insert into req_tc_mappings values (@reqid,@tcID,' ')
GO
```

- ***Relocate test case to other suite:***

```
CREATE PROCEDURE relocate_testcase_to_other_suite
@tc int  , @ts int
AS
declare @cnt as int ,   @summary varchar(400), @owner int , @updater int , @stepid
int , @step varchar(4000) , @Exp_result varchar(4000)

begin
select  @summary=summary,@owner=owner_id,  @updater=test_case_updatetimer_id
from test_cases where test_case_id= @tc
select @cnt=(max(test_case_id) +1) from test_cases
insert into  test_cases values (@cnt,@summary,@owner,@updater,@ts)
     declare  cur_copysteps  cursor for
     select Test_Case_Step_Id,Instructions, Expected_Results from test_case_steps
where test_case_id = @tc
                    open cur_copysteps
                          while(1=1)
                          begin
                          fetch      next      from      cur_copysteps      into
@stepid,@step,@exp_result
                          if @@fetch_status <> 0
                               break;
                          insert      into      test_case_steps      values
(@cnt,@stepid,@step,@exp_result)
                          end
close cur_copysteps
end
GO
```

➢ ***Test case steps add:***

```
CREATE PROCEDURE testcase_steps_add
@tc int , @step varchar(20) , @desc varchar(20)
 AS
declare @cnt_step as int

begin
 (select  @cnt_step  =  ( ISNULL(max (stepno),0) +1)  from testcase_steps where
testid = @tc)
print @cnt_step
if exists ( select testID from testcase where testId = @tc )
        begin
        insert into testcase_steps values  (@tc,@cnt_step,@step,@desc)
        end
else
        begin
        insert  into  testcase  values  (@tc,1,15,  'Some  summary  STORED
PROCEDURE','1-apr-2002',1)
        insert into testcase_steps values  (@tc,@cnt_step,@step,@desc)
end
end
GO
```

5. Software Test Plan

5. Software Test Plan

5.1. Introduction

This document is a procedural guide for listing the testing activities that should be carried out for the **QA Eye** Project. It describes the software test environment for testing, identifies the tests to be performed, and provides schedules for test activities.

5.1.1. Purpose of the Document

The Purpose and objective of STP is:

- Identify all the activities involved in testing,
- Resources required to execute testing activities and monitoring mechanisms.
- It also explains the strategy and approach for testing the software components and the defect tracking system to be used.

5.1.2. Scope of STP

The scope of Software Test Plan (STP) is to document the testing plan of QAEye project.

5.2. Scope of Testing

5.2.1. Features to be tested

- ***Login:***

Sr. No.	Features / Functions to be tested
1.	Verify the Client test box and
2.	Verifying the Authenticated user only can log in
3.	Verifying unauthorized user can not log in
4.	Verifying general UI validation of text boxes
5	Verifying LDAP logins

- ***Administrator:***

Sr. No.	Features / Functions to be tested
1.	Verifying Manage Clients
2.	Verifying Manage Projects
3.	Verifying Manage Users
4.	Verifying Manage Groups and Permissions

- ***Requirements:***

Sr. No.	Features / Functions to be tested
5.	Verify creating a Requirements
6.	Verify finding Requirements
7.	Verify viewing the Requirements
8.	Verify mailing Requirements
9.	Verify viewing Associated Defects
10.	Verify modifying the Requirements Tree
11.	Verify creating Tests from Requirements
12.	Verify Viewing Requirement History

- ***Test Plan:***

Sr. No.	Features / Functions to be tested
13.	Verify creating a Test Plan Tree
14.	Verify adding Tests to a Test Plan Tree
15.	Verify viewing the Test Plan Tree
16.	Verify associating Defects with a Test
17.	Verify mailing Tests
18.	Verify finding Tests in the Tree
19.	Verify sorting a Test Plan Tree
20.	Verify modifying a Test Plan Tree
21.	Verify linking Tests to a Requirement
22.	Verify designing Test Steps
23.	Verify managing Test Steps
24.	Verify linking Requirements to a Test

- ***Test Lab:***

Sr. No.	Features / Functions to be tested
25.	Verify adding a Test Set
26.	Verify adding Tests to a Test Set
27.	Verify setting the Test Set Notifications
28.	Verify setting the Test Set On Failure Rules
29.	Verify managing Test Sets
30.	Verify modifying a Test Run Schedule
31.	Verify finding Tests
32.	Verify viewing the Execution Flow Diagram
33.	Verify arranging Tests Sequentially

➢ ***QA Reports:***

Sr. No.	Features / Functions to be tested
34.	Verify Reports

➢ ***Features not to be tested***

Sr. No.	Features / Functions not to be tested
1.	For First Release Reports are not to be tested

5.3. Test Environment

This section refers to testing environment and Client supplied Hardware / Software.

5.3.1. Hardware

This section states all the hardware resources required for the project.

Sr. No.	Purpose	Configuration	Quantity	Critical (Yes / No)
5.	Client	Desktop Pentium III m/c, 256 MB RAM, 40 GB Hard Disc	4	
6.	Web server	P IV 2.4 Ghz, 512 MB RAM, 2 x 40 GB Hard Disk	1	
7.	Database Server	1 x P IV 2.4GHz, 1 GB RAM, 2 x 40GB IDE HDD	1	
8.	Application Server	Win2k Server, NET, source offsite, Visual Source Safe, IIS	1	

Table 5.1: **Hardware for QA**

5.3.2. Software

This section states all the software and specific languages needed for the project.

Sr. No.	Name of the software	Version/release details	No. Of licenses
1.	.net Framework	1.1	
2.	SQL Server	2000	1
3.	Visual Studio .Net	2003	3
4.	Sea-pines QA Wizard	4.0	1

Table 5.2: **Software for QA**

5.4. Test Strategy

5.4.1. Testing Process

Testing process to be followed:
The testing process highlights the broad-level phases to be executed. Each of these phases has series of steps to be executed. The various broad-level phases have been described below:

1. Identify the requirements to be tested. All test cases are derived using the current Design Specification.
2. Identify the expected results for each test.
3. Identify the testing-related equipments and reference document that are required to execute the testing Process. Setup the test environment.
   - QA Lead establishes the test environment, which includes hardware and software, as per the requirements.
4. Test Design
   - QA Engineer prepares the test cases.
   - QA Engineer prepares the test data/test scripts (in case of automated tools, or/ guidelines.
   - QA Engineer prepares test setup.
   - QA Lead reviews and approves test cases.
5. Test Execution
   - QA Engineer executes test cases and updates the actual results in test cases.
6. Test Report
   - QA Engineer creates test report for each build.
   - QA Engineer prepares test summary report.

5.4.2. Build Process

The following is the Build Process to be followed for the QA Eye.

- Build Engineer receives Build.
- Build Engineer identifies Build.
- Build Engineer creates Build.
- Build Engineer sends mail for build notification.

5.4.3. Type of Testing

Along with the type of testing also mention the approach to be followed for the testing, that is. Manual Testing or Automated Testing. Use Automated Testing Plan for planning automation activities in details.
The different types of testing that may be carried out in the project are as follows:

- Smoke Testing
- Integration Testing
- System Testing
- Regression Testing
- Load Testing
- Stress Testing

The different types of testing that may be carried out in the project are as follows:

- **Smoke testing:**

Smoke test is an initial set of tests that determine if a new software build is performing well enough to accept it for a major testing effort. It verifies the major functionality at high level. The Smoke test scenarios emphasize more on breadth than the depth. If the test fails, the build is returned to developers without testing.

- **Integration testing**:

Integration testing is performed to establish whether the components correctly interact with each other according to the specifications.

- **System Testing**:

It tests the system as a whole. It is a functional testing, performed to validate that the application meets requirement specifications.

- **Regression Testing**:

It is re-testing of a program after doing the modification. It helps in ensuring that faults have not been introduced or uncovered as a result of the changes made, and that the modified system still meets its requirements. It is performed whenever the software or its environment is changed.

- **Load Testing**:

It involves testing an application under heavy loads so that you can predict its working under different user loads. It also determines the point at which the systems response time degrades or fails.

- **Stress Testing**:

It is a testing conducted to evaluate a system or component at or beyond the limits of its specified requirements (for example, extreme processor utilization, insufficient memory, inadequate hardware, dependency on over-utilized shared resources).

5.5. Defect Tracking

For this QAEYE project we are using "**TTWeb**" for Defect tracking. QA will log new issues into the TTWeb which will be assigned to concerned Developer.

5.6. Resource Management

5.6.1. Roles and Responsibilities

Sr. No.	Roles	Responsibilities
1.	QA Lead	- Creating Test plan - Assigning various modules to QA - Review the Test cases - Sending WSR to PM
2.	QA Engineer	- Understanding the Requirements - Writing the test cases for it - Executing the test cases - Updating the Results - Sending BAR and TSR.

Table 5.3: **QA Roles**

5.7. Assumptions

- All functional requirements are properly defined and meet users' needs.
- The developers performs adequate unit testing before sending modules to QA.
- The Developers fix all the defects identified during the unit testing prior to integration and system testing. Else the defects should be mentioned in the release notes.
- The application should be delivered on the expected delivery date according to the schedule. Delivery and downtime delays shall cause adjustments in the test schedule and can become a risk for the on-time product delivery.
- QA team should be involved in initial project discussions and should have a working knowledge of the proposed production system prior to integration and system testing.
- Change control procedures are followed.
- The number of test cases that have a direct impact upon the amount of time it takes to execute the test plan.
- During the test process, all required interfaces are available and accessible in the QA environment.
- Testing occurs on the most current version of the build in the QA environment.
- All incidents identified during the testing are documented by QA and the priority and severity is assigned based upon the previously defined guidelines.
- The Project Manager is responsible for the timely resolution of all defects.
- Defect resolution does not impede testing.
- Communication between all groups on the project is paramount to the success of the project; therefore QA should be involved in all relevant project communication.
- Sufficient time is incorporated into the schedule not only for testing, but also for unit testing by developer, test planning, verification of defect fixes, and regression testing by QA

5.8. Dependencies

This section lists all the external and internal dependencies in executing the test cases for the project:

- For testing Admin module we need to test Login module first.
- Requirements module can be tested after the Administrator module
- Requirements module should be ready before testing Test Plan module
- For Test lab and Reports all the above module should be in working state.

5.9. Test Deliverables

Sr. No.	Deliverable Name	Planned End Date	Responsibility	Delivery Mechanism
1.	Software Test Plan	12-Oct-05	QA Lead	Softcopies
2.	Test Cases	13-Feb-05	QA Engineer	Softcopies
3.	Build Acceptance Report	13-Apr-06	QA Engineer	Softcopies
4.	Test Summary Report	30-Apr-06	QA Engineer	Softcopies

Table 5.4: **Test Deliverables**

5.10. Software Test Plan Updation

Software Test Plan is updated whenever there is change in Project modules. The STP can be updated in following conditions.

- Slippage in schedule.
- Change in the scope of the work and other commitments.
- Change in Requirements
- Enhancement in implemented feature
- Request from Client

6. Coding and Testing

6. Coding and Testing

6.1. C# Code Sample :

6.1.1. Authentication C# Code

```
using System;
using System.Collections;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Web;
using System.Web.SessionState;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Web.UI.HtmlControls;
using System.Data.SqlClient;
using System.DirectoryServices;

namespace MyQAEye
{
        /// <summary>
        /// Summary description for WebForm1.
            public class Default : System.Web.UI.Page
        {
            public string pwd;
            public int Uid;
            public int Flag;
            public string Uname,UmailId;
            SqlConnection cn;
            SqlCommand com;
            SqlDataAdapter adpt;
            DataSet ds;
            protected System.Web.UI.WebControls.Image Image1;
            protected System.Web.UI.WebControls.Label Label1;
            protected System.Web.UI.WebControls.TextBox txtUser;
            protected System.Web.UI.WebControls.Label Label2;
            protected System.Web.UI.WebControls.TextBox txtPword;
            protected System.Web.UI.WebControls.Button btnSmt;
            protected System.Web.UI.WebControls.Button btnCancel;
            QAeye.Connection con= new QAeye.Connection();

public bool IsAuthenticated(string domain, string username, string pwd)
            {
                //connecting 2 dierctory store
                string _path = "LDAP://DC=cybage,DC=com ";
                string domainAndUsername = domain + @"\" + username;
                try
                {
```

```
                    //search in directory store
            DirectorySearcher search = new DirectorySearcher(entry);
            search.Filter = "(SAMAccountName=" + username + ")";
            SearchResult result = search.FindOne();

                  if(null==result)
                        {
                              //entry.Close();
                              return false;
                        }
      ResultPropertyValueCollection resVal=result.Properties["cn"];
      ResultPropertyValueCollection res=result.Properties["mail"];
      Uname=resVal[0].ToString();
      UmailId=res[0].ToString();

      return true;

                  }
                  catch(Exception ex)
                  {
Response.Write("LDAP Authentication Failed"+ex.Message.ToString());
      return false;
                  }
            }
      public bool IsLdap(string UserName,string UPassword)
            {
                  //CHECK LDAP AUTHENTICATION
                  cn=con.doconnect();
                  com=new  SqlCommand("select  Login_Name,Password  from
                  users",cn);
                  ds=new DataSet();
                  adpt=new SqlDataAdapter(com);
                  adpt.Fill(ds);
                  int LDAP=1;
                  foreach (DataRow Row in ds.Tables[0].Rows)
                  {
                        if(Row["Login_name"].ToString()==UserName)
                        {
                  LDAP=Int32.Parse(Row["LDAP"].ToString());
                  pwd=Row["Password"].ToString();
                  Uid=Int32.Parse(Row["User_Id"].ToString());
                  Flag=1;
                  break;
                        }
                  }
                  //con.disconnect();
                  //LDAP Authentication
if(Flag==1 && LDAP==1)
```

```
return IsAuthenticated("cybage",txtUser.Text,txtPword.Text);
                    //LOCAL Authentication
if(Flag==1 && LDAP==0)
return (pwd==UPassword?true:false);
                    //LDAP authentication & ADD 2 Local Database
              if (Flag==0)
                    {

        if(IsAuthenticated("cybage",UserName,UPassword))
                          {
                                //User Addition in Users Table
        com.CommandText="select max(User_Id) from Users";
        int User_Id=Int32.Parse(com.ExecuteScalar().ToString());
        com=new            SqlCommand("insert            into            Users
        values("+(User_Id+1)+",'"+Uname+"','"+UserName+"','cybage','"+Uma
        ilId+"',"+1+","+8+")",cn);
        com.ExecuteNonQuery();
        //Created User Included in "EveryOne" Group
        com.CommandText="insert                into                User_Groups
        values("+100+","+(User_Id+1)+",'This User is in Default(EveryOne)
        Group')";
                          com.ExecuteNonQuery();
                          return true;
                          }
                    }
                    con.disconnect();
                    return false;

              }

        private void SetFocus(Control ctrl)
              {
                    // Define the JavaScript function for the specified
                    control.
                    string focusScript = "<script language='javascript'>"
                    + "document.getElementById('" + ctrl.ClientID +
                    "').focus();</script>";
                    // Add the JavaScript code to the page.
                    Page.RegisterStartupScript("FocusScript",ocusScript);
              }
        private void Page_Load(object sender, System.EventArgs e)
              {
              Session["S_prjID"]=-1;
              SetFocus(txtUser);
              }

              // #region Web Form Designer generated code
```

```
    private void btnSmt_Click(object sender, System.EventArgs e)
          {
                try
                {

    if(IsLdap(txtUser.Text,txtPword.Text))//Login Sucess
                      {
                            Session["S_tsID"]=-1;
                            Session["S_bpID"]=-1;
                            Session["S_prjID"]=-1;
                            Session["S_userID"]=Uid;
                            //Get GroupID
    com.CommandText="Select    Group_Id    from    User_Groups    where
    User_Id="+Uid;
    Session["S_grpID"]=Int32.Parse(com.ExecuteScalar().ToString());
    con.disconnect();
    //btnSmt.Text=Session["S_grpID"].ToString();
          if (Int32.Parse(Session["S_grpID"].ToString())==101)
                            Response.Redirect("ManageClients.aspx");
                            else
                            Response.Redirect
                            ("manageProjects.aspx");
                            }
                      else // Login Fail
                            {
                Response.Write("Authentication     Failed     -InValid
                UserName/Password");//ERROR MSG
                            }
                }
                catch(Exception ex)
                {
    con.disconnect();
    Response.Write("Authentication Failed"+ex.Message.ToString());
                }
          }
private void txtUser_TextChanged(object sender, System.EventArgs e)
          {

          }

    }
}
```

6.2. Automation testing

For automation testing of this project QA Wizard tool is used. QA Wizard is a test automation tool designed for testing Web, Windows, and Java-based applications. Its architecture enables you to quickly develop simple test scripts by navigating through an application. QA Wizard's advanced features, such as checkpoints and variables, also help you easily develop more complex scripts.
QA Wizard eliminates writing scripts manually in a 3GL such as VB Script or JavaScript. Debugging time is virtually eliminated since you do not have to remember how to instantiate a variable or know whether a specific application object exists. QA Wizard's scripting language provides all of the features of a modern structured language, including flow control, subroutines, constants, conditionals, variables, assignment statements, and functions.
QA Wizard supports Web, Windows, and Java applications. Web script recording is supported in Microsoft Internet Explorer. Web script playback is supported in Internet Explorer and Mozilla Firefox. QA Wizard supports Microsoft Visual SourceSafe and Seapine Surround SCM. SCC integration lets you perform most SCC tasks from QA Wizard. After adding scripts to source control, you will be able to easily share script changes and maintain a record of changes made to scripts.

6.2.1. Add Client Automation scripts

Step #	Action Type	Object Type	Window	Text	Alias
0	Main	main			
1	Open Browser		Window0		
2	Navigate		Window0	http://192.168.14.114/Clients.aspx	
3	Type Text	Password	Web Browser	******	
4	Key Press [Enter]	Password	Web Browser	******	
5	Navigate		Window0	http://192.168.14.114/client.aspx	
6	Mouse DblClick	Submit Image	Window0		imgNewClient
7	Input	Text Field	Window0		txtClientName
8	If Check Attributes	Text Field	Window0		
9	Fail	Continue			
10	Key Press [Tab]	Text Field	Window0		txtClientName
11	Input	Text Field	Window0		txtAcctCode
12	If Check Attributes	Text Field	Window0		
13	Fail	Continue			
14	Key Press [Tab]	Text Field	Window0		txtAcctCode
15	Input	Text Field	Window0		txtSeats
16	Check Attributes	Text Field	Window0		txtSeats
17	Fail	Continue			
18	Mouse Click	Submit Image	Window0		imgSave

6.2.2. Add Project Script

Step #	Action Type	Object Type	Window	Text	Alias
0	Main	main			
1	Navigate		Window0	http://192.168.14.114/Project.aspx?CL	
2	Mouse DblClick	Submit Image	Window0		imgNewProje
3	Select	List Box (DropDown ...	Window0	-- All Clients -- @#$ 11 121 2 3 54 aaaa	cboClientNan
4	Navigate		Window0	http://192.168.14.114/Project.aspx?CL	
5	Mouse Click	Text Field	Window0		txtProjectNan
6	Input	Text Field	Window0		txtProjectNan
7	Key Press [Tab]	Text Field	Window0		txtProjectNan
8	Input	TextArea Control	Window0		txtDescriptior
9	Key Press [Tab]	TextArea Control	Window0		txtDescriptior
10	Input	Text Field	Window0		txtConceptCc
11	Key Press [Tab]	Text Field	Window0		txtConceptCc
12	Mouse Click	Image	Window0		imgField_Star
13	Mouse Move	Table's Cell	Window0	17	calCell
14	Mouse Move	Table's Cell	Window0	10	calCell
15	Mouse Move	Table's Cell	Window0	11	calCell
16	Mouse Move	Table's Cell	Window0	18	calCell
17	Mouse Move	Table's Cell	Window0	17	calCell
18	Mouse Move	Font	Window0	18	cellText
19	Mouse Move	Table's Cell	Window0	18	calCell
20	Mouse Move	Table's Cell	Window0	17	calCell
21	Mouse Move	Font	Window0	17	cellText
22	Mouse Click	Font	Window0	17	cellText
23	Mouse Click	Button	Web Browse	OK	
24	Mouse Click	Image	Window0		imgField_Star
25	Mouse Move	Table's Cell	Window0	12	calCell
26	Mouse Move	Table's Cell	Window0	19	calCell
27	Mouse Move	Font	Window0	18	cellText
28	Mouse Click	Font	Window0	18	cellText
29	Mouse Click	Image	Window0		imgField_End
30	Mouse Move	Table's Cell	Window0	11	calCell
31	Mouse Move	Table's Cell	Window0	19	calCell
32	Mouse Click	Table's Cell	Window0	19	calCell
33	Mouse Click	Image	Window0		imgExpiry_Da
34	Mouse Move	Table's Cell	Window0	18	calCell
35	Mouse Move	Font	Window0	26	cellText
36	Mouse Move	Table's Cell	Window0	27	calCell

		A	B	C	D
	Iteration	Select List Box (DropDo Selected Text cboClientName	Input Text Field Value txtProjectName	Input TextArea Control Value txtDescription	Input Text Field Value txtConceptCount
1	main	Amol_cc	TEST AMOLCC	This is for tersting	12

6.2.3. Add User script

Step #	Action Type	Object Type	Window	Text	Alias
0	Main	main			
1	Navigate		Window0	http://192.168.14.114/user.aspx?CLNO	
2	Mouse DblClick	Submit Image	Window0		imgNewUser
3	Select	List Box (DropDown ...	Window0	-- All Clients -- @#$ 11 121 2 3 54 aaaa	cboClientNan
4	Navigate		Window0	http://192.168.14.114/user.aspx?CLNO	
5	Mouse Click	Text Field	Window0		txtName
6	Input	Text Field	Window0		txtName
7	Key Press [Tab]	Text Field	Window0		txtName
8	Input	Text Field	Window0		txtFullName
9	Key Press [Tab]	Text Field	Window0		txtFullName
10	Input	Password Field	Window0		txtPassword
11	Key Press [Tab]	Password Field	Window0		txtPassword
12	Input	Password Field	Window0		txtCfmPasswc
13	Mouse Click	Submit Image	Window0		imgSave
14	Navigate		Window0	http://192.168.14.114/user.aspx?CLNO	
15	Mouse Click	Submit Image	Window0		imgCancel
16	Navigate		Window0	http://192.168.14.114/Users.aspx?CLN	
17	Open Browser		Window1		
18	Close Browser		Window1		
19	Open Browser		Window1		
20	Navigate		Window1	http://carlosag.net/Tools/CodeTranslat	
21	Mouse Click	File Field	Window1		ctl00$Conten
22	Mouse Click	Combo Box.Button	Web Browse	Desktop	
23	Select	List Box Item	Web Browse	CEON (F:)	
24	Mouse DblClick	Item	Web Browse	QA_Wizard	
25	Mouse Up	Items	Web Browse		
26	Mouse Click	Tool Bar Button	Web Browse		
27	Mouse DblClick	Item	Web Browse	ASP.net	
28	Mouse Up	Item	Web Browse	ComponentArt_WebUI_2006_1_f...	
29	Mouse DblClick	Item	Web Browse	AdminCS	
30	Mouse Up	Items	Web Browse		
31	Mouse DblClick	Item	Web Browse	Includes	
32	Mouse Up	Item	Web Browse	UserControls	
33	Mouse DblClick	Item	Web Browse	UserControls	
34	Mouse Up	Item	Web Browse	GridControl.ascx	
35	Mouse Click	Item	Web Browse	Footer.ascx.vb	
36	Mouse Click	Button	Web Browse	&Open	

	Iteration	A Select List Box (DropDo Selected Text cboClientName	B Input Text Field Value txtName	C Input Text Field Value txtFullName	D Input Password Field Value txtPassword	E Input Password Field Value txtCfmPassword	F Select List Box Item Text
1	main	ABCD	Amolc21	Amol Chaudhari	****	****	CEON (F:)
2	main	ABCD	Amolc21	Amol Chaudhari	****	****	CEON (F:)

6.2.4. Result of Client Script

mary

Workspace	Project	Script	Time			Number of steps	Passed	Failed	Warnings	Application	Mode	Send summary
			Start	End	Total							
Admin	Administrator	Clients	11:13:34	11:14:39	00:01:05	18	16	2	0		User	Summary

pt Summary

Workspace	Project	Script	Execute Mode	Time			Status
				Start	End	Total	
Admin	Administrator	Clients	User	11:13:34	11:14:39	00:01:05	16 passed 2 failed 0 warnings

ation report

Status	Iteration No	Failed steps	Time		
			Start	End	Total
	1	2	11:13:34	11:14:39	00:01:05

6.2.5. Automated Status

Pretail steps

	Iteration / Step	Action Type	Object Type	Variable
	1 / 0	Loop	main	
	1 / 1	Open Browser - Internet Explorer		
	1 / 2	Navigate	http://192.168.14.114/Clients.aspx	
	1 / 3	Type Text	Password	A1=**********
	1 / 4	Key Press [Enter]	Password	
	1 / 5	Navigate	http://192.168.14.114/client.aspx	
	1 / 6	Mouse DblClick	Submit Image	
	1 / 7	Input	Text Field Image: imagen	B1=TEST_DATA
	1 / 8	If Check Attributes	Text Field	
	1 / 9	Fail	Continue	
	1 / 10	Key Press [Tab]	Text Field	
	1 / 11	Input	Text Field	C1=21
	1 / 12	If Check Attributes	Text Field	
	1 / 14	Key Press [Tab]	Text Field	
	1 / 15	Input	Text Field	D1=40
	1 / 16	Check Attributes	Text Field	
	1 / 17	Fail	Continue	

6.2.6. Defect Classification

The following are the **defect priorities** defined according to their precedence:

Sr. No.	Defect Priority	Description
1.	P1	Further development and / or testing cannot occur until the defect has been repaired. The system cannot be used or not advisable to use until the repair has been affected. The defect must be resolved as soon as possible because it is impairing development/and or testing activities OR it is violating an important business constraint (Implementation contrary to requirements). System use will be severely affected/or will be insecure to the user until the defect is fixed.
2.	P2	The defect should be resolved in the normal course of development activities. It can wait until a new build or version is created. The defect does not result in a failure, but causes the system to produce incorrect, incomplete, or consistent results, or the defect impairs the systems usability.
3.	P3	The defect is an irritant which should be repaired but which can be repaired after more serious defects have been fixed. The defect does not cause a failure, does not impair usability, and the desired processing results are easily obtained by working around the defect.

Table 6.1: **Defects Classification**

The following are the **defect severity** defined according to their precedence:

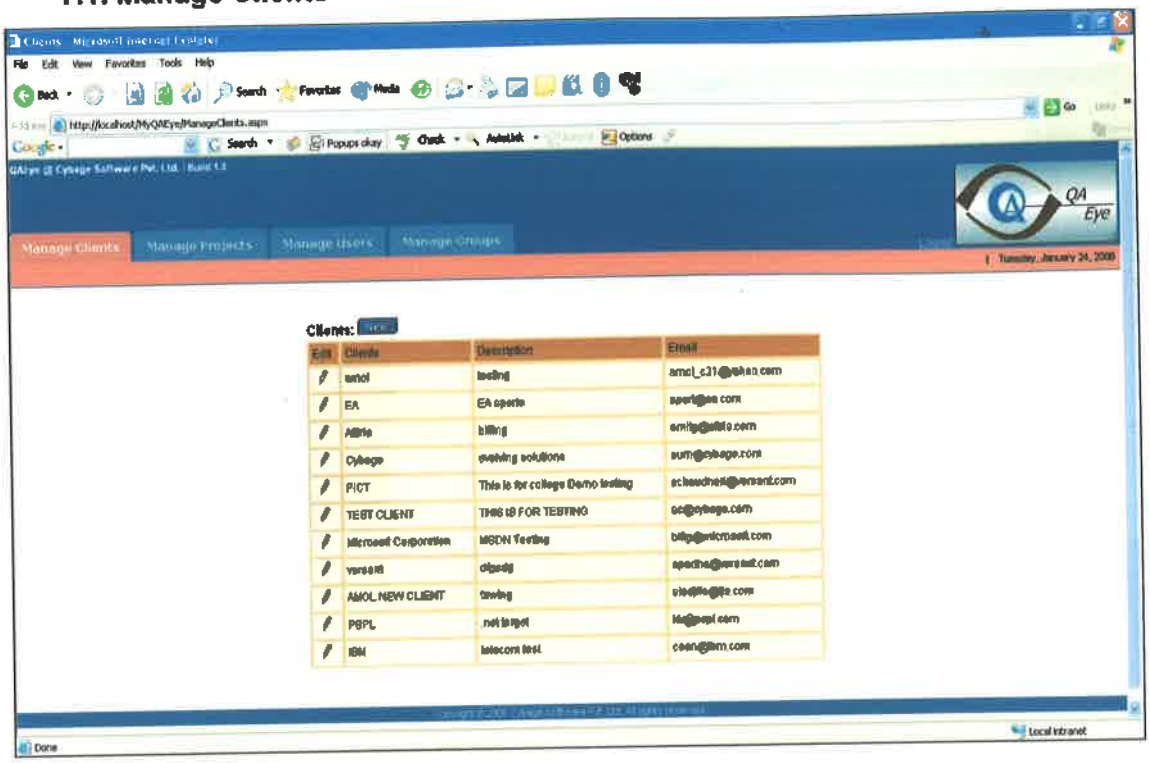
Sr. No.	Defect Severity	Description
1.	Causes Crash	Defect leads to crashing of system.
2.	Critical	System cannot work if the defect is not fixed.
3.	Major	Defect has a large impact on the working of the system.
4.	Minor	Defect does not largely affect the working of the system.
5.	Enhancement	These are the features requested for improving the current system.

Table 6.2: **Defects severity**

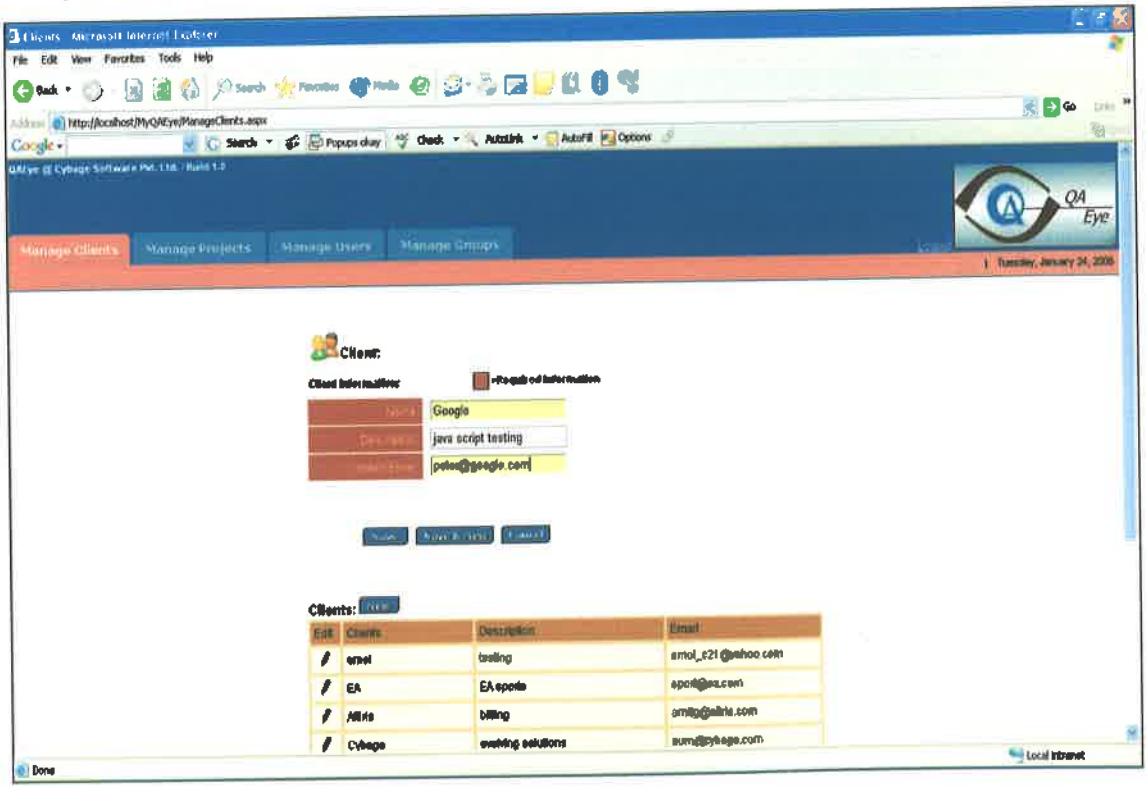
7. Snapshots

7. Snapshots

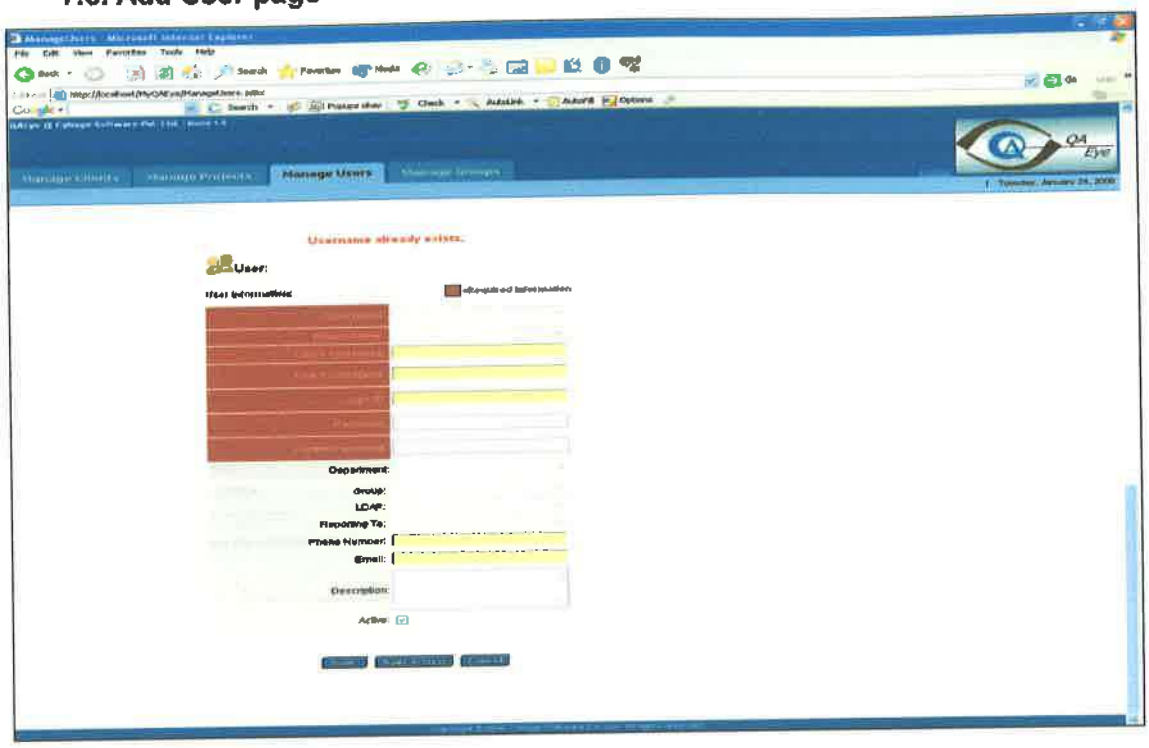
7.1. Manage Clients



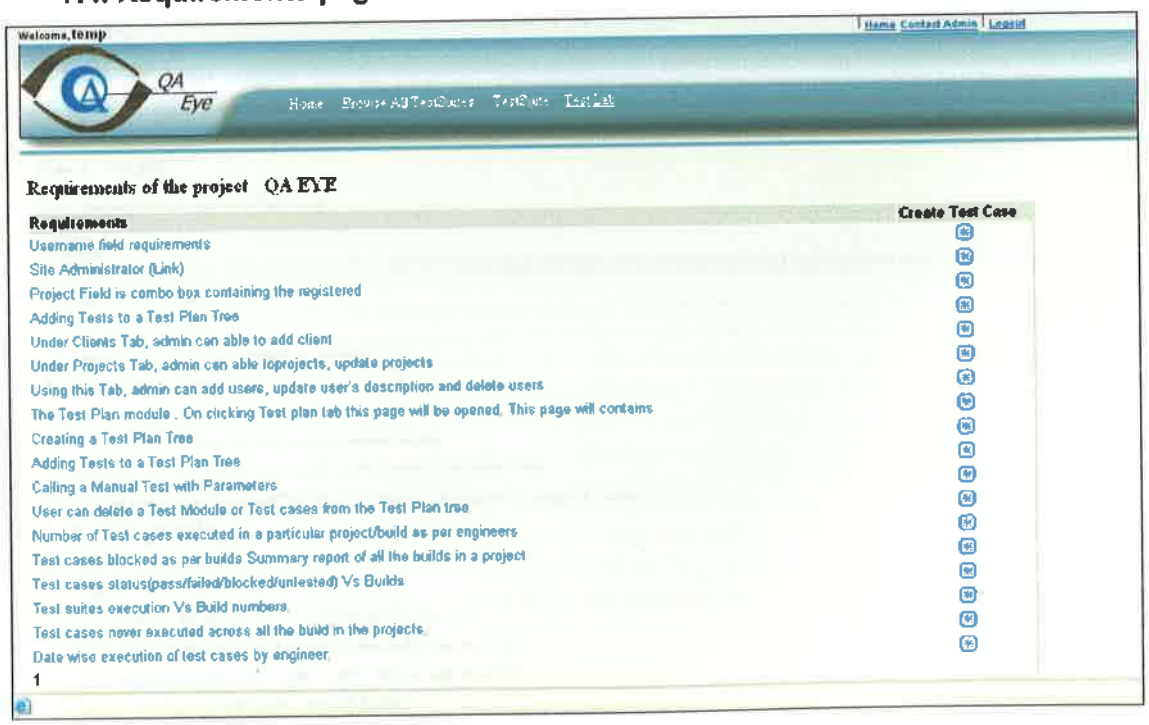
7.2. New Clients



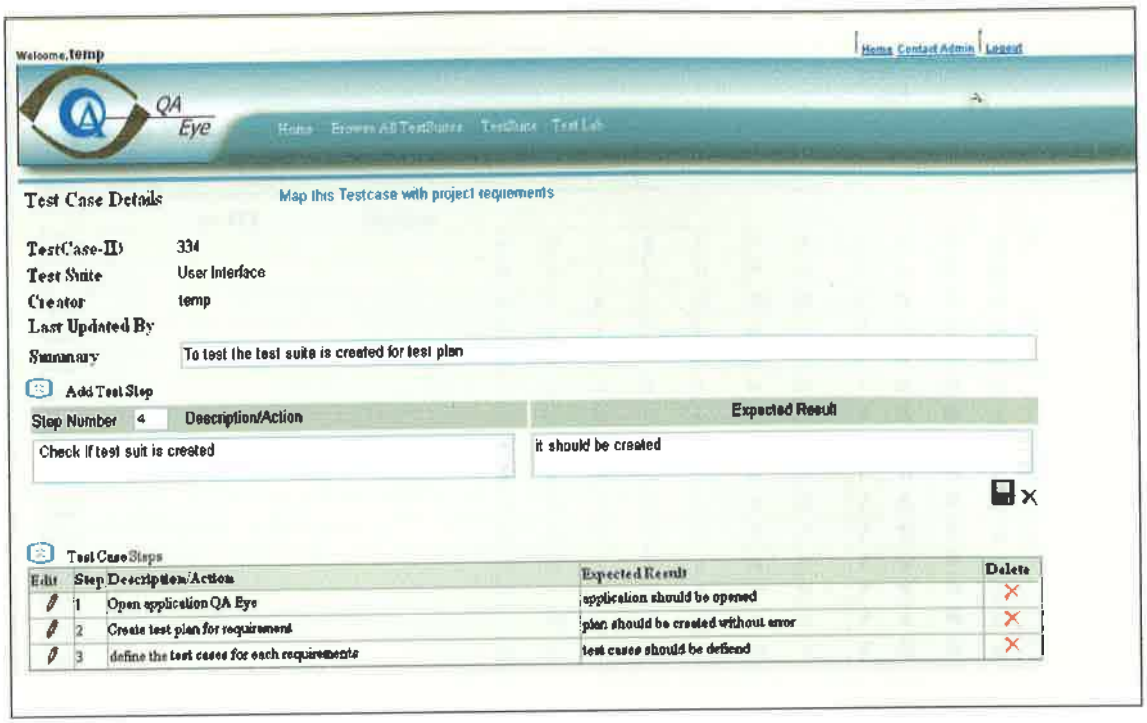
7.3. Add User page



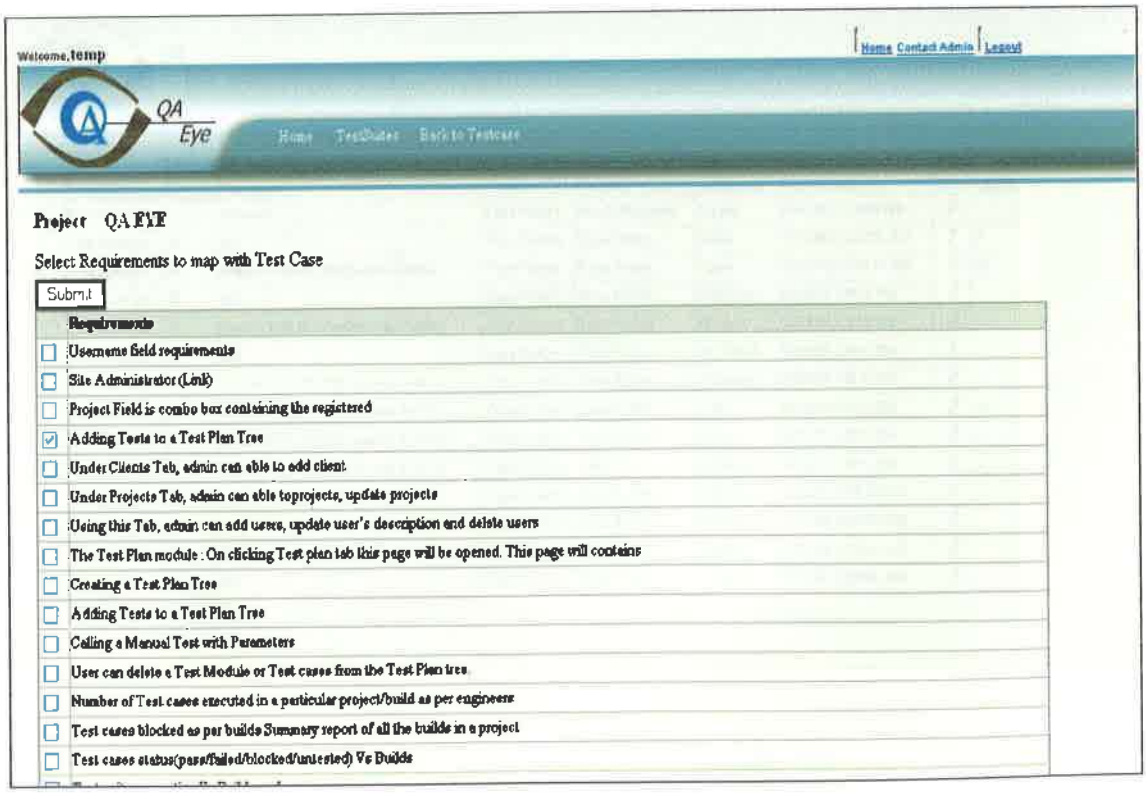
7.4. Requirements page



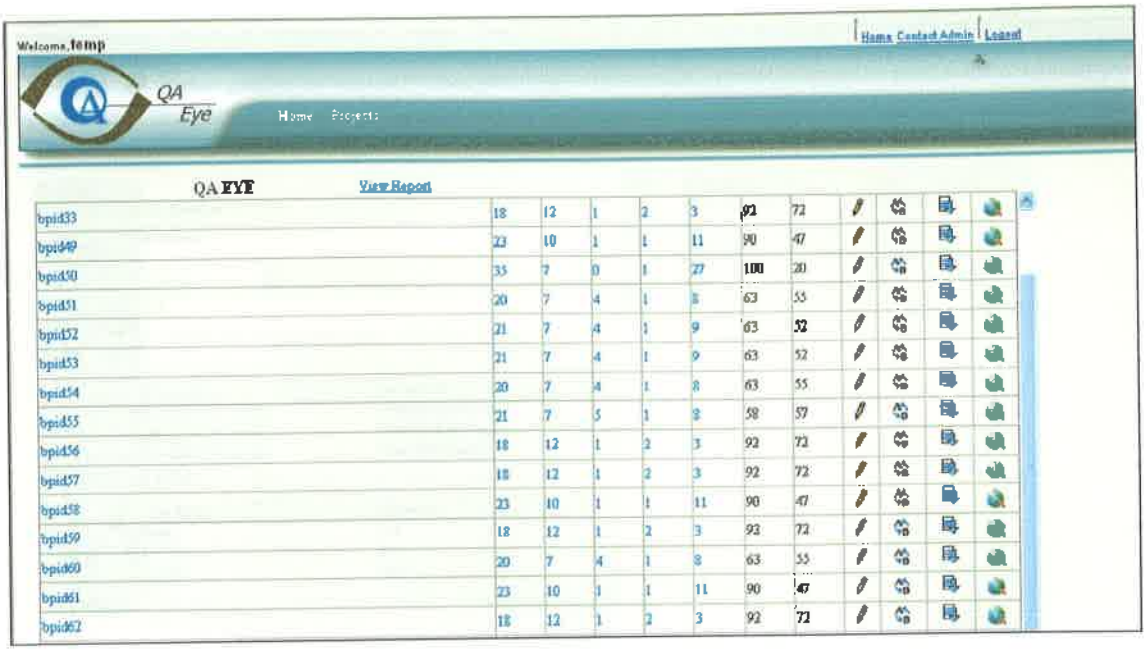
7.5. Add test case



7.6. Map test case to requirement

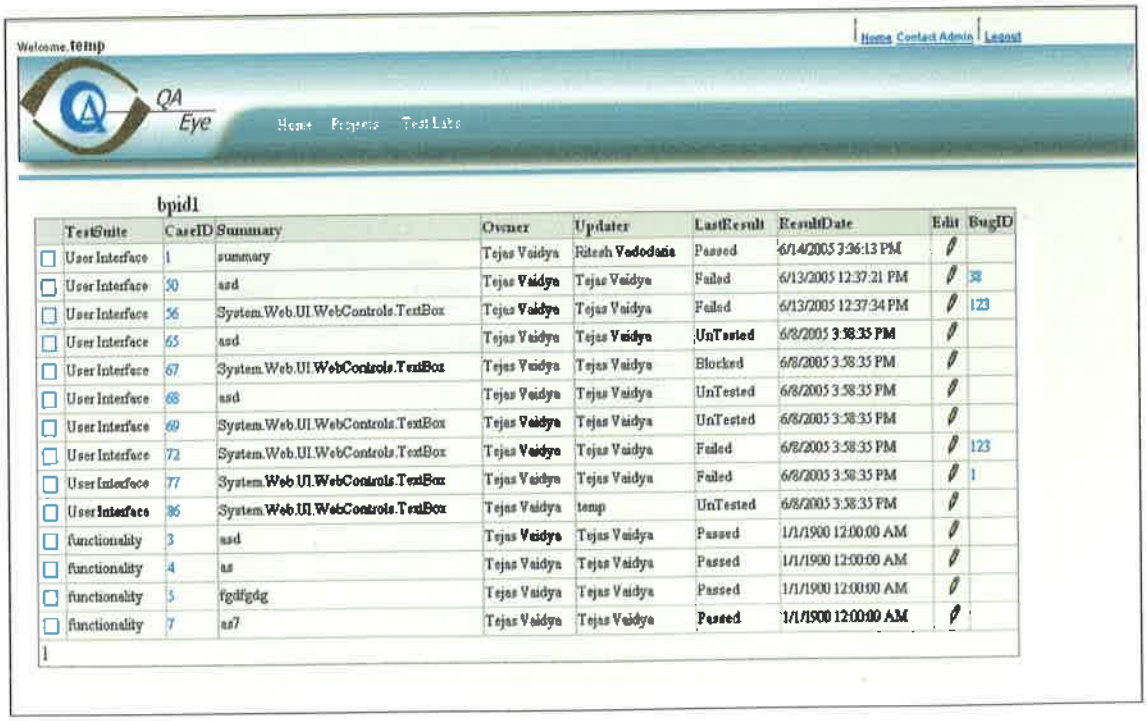


7.7. Browse Build plans



bpid33	18	12	1	2	3	92	72				
bpid49	23	10	1	1	11	90	47				
bpid50	35	7	0	1	27	100	20				
bpid51	20	7	4	1	8	63	55				
bpid52	21	7	4	1	9	63	52				
bpid53	21	7	4	1	9	63	52				
bpid54	20	7	4	1	8	63	55				
bpid55	21	7	5	1	8	58	57				
bpid56	18	12	1	2	3	92	72				
bpid57	18	12	1	2	3	92	72				
bpid58	23	10	1	1	11	90	47				
bpid59	18	12	1	2	3	92	72				
bpid60	20	7	4	1	8	63	55				
bpid61	23	10	1	1	11	90	47				
bpid62	18	12	1	2	3	92	72				

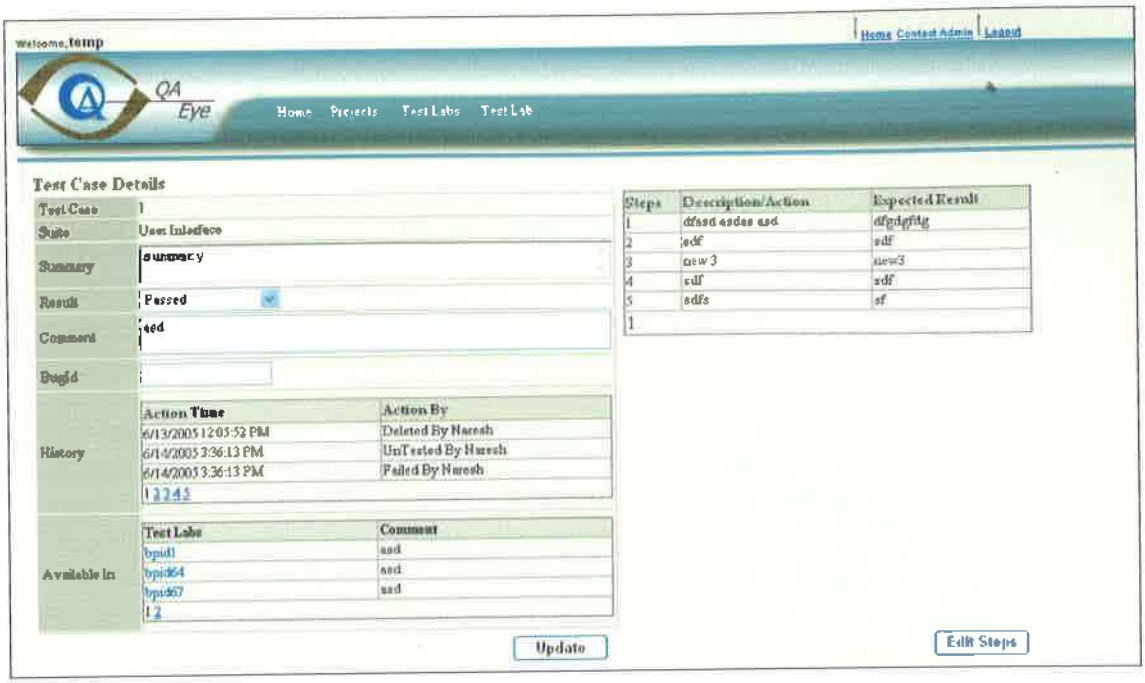
7.8. Browse test plan



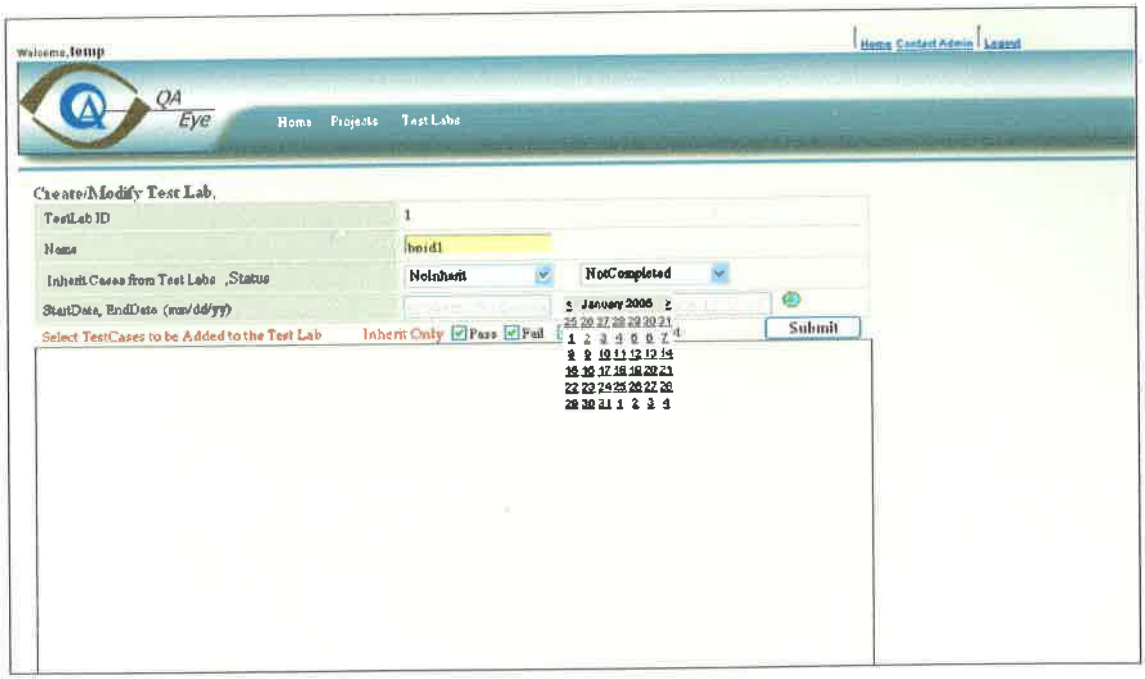
	TestSuite	CaseID	Summary	Owner	Updater	LastResult	ResultDate	Edit	BugID
☐	User Interface	1	summary	Tejas Vaidya	Ritesh Vadodaria	Passed	6/14/2005 3:36:13 PM		
☐	User Interface	50	asd	Tejas Vaidya	Tejas Vaidya	Failed	6/13/2005 12:37:21 PM		38
☐	User Interface	56	System.Web.UI.WebControls.TextBox	Tejas Vaidya	Tejas Vaidya	Failed	6/13/2005 12:37:34 PM		123
☐	User Interface	65	asd	Tejas Vaidya	Tejas Vaidya	UnTested	6/8/2005 3:58:35 PM		
☐	User Interface	67	System.Web.UI.WebControls.TextBox	Tejas Vaidya	Tejas Vaidya	Blocked	6/8/2005 3:58:35 PM		
☐	User Interface	68	asd	Tejas Vaidya	Tejas Vaidya	UnTested	6/8/2005 3:58:35 PM		
☐	User Interface	69	System.Web.UI.WebControls.TextBox	Tejas Vaidya	Tejas Vaidya	UnTested	6/8/2005 3:58:35 PM		
☐	User Interface	72	System.Web.UI.WebControls.TextBox	Tejas Vaidya	Tejas Vaidya	Failed	6/8/2005 3:58:35 PM		123
☐	User Interface	77	System.Web.UI.WebControls.TextBox	Tejas Vaidya	Tejas Vaidya	Failed	6/8/2005 3:58:35 PM		1
☐	User Interface	86	System.Web.UI.WebControls.TextBox	Tejas Vaidya	temp	UnTested	6/8/2005 3:58:35 PM		
☐	functionality	3	asd	Tejas Vaidya	Tejas Vaidya	Passed	1/1/1900 12:00:00 AM		
☐	functionality	4	as	Tejas Vaidya	Tejas Vaidya	Passed	1/1/1900 12:00:00 AM		
☐	functionality	5	fgdfgdg	Tejas Vaidya	Tejas Vaidya	Passed	1/1/1900 12:00:00 AM		
☐	functionality	7	as7	Tejas Vaidya	Tejas Vaidya	Passed	1/1/1900 12:00:00 AM		

1

7.9. Test results



7.10. Create and Modify test lab



8. Conclusion

8. Conclusion

Application testing is a complex process involving the development and execution of thousands of tests. Often, tests are required for multiple hardware platforms, multiple configurations (computers, operating systems, and browsers) and multiple application versions. Managing all aspects of the testing process can be time-consuming and difficult.

QA Eye helps you maintain a project database of tests that cover all aspects of your application's functionality. Every test in your project is designed to fulfill a specified testing requirement of your application. To meet the various goals of a project, you organize the tests in your project into unique groups. QA Eye provides an intuitive and efficient method for scheduling and executing test sets, collecting test results, and analyzing the data.

QA Eye also features a sophisticated system for tracking application defects, enabling you to monitor defects closely from initial detection until resolution. By linking QA Eye to your e-mail system, defect tracking information can be shared by all application development, quality assurance, customer support, and information systems personnel. QA Eye simplifies and organizes test management by giving you systematic control over the testing process. It helps you create a framework and foundation for your testing workflow.

QA Eye guides you through the requirements specification, test planning and test execution phases of the testing process. By integrating all the tasks involved in application testing, it helps ensure that your customers receive the highest quality applications.

In Short,

- QA Eye simplifies and organizes test management
- QA Eye helps you create a framework and foundation for your testing workflow.
- QA Eye helps you maintain a project database of tests that cover all aspects of your application's functionality.
- QA Eye guides you through the requirements specification, test planning, test execution, and defect tracking phases of the testing process.

➢ Acronyms and Definitions

Sr. No.	Acronyms	Definitions
1.	QA Eye	Quality Assurance Eye
2.	PBN	Project by Net
3.	PM	Project Manager
4.	QA Engineer	Quality Assurance Engineer
5.	SOW	Statement of Work
6.	SRS	Software Requirement Specification
7.	PRD	Product Requirement Document
8.	HLD	High level Design
9.	LLD	Low level Design
10.	CCB	Change Control Board
11.	QL	Quality Lead
12.	CC	Configuration Controller
13.	SQA	Software Quality Assuror
14.	STP	Software Test Plan
15.	SPMP-D	Software Project Management Plan - Development
16.	LDAP	Lightweight directory access protocol.
17.	BAR	Build Acceptance Report
18.	TSR	Test Summary Report
19.	WSR	Weekly Status Report
20.	TTWeb	Test Track Web
21.	DD	Design Document
22.	CAR	Corrective Analysis and Resolution Process
23.	MOM	Minutes of Meeting

➢ References

- Alexander Ian F., and Richard Stevens, 2002." **Writing Better Requirements**". London: Addison-Wesley.
- Collard, Ross. 1999. "**Test Design**." Software Testing & Quality Engineering 1(4):30 37.
- "**Applied Microsoft .NET Framework Programming**", by Jeffrey Richter
- "**Programming C# ** ", 2nd Edition, Jesse Liberty, Publisher: O'Reilly, Second Edition February 2002
- "**ASP.NET Web Developer's Guide**", by Mesbah Ahmed, Chris Garrett, Jeremy Faircloth
- "**Advance Database Management Systems**" by Korth
- Gomaa, H., 1995, **"Design Methods for Domain Specific Software Architectures,"** Proceedings of the First International Workshop on Architectures for Software Systems,April. pp. 101-108.

➢ Online References

Mercury Help for Requirements	http://www.mercury.com/TestDirector
MSDN .Net Start Page	http://msdn.microsoft.com/net/
Visual Studio .Net	http://msdn.microsoft.com/net/
ASP.NET	http://www.asp.net/ http://www.aspng.com/
C# Corner	http://www.c-sharpcorner.com/ http://www.c-sharpcorner.com/forum/ http://www.csharptoday.com
SQL Server	http://www.syngress.com