2011

# Enterprise management studio

Himabindu Miriyala

# ENTERPRISE MANAGEMENT STUDIO

---

A Project

Presented to the

Faculty of

California State University,

San Bernardino

---

In Partial Fulfillment

of the Requirements for the Degree

Master of Science

in

Computer Science

---

by

Himabindu Miriyala

June 2011

ENTERPRISE MANAGEMENT STUDIO

---

A Project

Presented to the

Faculty of

California State University,

San Bernardino

---

by

Himabindu Miriyala

June 2011

Approved by:

[redacted]

Dr. Ernesto Gomez, Advisor, School of Computer Science and Engineering

24 MAY 2011
Date

[redacted]

Dr. David Turner

[redacted]

Dr. Kerstin Voigt

ABSTRACT

Enterprise Management Studio (EMS) is a web-based system that can be used to keep track of all project details like project names, descriptions, managers, estimated project start dates, completion dates and durations, and resources assigned to projects. The system provides a user-friendly graphic interface to manage project information and user accounts. This system also facilitates tracking the issues of projects, provides the capability of searching for projects, and generates various kinds of reports. EMS also allows different project members who are located in different places to access and modify the project information. In addition to these, EMS also provides security to confidential information by having it viewed by only those who have been given access. EMS is written in C Sharp and is built on the .NET framework, which helped to reduce the lines of code needed to implement the desired functionality. The Asp.net framework and Ajax techniques provided by the .NET framework were used to implement web pages, and Ado.net framework was used to allow the web pages to communicate with the database.

ACKNOWLEDGMENTS

I thank the faculty of Computer Science and Engineering Department for giving me an opportunity to pursue my M.S. in Computer Science and at California State University, San Bernardino. I express my sincere gratitude to my advisor, Dr. Ernesto Gomez, and my other committee members Dr. David Turner and Dr. Kerstin Voigt for supporting me and guiding me through this project. Finally, I would like to specially thank my husband who encouraged me and supported me to pursue my masters during these past few years.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

# CHAPTER ONE

# INTRODUCTION

In a software development company, there may be multiple projects that are in development simultaneously. It is often difficult to track the current status of all projects, including project errors, people working on any given project, and other project details. Project managers would have to communicate with each person working on individual projects to know all these details. The purpose of this project is to simplify the communication between all levels of personnel by constructing a software system called Enterprise Management Studio (EMS).

Enterprise Management Studio is a web-based application designed to manage all the projects of a software company within a single centralized system. This application is a system that can be used to keep track of all project details such as project names, descriptions, managers, estimated project start dates, completion dates and durations, and resources assigned to projects. This system also facilitates tracking the issues of projects, provides security levels to the project, manages user accounts, uploads attachments related to projects, and publishes progress of the projects.

Enterprise Management Studio permits different members of a team who are located in different places, to access and modify the project information. It also provides the capability of searching for projects, filtering the data, and searching for users based on usernames or email IDs. This application can also be used by project teams to keep track of bugs, enhancements, and project timelines that can be easily accessible online to managers, developers, and users. With such timesaving organizational tools, EMS can be a useful collaborative tool and helpful not only to a sole-proprietary business but also those with several employees. EMS helps in reaching the business goals faster, as less work time is wasted, and it optimizes decision-making.

## Background

For some companies, the process of project management is manual and relies upon the storage of paper records. Using this method, recording, and retrieval of project details is tedious and time-consuming. Also, when using manual methods, the selection of developers available for project work is done by managers who approach developers in person and inquire as to their availability to work on any given project. In situations of mismanagement,

development project work may become delayed and possibly extend completion times for projects. Additionally, there may be a lack of availability of accurate and proper information relevant to different levels of employees within a firm.

The paper-based process is very time-consuming and reviews may not be done productively. The error corrective processes may not be done in a timely manner, and reports cannot be generated in an effective and meaningful way. Moreover, the administrative cost is extremely high.

Although there are some web-based project management systems such as "REMOTE: A Complete Software Process Management Tool" [5], "Online Project Management System" [6], that have been implemented by CSUSB students in the past, I found that those systems lacked certain features; I chose to incorporate those features in EMS. EMS includes the following additional features which are lacking in the above mentioned web-based project management systems:

1. EMS provides the users the ability to add issues that are related to each other in a project. An issue could be either a bug or a task in a project.
2. EMS allows the users to maintain history of information related to issues such as date

modified, username, item changed, previous value and new value of the item.

3. Whenever the issue is updated or modified, EMS users will be sent these issue notifications through emails.
4. The users will be able to provide the issue description or comments through WYSIWYG editor. The editor is used to do some powerful editing functions that can be done by MS Word application.

## Purpose

It can be challenging to manage projects if the business is set up at home and is run by working with a group of people all based in different locations. EMS can provide a simple solution even if the company work is outsourced or the work is done by a group of home-based employees. The purpose of the project is to develop a web application that will publish information about all projects of a company under development in one single place. The system can be used by a wide variety of groups who do projects. The system created has been specifically tailored for a medium-level company that has employees in different locations and levels. In addition, the system

has been tailored for companies that do projects for their clients. This system is web-based and, hence, will be platform and location independent. Efforts were made to make a system that is unique in functionality and more efficient than the existing ones. This web-based system should provide help in managing different projects as well as the employees associated with the project tasks and it should provide instant status of involvements of an employee in the task.

The system should be able to allocate all the available members to be part of the team and also provide an easy selection procedure to make them available when required. Additionally, the system could provide security to the clients by protecting their confidential information. It should reduce the redundancy of making manual activity logs, which will speed up the processing within the firm.

## Project Scope

EMS users can view information instantly about the projects under process within the company. The scope of this project is to enable the users of an organization to view the issues through the Internet. The companies that use it will be able to keep better track of their project

progress, upload/download files related to the project, and generate various kinds of reports. The users can view the progress of the projects in real time and get the latest information every time they access it. The system will provide instant information regarding the work in progress within the company. In addition to this, the system has an interactive interface through which a user can easily interact with different areas of the application. The application will be made as simple as surfing a website; thereby non-technical persons can also use the application easily.

## Project Products

This project led to the following products:

- Implementation of EMS: a working website with ASP.NET pages as the front end and SQL RDBMS as the back end.
- Systems Manual: a project report (this report) with design details and user specifications.

## Definitions, Acronyms, and Abbreviations

The definitions, acronyms, and abbreviations used in the document are described in this section.

EMS: Enterprise Management Studio

.NET Framework: .NET Framework is the basic framework for the new Microsoft .NET Platform. It is developed and marketed by Microsoft. It provides a common environment for programmers to build, deploy, and run web applications.

ASP: ASP stands for Active Server Pages. ASP is a server-side scripting language used to develop interactive and dynamic websites and web-applications.

ASP.NET: ASP.NET is a technology that is used to develop web-based applications. It is built on Common Language Runtime that allows programmers to write the code in many languages such as C Sharp, Visual Basic, and C++ etc. It is part of Microsoft's .NET platform and is the successor to ASP.

ADO.NET: ADO.NET is a set of computer software components, and is used by the programmers to access data and data services.

C Sharp: C Sharp is one of the programming languages developed by Microsoft and is considered as simple and object-oriented programming language.

SQL: SQL often referred to as Structured Query Language, is a standard language for relational

database management systems. Programmers use "SEQUEL" as an alternative name to call SQL.

Ajax: Ajax stands for Asynchronous JavaScript and XML. It is a group of interrelated web development techniques used on the client-side to create interactive web applications.

CSUSB: California State University, San Bernardino.

HTML: HTML stands for Hyper Text Markup Language, and is used to create web pages.

IIS: Internet Information Services is a web server application that was created by Microsoft for use with Microsoft windows.

CHAPTER TWO

ARCHITECTURE

The system is web based. The front-end pages and business-tier are written in c sharp. The whole project is built on .NET Framework. The database has been created using Microsoft SQL Server 2005. Microsoft Active Data Objects (ADO) has been used to access the SQL Server. A client-based script written in JavaScript has been incorporated for better efficiency and faster client side computations. The whole application was developed using Microsoft Visual Studio 2008 .Net. The system can be hosted on IIS server with .Net on it. Also, HTML/DHTML has been used to display pages and reports. The interaction diagram among various components is shown in Figure 1.

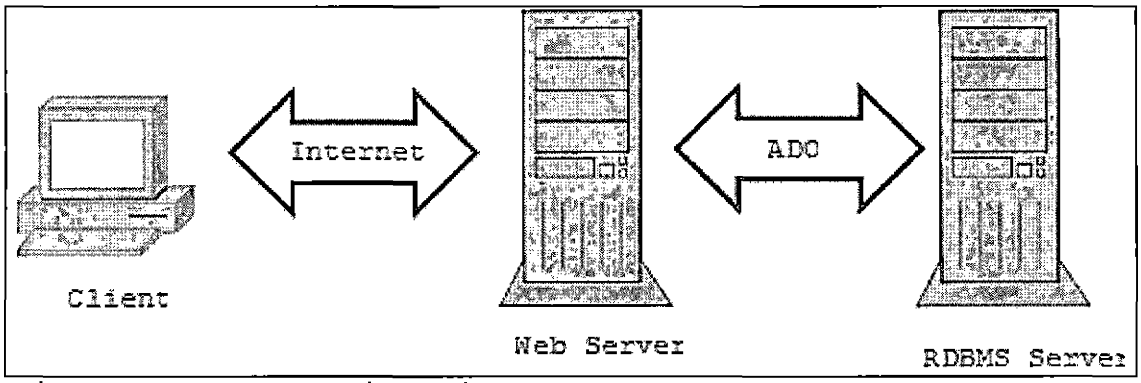


Figure 1. Interaction Diagram among Components

The components chosen for the system were based on i) Platform independence, ii) Efficiency and iii) user friendliness.

The user interface is built using ASP.net web components, HTML, and CSS. JavaScript was used to speed up the efficiency of the system by doing most of the processing on the client side.

ASP.NET has a wide range of libraries available. It offers a variety of options to improve the user response and decreases the time to access the static content in a web page. ASP.NET offers many advanced security features in addition to the simple ones like authentication and authorization. It also has built in compatible components, which can be used to run ADO, thus providing an effective gateway with the database server. Also, it runs consistently on IIS server.

IIS has been used to host the system. IIS was chosen because it comes with all components that are required to process the ASP.Net pages and it works with a variety of browsers and operating systems.

The application should have the ability to track large amounts of critical data and so, the database chosen for the system is SQL server 2005. It supports the integration with Visual Studio development environment as

well as the .NET CLR (common language runtime). The performance of query execution is much faster and can be used to program powerful SQL code with structured exception handling. Also, SQL server is compatible with wide range of database drivers, which can be accessed using ADO and ASP.NET.

### 3-Tier Architecture

The system is organized into three major disjunctive tiers as following:

- Application Layer or Presentation Layer. This is the top level of the application. Application Layer is the form, which provides the user interface to either programmer or the end user.
- Business Layer. This layer is a collection of classes where programmers write the functions to get the data from Application layer and pass the data to the Data Layer.
- Data Access Layer. This Layer is a collection of classes, which is used to get or set the data to the database back and forth. The information is stored into and retrieved from this layer. This layer gets the data from the business layer and

sends it to the database or gets the data from the database and sends it to the business layer.

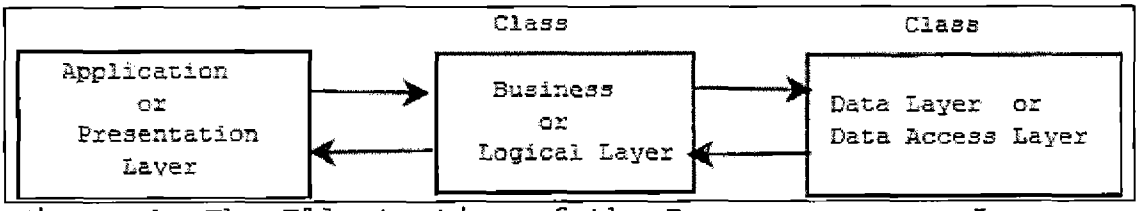


Figure 2. The Illustration of the Processes among Layers in 3-tier Architecture of Enterprise Management Studio

## ADO.NET Architecture

The EMS project uses ADO.NET to provide communication between database and business layer through several key components.

Application: The Application processes and calls ADO.NET functions by submitting the SQL statements and getting the results from the database.

Dataset: The dataset acts as a local copy of the relevant portions of the database, and the data in it can be manipulated and updated independent of the database.

Data Reader: The Data Reader is a component of the data provider. It provides a direct, read-only SQL interface to the database.

Data Adapter: It acts as a mediator facilitating all the communication between the database and the Dataset.

Connection: The connection object is used to create the connection to the database.

Command: The command object is used to execute a command.

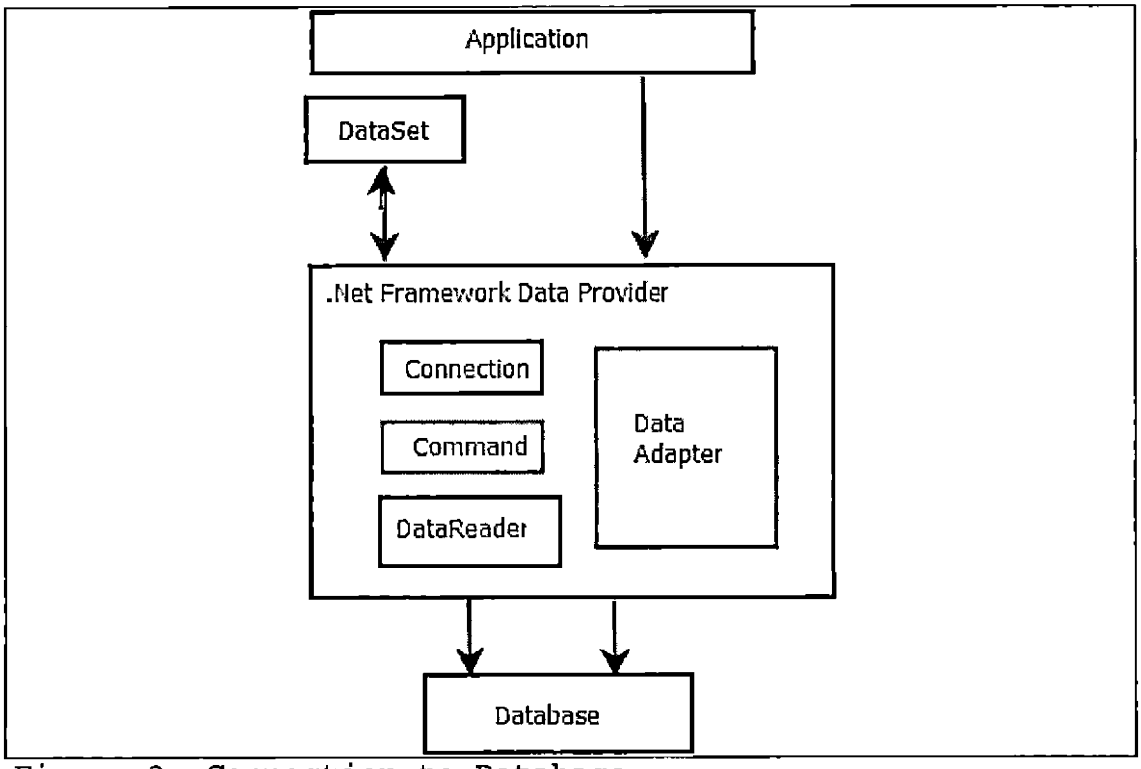


Figure 3. Connection to Database

The namespace System.Data.SQLClient is imported in EMS to access the ADO.NET functionality. The namespace

includes all the functions that are needed to manipulate data from the SQL Server Objects.

## Software Interfaces

The language and applications used to develop the project are ASP.NET, ADO.NET, C sharp, Ajax, and CSS. The compiler used is Visual Studio 2008 compiler. The database used is Microsoft SQL Server 2005 Professional. The Operating System on which the project runs is Windows 2000/XP/Vista. IIS is the web server on which the operating system is running. The Internet Browsers used to view the software interface for the project are Netscape, Internet Explorer, and Mozilla Firefox.

## ASP.NET AJAX Control Toolkit

The ASP.NET AJAX Control Toolkit is an open source project for creating ASP.NET AJAX controls. The toolkit contains a rich set of controls that can be used to build highly interactive and responsive web applications. More than ten controls from the toolkit have been used in EMS.

### Web User Controls

Web User Controls, also called as User Controls, work mostly like an Asp.net Web Page. One can add the server-side web controls and static HTML markup to a user control, and define methods and properties to the control.

The user controls can then be embedded into ASP.NET web pages. Unlike a web page, a user control cannot be requested independently. The control is compiled when the web page containing the control is requested and is stored in memory for subsequent requests. All the web pages pertaining to the project like the introduction, description, members, roles, version and summary have been designed with the web user controls which were later embedded into the respective web pages.

Wizard Control

A wizard can be used to build a module linearly from step 1 to step N. It also allows the user to skip the steps that are unnecessary and lets the user return to previous steps and modify the data in those steps. A wizard control contains one or more Wizard Step controls that represent steps in the wizard. This control has been used in EMS to create a new project. It comprises of seven wizard steps, each of which is loaded in the application during run-time.

Collapsible Panel Extender Control

The Collapsible Panel Extender Control can be used to collapse and extend the panels automatically with a simple mouse-click. It has a target panel which enables the collapse and open behavior. The collapse and extend

behavior for the controls on the panel can be specified by the user. The control helps to achieve smooth animation on the web pages.

```
<ajaxToolkit:CollapsiblePanelExtender ID="cpe" runat="Server"
    TargetControlID="ContentPanel"
    ExpandControlID="Toggle"
    CollapseControlID="Toggle"
    Collapsed="False"
    ExpandDirection="Horizontal"
    ImageControlID="ToggleImage"
    TextLabelID="Toggle"
    ExpandedText="Hide Filter"
    CollapsedText="Show Filter"
    SuppressPostBack="true"
    CollapsedSize="0"
    ExpandedSize="180"
/>
```

Figure 4. Collapsible Panel Extender Control

Update Panel Control

A common application of AJAX is partial page loads. This can be achieved using Update Panel Control. More than one Update Panel Controls can be used per page, and the user can make changes for the entire page by selecting the respective update panel control.

FCK Editor

FCK editor, currently called as CKeditor, is an open source WYSIWYG (what you see is what you get) text editor from CKSource that can be used in web pages. Most of the powerful WYSIWYG editing functions that can be done in

desktop editor can be achieved in the web pages with FCK editor. It is relatively lightweight and does not require any kind of client installation on the client computer. It has been used in setting the welcome message in the Application configuration page and in various other pages.

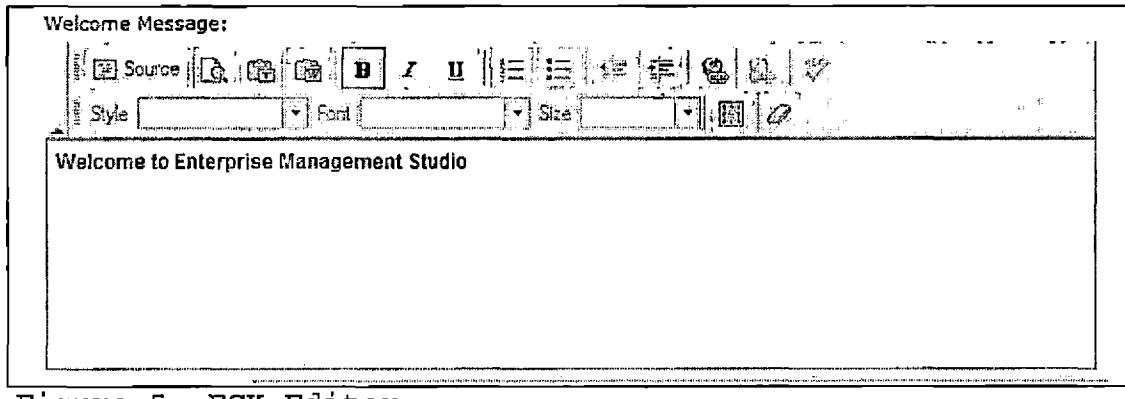


Figure 5. FCK Editor

# CHAPTER THREE

# DATABASE DESIGN

## Data Analysis

EMS implements the database solution through ADO.NET Framework. ADO.NET acts as a bridge between SQL server and ASP.NET application by using Dataset and Data Adapter or Data Reader. This helps to minimize the lines of code to store and retrieve the data. ADO.NET enhances the system performance by maintaining a disconnected database access model. When an application interacts with the database, the connection is opened to serve the request of the application and is closed as soon as the request is completed. Also, when the database is updated, the connection remains open until the operation is completed. Thus, ADO.NET conserves system resources and provides security for databases. The .Net Framework uses System.Data.SqlClient namespace as the Data Provider for the SQL Server.

## Database Specification

In designing the schema for the EMS database, two distinct parts have been identified. One is the conceptual model that includes entities having relationships between them. The second one is the Logical model that includes a

particular domain having structures and constraints of data. The Entity Relational (ER) diagram for the EMS is shown below in Figure 6.

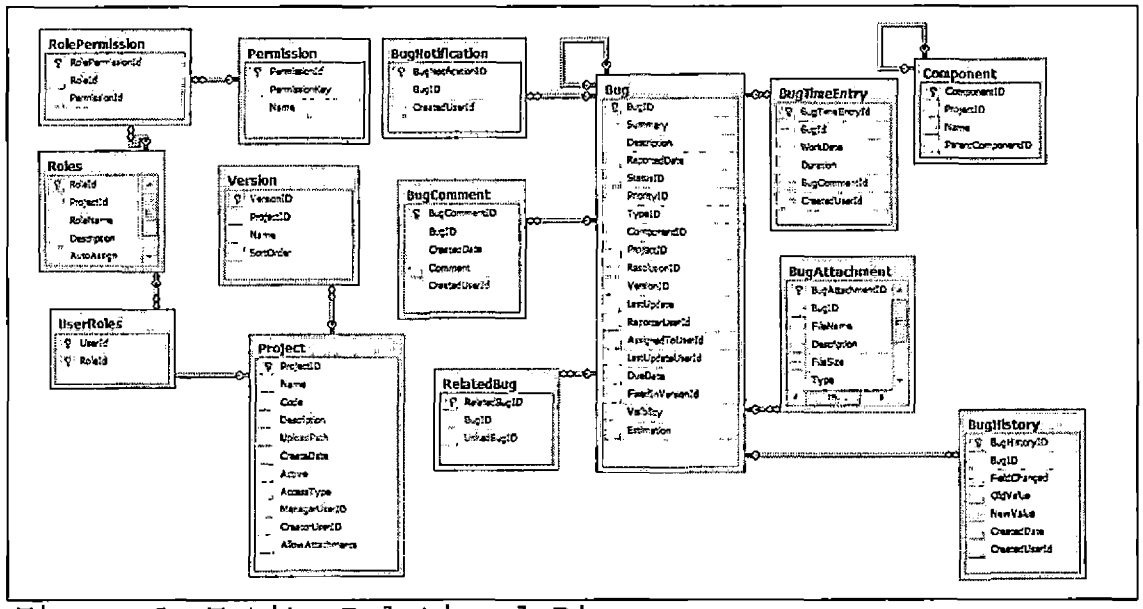


Figure 6. Entity Relational Diagram

## Database Schema Logical Model - Relational Schema

The database relational schema is shown below in Table 1. The primary key for each table is underlined.

Table 1. Database Relational Schema

Bug

| BugID | Summary | Description | ReportedDate | StatusID |
|---|---|---|---|---|
| PriorityID | TypeID | ComponentID | ProjectID | ResolutionID |
| VersionID | Last Update | Reporter UserId | AssignedTo UserId | LastUpdate UserId |

BugAttachment

| BugAttachmentID | BugID | FileName | Description | File Size |
|---|---|---|---|---|
| Type | UploadedDate | Uploaded UserId | | |

BugComment

| BugCommentID | BugID | CreatedDate | Comment | CreatedUserId |
|---|---|---|---|---|

BugHistory

| BugHistoryID | BugID | FieldChanged | OldValue | NewValue |
|---|---|---|---|---|
| CreatedDate | CreatedUserId | | | |

BugNotification

| BugNotificationID | BugID | CreatedUserId | | |
|---|---|---|---|---|

BugTimeEntry

| BugTimeEntryId | BugID | WorkDate | Duration | BugCommentId |
|---|---|---|---|---|
| CreatedUserId | | | | |

Component

| ComponentID | ProjectID | Name | ParentComponentID | |
|---|---|---|---|---|

HostSettings

| SettingName | SettingValue | | | |
|---|---|---|---|---|

Permission

| PermissionId | PermissionKey | Name | | |
|---|---|---|---|---|

Priority

| PriorityID | Name | ImageUrl | | |
|---|---|---|---|---|

| BugID | Summary | Description | ReportedDate | StatusID |
|---|---|---|---|---|

Project

| ProjectID | Name | Code | Description | UploadedPath |
|---|---|---|---|---|
| CreatedDate | Active | Access Type | Manager UserID | ManagerUserID |
| Creator UserID | Allow Attachments | | | |

Resolution

| ResolutionID | Name | | | |
|---|---|---|---|---|

RelatedBug

| RelatedBugID | BugID | LinkedBugID | | |
|---|---|---|---|---|

RolePermission

| RolePermissionId | RoleId | PermissionId | | |
|---|---|---|---|---|

Roles

| RoleId | ProjectId | RoleName | Description | |
|---|---|---|---|---|

Status

| StatusID | Name | | | |
|---|---|---|---|---|

Type

| TypeID | Name | ImageUrl | | |
|---|---|---|---|---|

UserProjects

| UserId | ProjectId | CreatedDate | | |
|---|---|---|---|---|

Version

| VersionID | ProjectId | Name | SortOrder | |
|---|---|---|---|---|

## Database Design

The logical model establishes the following detailed design in Sql database. The following tables describe field, data type, length, primary key, null or non-null keys, and description of the field. The password in the

database is encrypted and it is shown in the web pages as some special character repeated as many times as the length of the actual password.

Table 2. Structure of Table Bug

| Field | Type | Null | Description |
|---|---|---|---|
| BugID | Int | No | A 4 byte length key that uniquely identifies the bug. This identifier is generated by the DBMS as a unique identifier. Therefore I have chosen it to be the primary key in the table. |
| Description | Ntext | No | Gives a detailed explanation of the bug. |
| ReportedDate | Datetime | No | Date and Time (MM/DD/YYYY hh: mm: ss) when the bug is reported. |
| StatusID | Int | No | Indicates if the bug is open, in-progress, closed, re-opened and resolved.<br>Open - The status is set to Open when the bug is reported.<br>In-Progress - The status is set to In-Progress when the user starts working on the bug.<br>Closed - The status is set to closed when the bug is solved.<br>Re-Opened - The status is set to Re-Opened when the closed bug is opened again.<br>Resolved - The Status is set to resolved when the reopened bug is closed. |

| Field | Type | Null | Description |
|---|---|---|---|
| PriorityID | Int | No | Identifies whether the bug is major, minor, critical and trivial. Major - The priority is set to major if the bug to be fixed is of greater importance. Minor - The priority is set to minor if the bug to be fixed is of lesser importance. Critical - The Priority is set to critical if the bug needs to be fixed urgently. Trivial - The Priority is set to trivial if the bug to be fixed is of very little importance. |
| TypeID | Int | No | Identifies the type of an issue. It can be a new feature, improvement and task. |
| ProjectID | Int | No | Identifies the project to which this bug is associated. |
| LastUpdate | Datetime | No | Date and Time (MM/DD/YYYY hh: mm: ss) when the bug is last updated. |
| ReporterUserId | uniqueidentifier | No | Identifier of the user who reports the bug. |
| AssignedToUserId | Uniqueidentifier | | Identifier of the user who will correct the bug. |
| LastUpdateUserId | uniqueidentifier | No | Identifier of the user who last updated the bug. |
| DueDate | Datetime | | Latest Date (MM/DD/YYYY hh: mm: ss) by which the bug needs to be fixed. |

Table 3. Structure of Table Bug Attachment

| Field | Type | Null | Description |
|---|---|---|---|
| BugAttachmentID | Int | No | A 4 byte length key that uniquely identifies the bug attachment. This identifier is generated by the DBMS as a unique identifier. Therefore I have chosen it to be the primary key in the table. |
| BugID | Int | No | Identifies the bug to which this bug attachment is associated. |
| FileName | nvarchar(100) | No | Name of the uploaded file. |
| Description | nvarchar(80) | | Gives a detailed explanation of the uploaded file. |
| FileSize | Int | | Size of the file in bytes. |
| Type | nvarchar(50) | | Type of the uploaded file extension, For e.g., .doc, .txt, .html and .exe. |
| UploadedDate | Datetime | No | Date and time (MM/DD/YYYY hh: mm: ss) when the file is uploaded. |
| UploadedUserId | uniqueidentifier | No | Identifier of the user who uploaded the file. |

Table 4. Structure of Table BugComment

| Field | Type | Null | Description |
|---|---|---|---|
| BugCommentID | Int | No | A 4 byte length key that uniquely identifies the bug comment. This identifier is generated by the DBMS as a unique identifier. Therefore I have chosen it to be the primary key in the table. |
| BugID | Int | No | Identifies the bug to which this bug comment is associated. |
| CreatedDate | Datetime | No | Date and time (MM/DD/YYYY hh: mm: ss) when the user comments on the bug. |
| Comment | Ntext | | Gives a detailed explanation of the comment. |
| CreatedUserId | Uniqueidentifer | No | Identifier of the user who comments on the bug. |

Table 5. Structure of Table BugHistory

| Field | Type | Null | Description |
|---|---|---|---|
| BugHistoryID | Int | No | A 4 byte length key that uniquely identifies the bug history. This identifier is generated by the DBMS as a unique identifier. Therefore I have chosen it to be the primary key in the table. |
| BugID | Int | No | Identifies the bug to which this bug history is associated. |
| FieldChanged | nvarchar(50) | | Specifies any change in the bug details such as status, priority and type. |
| OldValue | nvarchar(50) | | Previous Value of the field. |
| NewValue | nvarchar(50) | | New Value of the field. |
| CreatedDate | Datetime | No | Date and time (MM/DD/YYYY hh: mm: ss) when the bug history was created. |
| CreatedUserId | uniqueidentifier | No | Identifier of the user who does any changes in the bug details such as status, priority and type. |

Table 6. Structure of BugNotification

| Field | Type | Null | Description |
|---|---|---|---|
| BugNotificationID | Int | No | A 4 byte length key that uniquely identifies the bug notification. This identifier is generated by the DBMS as a unique identifier. Therefore I have chosen it to be the primary key in the table. |
| BugID | Int | No | Identifies the bug to which this bug notification is associated. |
| CreatedUserId | uniqueidentifier | No | Identifier of the user who sends the message related to the bug. |

Table 7. Structure of Table BugTimeEntry

| Field | Type | Null | Description |
|---|---|---|---|
| BugTimeEntryId | Int | No | A 4 byte length key that uniquely identifies the bug time entry. This identifier is generated by the DBMS as a unique identifier. Therefore I have chosen it to be the primary key in the table. |
| BugID | int | No | Identifies the bug to which this bug time entry is associated. |
| WorkDate | Datetime | | Date and time (MM/DD/YYYY hh: mm: ss) when the user starts working on the bug. |
| Duration | decimal (4,2) | | Number of hours taken to work on the bug. |
| BugCommentId | Int | No | Identifies the bug comment to which this bug time entry is associated. |
| CreatedUserId | uniqueidentifier | No | Identifier of the user who enters the time spent on the bug. |

Table 8. Structure of Table Component

| Field | Type | Null | Description |
|---|---|---|---|
| ComponentID | Int | No | A 4 byte length key that uniquely identifies the component. This identifier is generated by the DBMS as a unique identifier. Therefore I have chosen it to be the primary key in the table. A component can be categorized as a web-application, or windows-application. |
| ProjectID | Int | No | Identifies the project to which this component is associated. |
| Name | nvarchar(50) | | Name of the component |
| ParentComponentID | Int | No | Root identification number of the component |

Table 9. Structure of Table Permissions

| Field | Type | Null | Description |
|---|---|---|---|
| PermissionId | int | No | A 4 byte length key that uniquely identifies the permission. This identifier is generated by the DBMS as a unique identifier. Therefore I have chosen it to be the primary key in the table. |
| PermissionKey | nvarchar(50) | | Gives permission to user to add, edit, view and delete the bug. |
| Name | nvarchar(50) | No | Names the permission type such as add, edit, view and delete. |

Table 10. Structure of Table Priority

| Field | Type | Null | Description |
|---|---|---|---|
| PriorityID | int | No | A 4 byte length key that uniquely identifies the priority. This identifier is generated by the DBMS as a unique identifier. Therefore I have chosen it to be the primary key in the table. |
| Name | nvarchar(50) | No | Differentiates the priority such as major, minor, critical and trivial. |
| ImageUrl | nvarchar(50) | | Image symbol of Priority. |

Table 11. Structure of Table Project

| Field | Type | Null | Description |
|---|---|---|---|
| ProjectID | Int | No | A 4 byte length key that uniquely identifies the project. This identifier is generated by the DBMS as a unique identifier. Therefore I have chosen it to be the primary key in the table. |
| Name | nvarchar(50) | | Name of the project. |
| Code | nvarchar(3) | | 3 character length key that uniquely identifies the project. |
| Description | nvarchar(1000) | | Gives a detailed explanation of the project. |
| UploadedPath | nvarchar(80) | | Path of the file that was uploaded. |
| CreatedDate | Datetime | No | Date and time (MM/DD/YYYY hh: mm: ss) when the user created the project. |
| Active | Int | No | Gives the status of the project (yes or no). |
| AccessType | int | | Access Type is categorized as public or private indicating if the project can be accessed by all the users or only those who have access. |
| ManagerUserID | uniqueidentifier | No | Identifier of the manager for the project. |
| CreatorUserID | uniqueidentifier | No | Identifier of the user who created the project. |

Table 12. Structure of Table Resolution

| Field | Type | Null | Description |
|---|---|---|---|
| ResolutionID | int | No | A 4 byte length key that uniquely identifies the resolution. This identifier is generated by the DBMS as a unique identifier. Therefore I have chosen it to be the primary key in the table. |
| Name | nvarchar(50) | No | Names the resolution type such as fixed, invalid, won't fix and Duplicate. |

Table 13. Structure of Table RelatedBug

| Field | Type | Null | Description |
|---|---|---|---|
| RelatedBugID | int | No | A 4 byte length key that uniquely identifies the related bug. This identifier is generated by the DBMS as a unique identifier as I have chosen it to be the primary key in the table. |
| BugID | Int | No | Identifies the bug to which this related bug is associated. |

Table 14. Structure of Table Roles

| Field | Type | Null | Description |
|---|---|---|---|
| RoleId | int | No | A 4 byte length key that uniquely identifies the role. This identifier is generated by the DBMS as a unique identifier. Therefore I have chosen it to be the primary key in the table. |
| ProjectId | int | | Identifies the project to which this role is associated. |
| RoleName | nvarchar(256) | No | Name of the role such as project manager, developer, tester and reporter. |
| Description | nvarchar(256) | | Gives a detailed explanation of the role. |

Table 15. Structure of Table Status

| Field | Type | Null | Description |
|---|---|---|---|
| StatusID | int | No | A 4 byte length key that uniquely identifies the status. This identifier is generated by the DBMS as a unique identifier. Therefore I have chosen it to be the primary key in the table. |
| Name | nvarchar(50) | No | Gives the status of the bug such as open, in-progress, closed, reopened and resolved. |

Table 16. Structure of Table UserRoles

| Field | Type | Null | Description |
|---|---|---|---|
| UserId | uniqueidentifier | No | Identifies the logged in user. |
| RoleId | int | No | Identifies the role to which this user is associated. |

Table 17. Structure of Table Version

| Field | Type | Null | Description |
|---|---|---|---|
| VersionID | int | No | A 4 byte length key that uniquely identifies the version. This identifier is generated by the DBMS as a unique identifier. Therefore I have chosen it to be the primary key in the table. |
| ProjectId | int | | Identifies the project to which this version is associated. |
| Name | nvarchar(50) | | Name of the version |
| SortOrder | int | | Sorts out the order of version such as ascending or descending order. |

# CHAPTER FOUR

# PROJECT IMPLEMENTATION

## User Interface Design

The Enterprise Management Studio is a pure web-based application programmed with Asp.net.

Asp.net supports the most popular browsers: Internet Explorer, Firefox, Google, Chrome etc. Therefore, the user interfaces of EMS support most popular web browsers and can be accessed by all kinds of users.

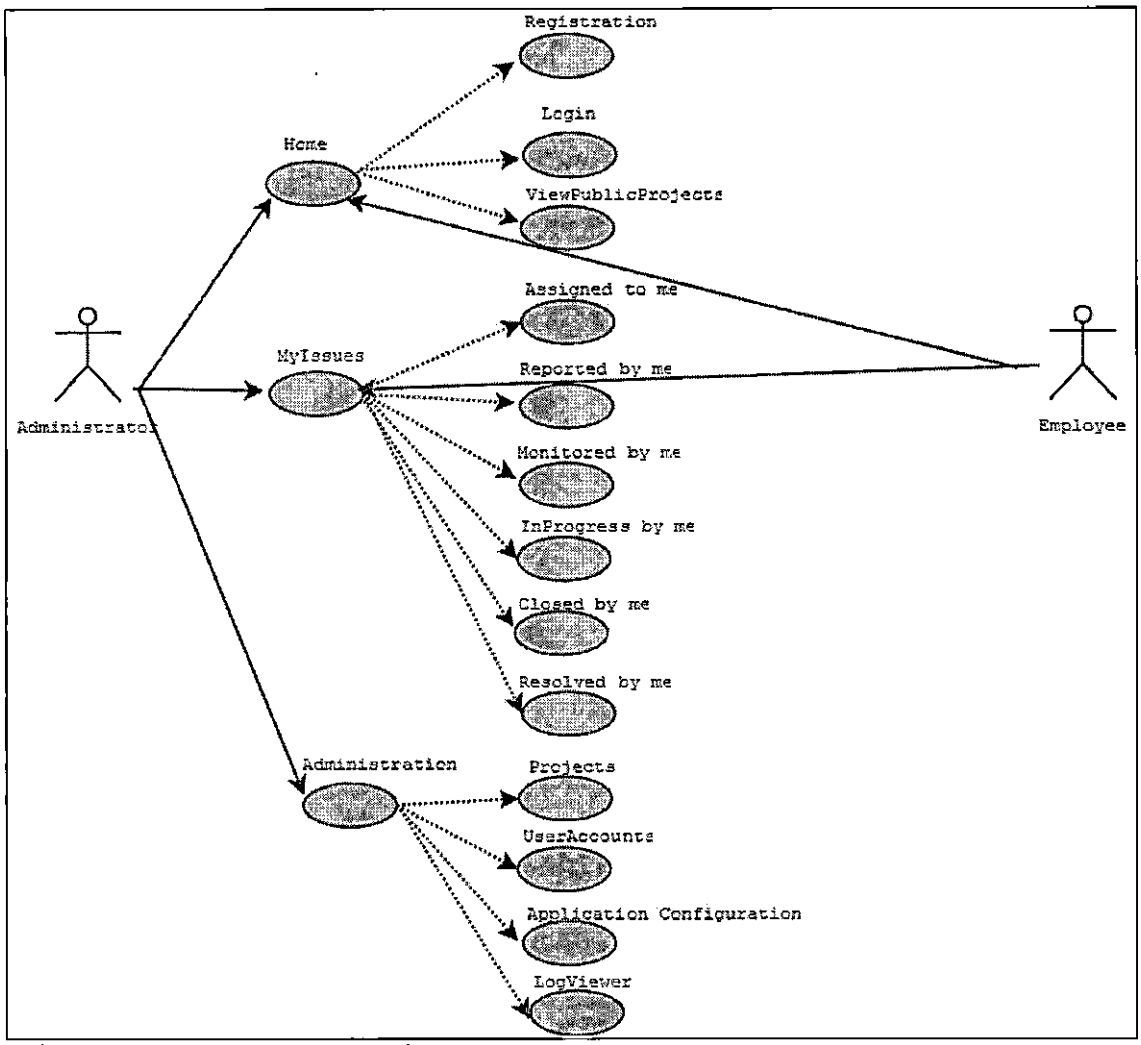


Figure 7. Use Case Diagram

All interfaces are generated dynamically on the server side and can fully or partially update the client pages, which gives the user a good experience of surfing the web-application. The above figure is the Use Case Diagram for the system. The users will be guided to their interface via the login page. The system will redirect the

user on the basis of their access rights. The interface is windows point-and-click based and is easy to use.

Login Page

This page will be the first page that all users will see when they enter EMS and this is also the home page of the system. It provides both administrator and user logins, and will contain a welcome message to the users. The page checks the validity of the user by checking the username and password that the user enters with the one in the database. Once the user is authenticated, the page redirects him/her according to his/her predetermined access rights.

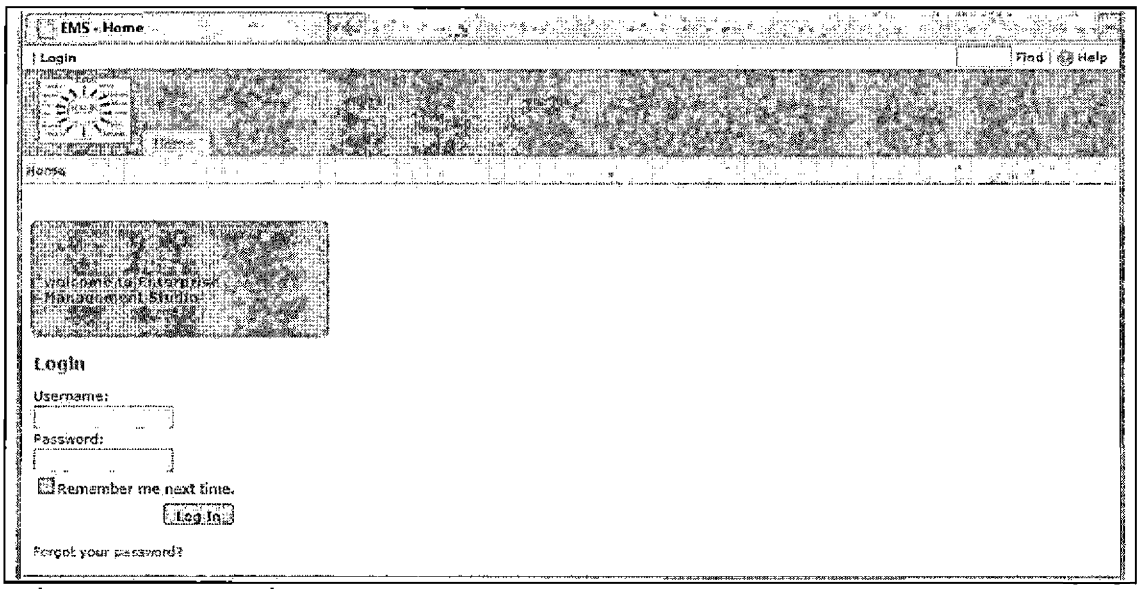


Figure 8. Login Page

After Login Page

This page is visible to the administrator after he/she logs in to the application. Two panels have been used in this page: the first panel is called "ProjectPanelHeader," which shows the project title and manager of the project. The second panel is called "ProjectPanelDetails," which displays the project description and a few links below it. The links on the left hand side act like the shortcuts to the respective pages. Each of them will be described in further detail. The links on the right hand side of the page give the latest information about the issues. The Ajax collapsible panel extender control has been used in this page for collapsing and extending the image.

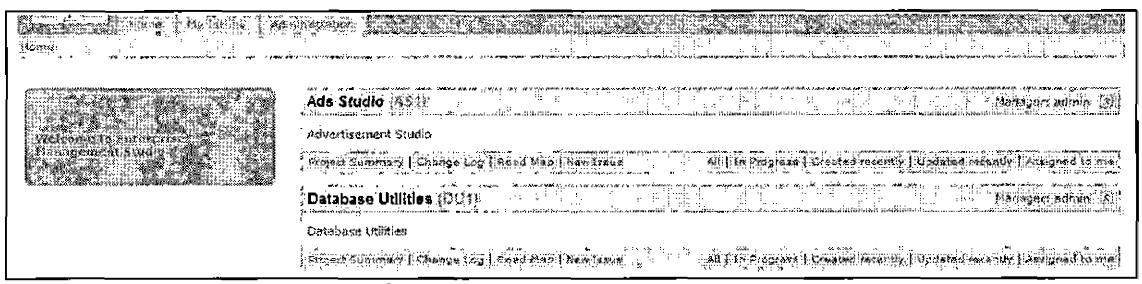


Figure 9. After Login Page

Administration Page

This page is visible only to the administrator and the managers of the project. This interface is used by the

administrator to do various things, which includes creating projects, managing user accounts, application configuration and log viewer. Administrator of the system will be responsible for doing the entire initial set up.

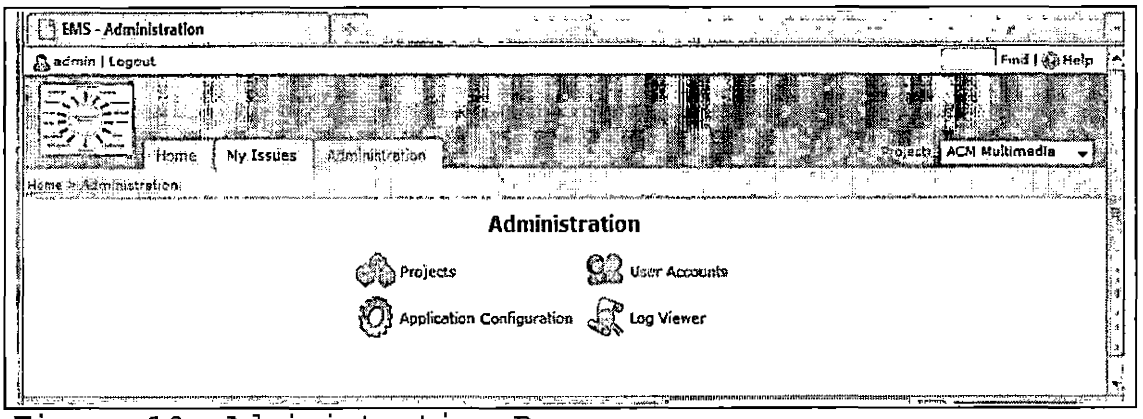

Figure 10. Administration Page

Projects Page

This page is visible only to the administrator and the project manager. This page is used to create new projects and is also used to clone a project. The project manager will not have the access rights to create a new project. This can be done only by the administrator. It also gives the list of all the projects that have been created recently. The project information can be edited on this page when the user clicks on the links in the list. The Sitemap control has been used to display the navigation menu in all the pages.

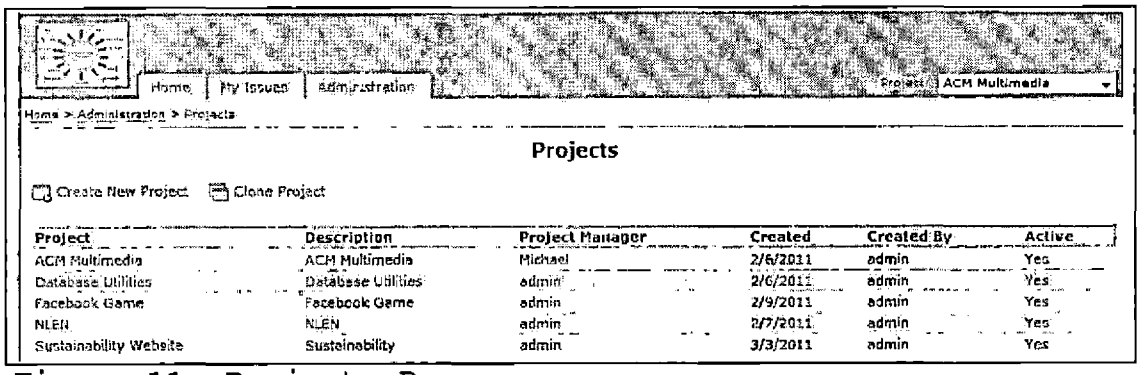

Figure 11. Projects Page

Create New Project Page

This page is visible only to the administrator. The project manager does not have the access rights to create a new project. A wizard step control has been used to create a new project. The wizard comprises of seven steps, after which the project will be created. The administrator has the privilege to add, modify, delete, and view the project information in the wizard steps. The wizard has to be completed to create a new project.

New Project Wizard (Step 1 of 7) Page

This is the first step of the wizard, which is the Introduction Step. The administrator has an option to skip this step the next time he creates a project. In order to do that, the "Skip Next Time" checkbox needs to be checked.

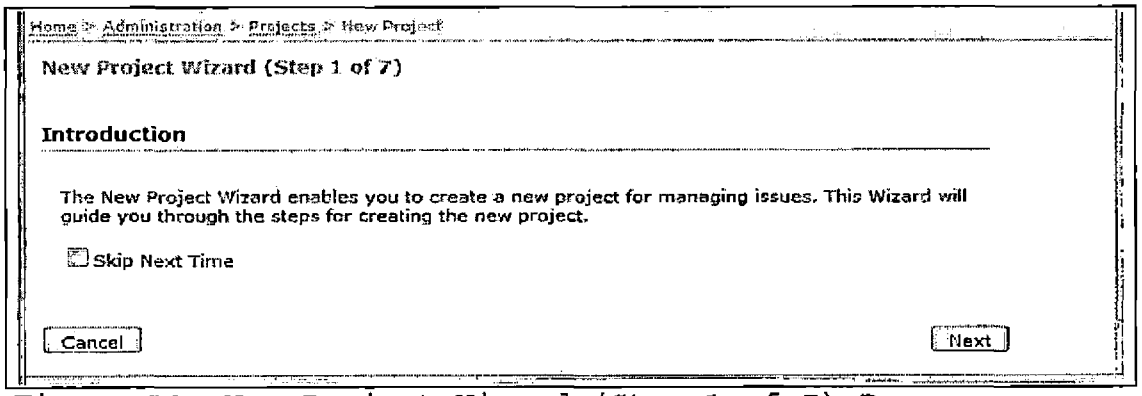

Figure 12. New Project Wizard (Step 1 of 7) Page

New Project Wizard (Step 2 of 7) Page

To create a new project, the administrator must type in the project name, description, code, select the manager for the project from the drop-down menu, checks the allow attachments checkbox if he wants to allow any attachments, type the upload path where he wants the attachments to be uploaded to and selects the access type for the project. By default, the status of the project will be active. After adding all the information, the administrator will click the "Next" button to go to the next step in the wizard. The previously added information can be edited by selecting the project from the "Projects" Page. This action will populate the fields with previously entered data. After editing, the information will be saved when the administrator clicks on the "Save" button. The Project can be deleted by clicking the "Delete" button.

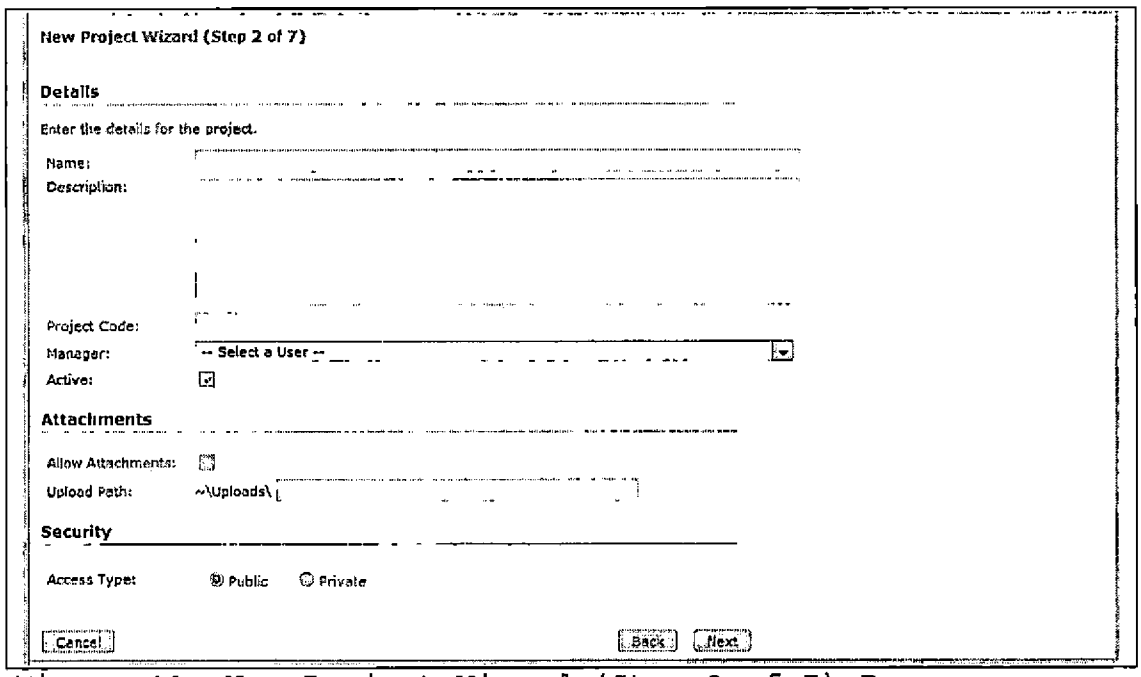

Figure 13. New Project Wizard (Step 2 of 7) Page

New Project Wizard (Step 3 of 7) Page

A Project is identified by the category it belongs to. For example, a project can be a windows-based or a web-based one. This page is used to add categories for a project. The administrator has to click on "Root Category" first, type the name of the category in order to add the category for a project. Otherwise, the category will not appear on the screen. To create sub-categories for a project, the administrator will select the parent category and click the "Add Category" link. To delete a category, the child categories have to be deleted first followed by the root (parent) categories.

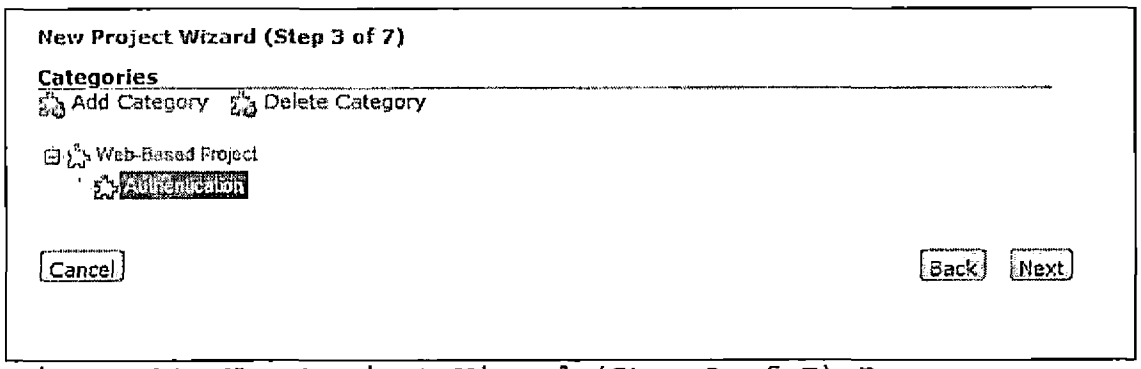


Figure 14. New Project Wizard (Step 3 of 7) Page

New Project Wizard (Step 4 of 7) Page

This Page is used to add the version for a project. The version number has to be typed in the textbox provided and "Add Version" link needs to be clicked for the version to appear. Once the version is added, it can be edited on the same screen.

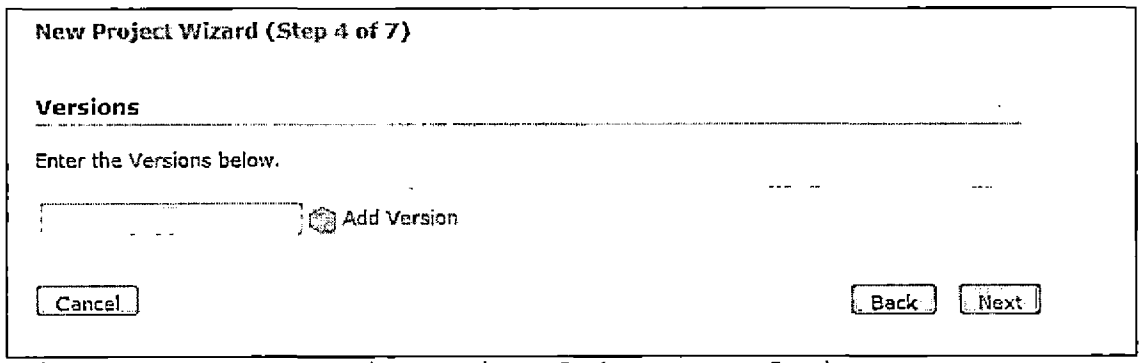


Figure 15. New Project Wizard (Step 4 of 7) Page

New Project Wizard (Step 5 of 7) Page

This page displays the list of roles that have been created. When a new project is created, some roles are predefined for the user. To add a new role and assign

permissions to that role, the "Add New Role" link needs to be clicked.

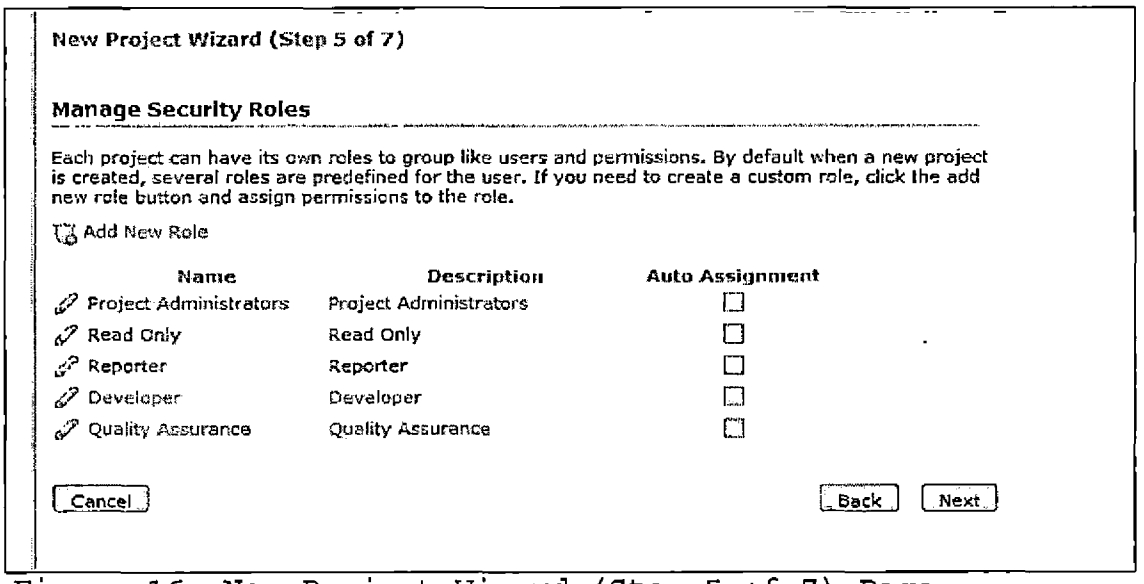


Figure 16. New Project Wizard (Step 5 of 7) Page

Add New Role Page

This page is used to add a new role for a project. If the "Auto Assignment?" box is checked for a particular role in a project, all the newly created users will be assigned to that role automatically. The project members will be authorized by the project administrator to add, edit, view and delete the project information such as comments, issues, attachments, descriptions etc.

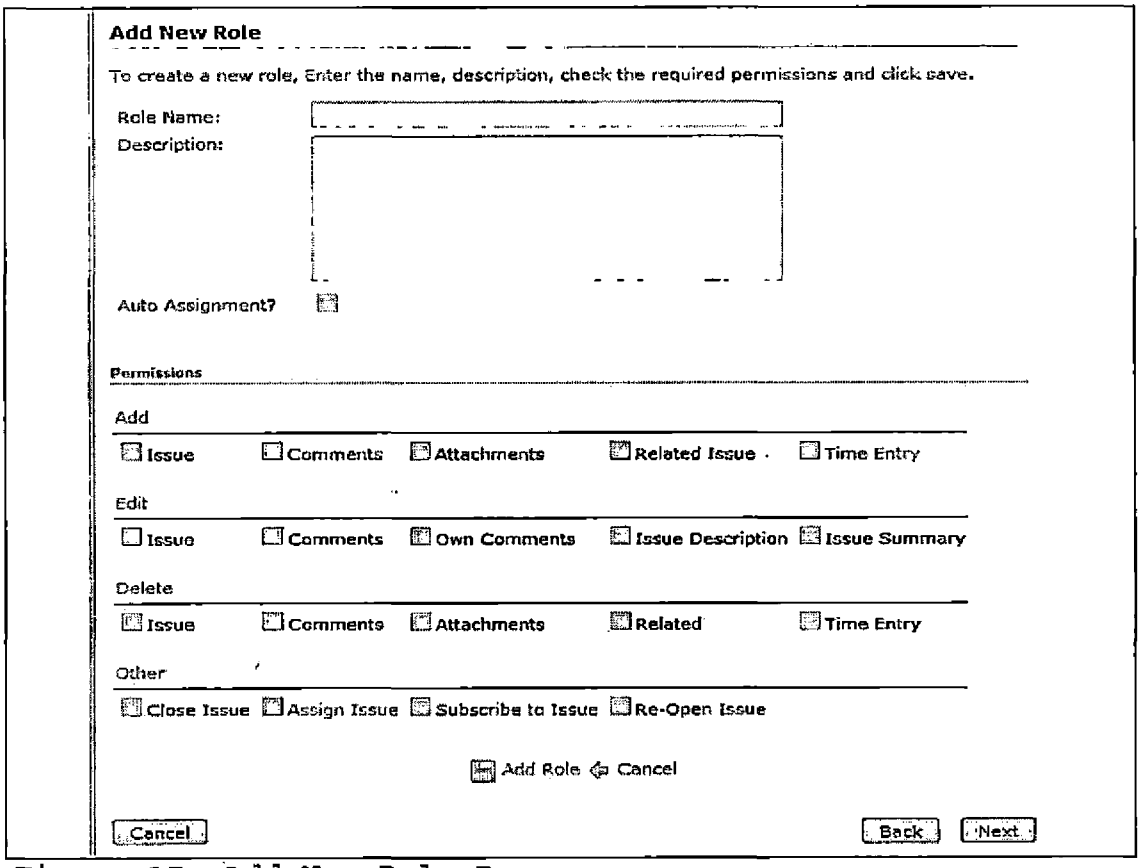


Figure 17. Add New Role Page

## New Project Wizard (Step 6 of 7) Page

This page is used to select the members for the project and assign roles to them.

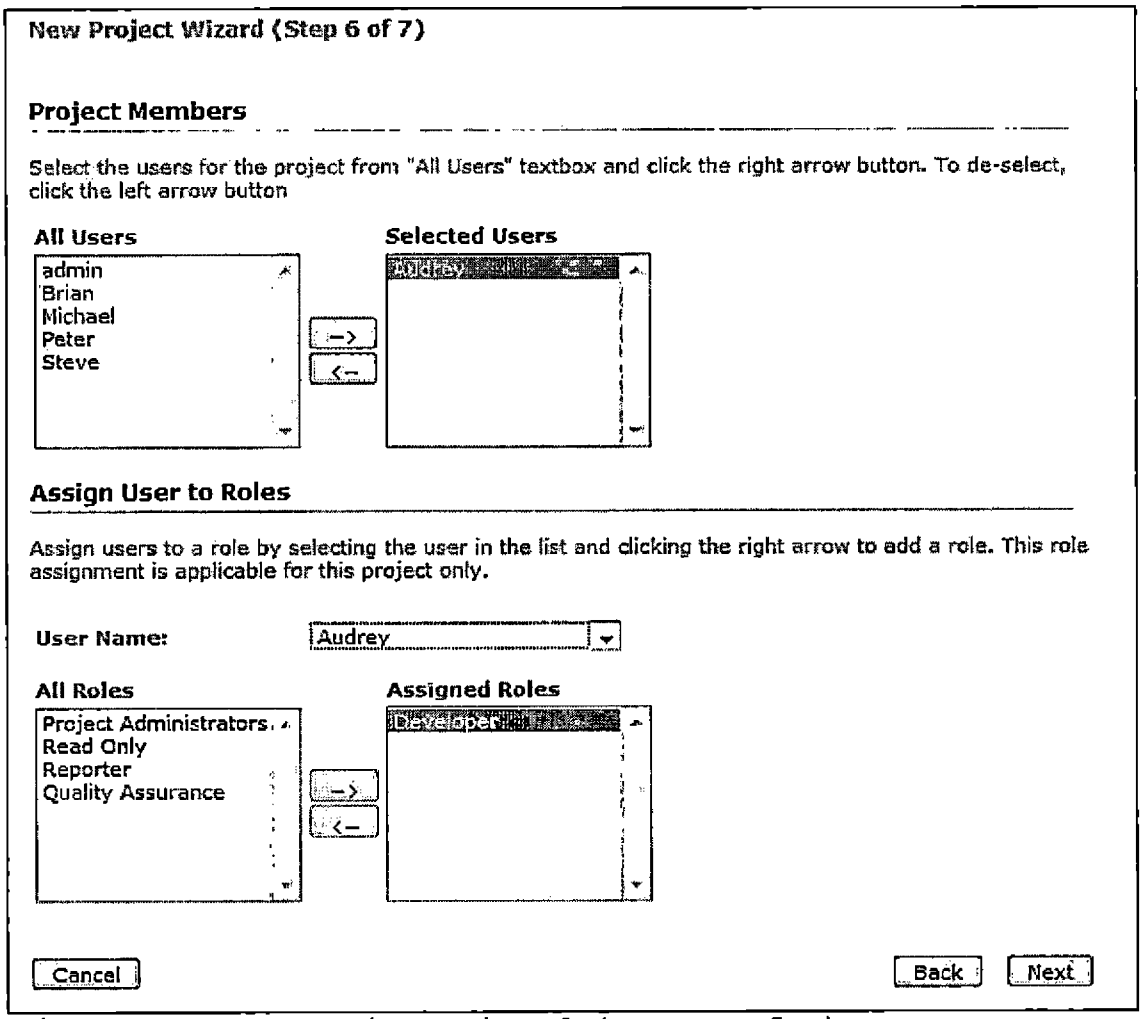


Figure 18. New Project Wizard (Step 6 of 7) Page

## Project Wizard (Step 7 of 7) Page

This is the final step in the project creation page. The administrator arrives at this page once he completes all the above steps in the wizard.

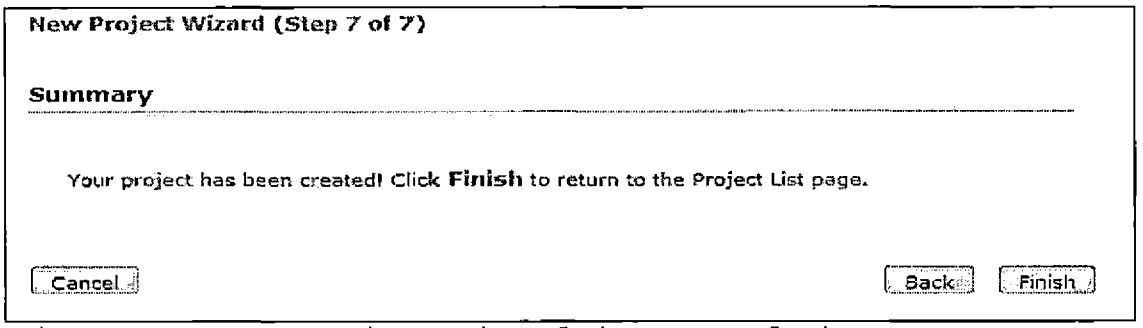


Figure 19. New Project Wizard (Step 7 of 7) Page

New Issue Page

This page is used to create a new issue for a project. Issue here could be either a bug or a task of a project. The FCK Editor panel has been used in this page for the description. The page also provides an option to upload any attachments to the issue.

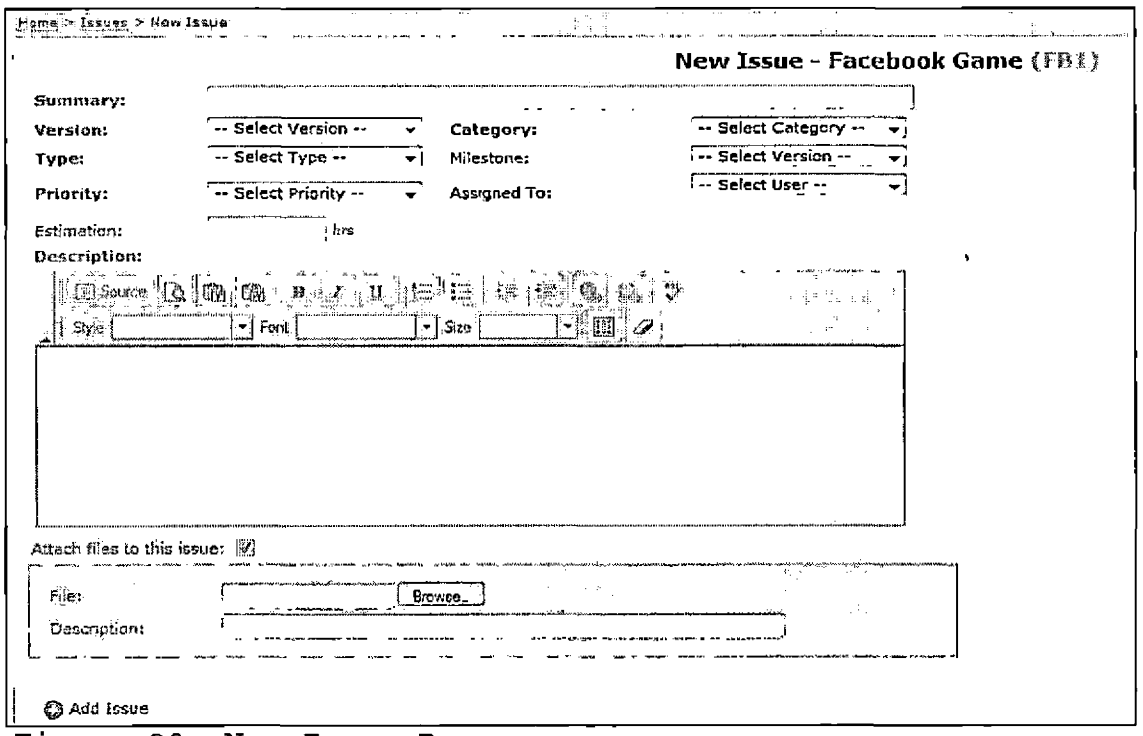

Figure 20. New Issue Page

## My Issues Page

This page displays a list of all actions performed on the issues such as Assigned, Reported, Monitored, In Progress, Closed, and Resolved. The Data Grid View control has been used to display the columns in the page. The page also displays the project id, type, priority, version, milestone, category, assigned to, reporter, due date, last update and created date columns for each.

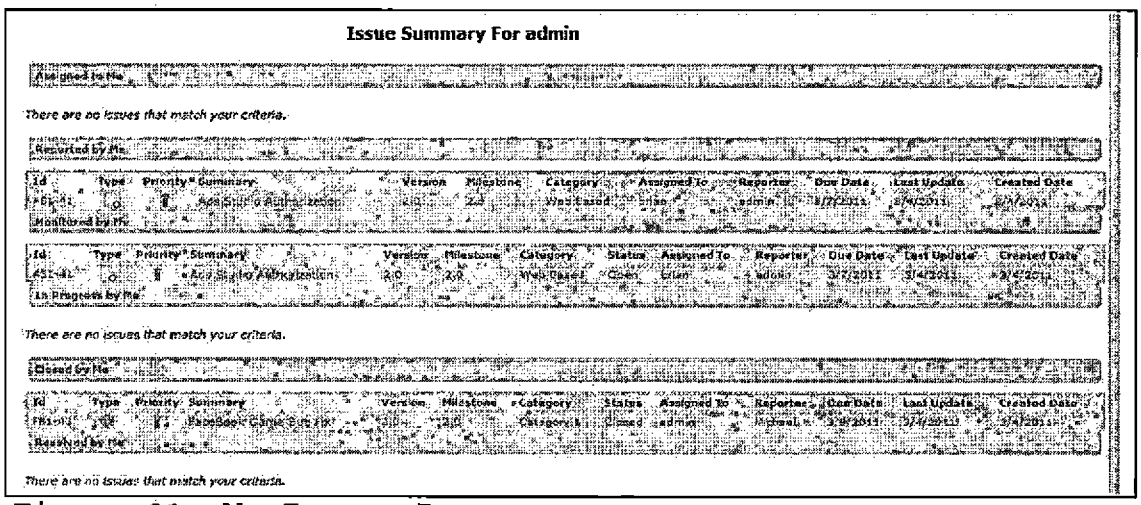

Figure 21. My Issues Page

## Issue Details Page

This page displays the issue description, details and also gives the time information. The user can edit the details by selecting the item from the drop down list. To save the changes to the database, the users will have to click on "Save". The page also has five tabs at the bottom which allows the user to comment on the issue, display issue history, send notifications to the people concerned, add related issues and enter the time tracking information.

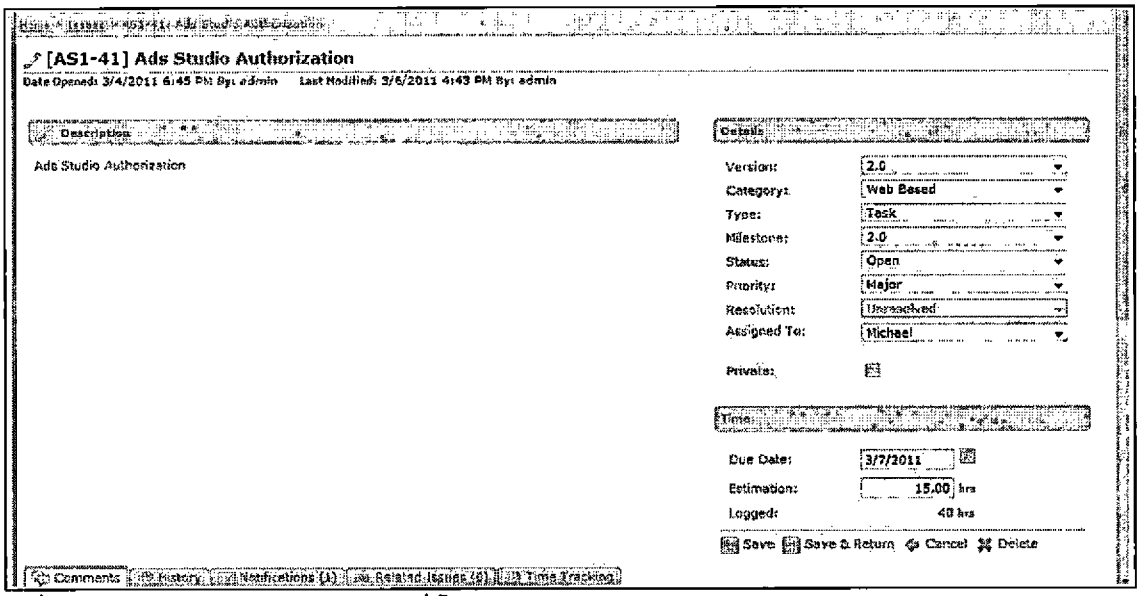

Figure 22. Issue Details Page

## Comments Tab Page

This page is used to add comments to the issue. The FCK editor has been used to add the comment description. Once the comment is added, it will be displayed above the FCK Editor Panel.

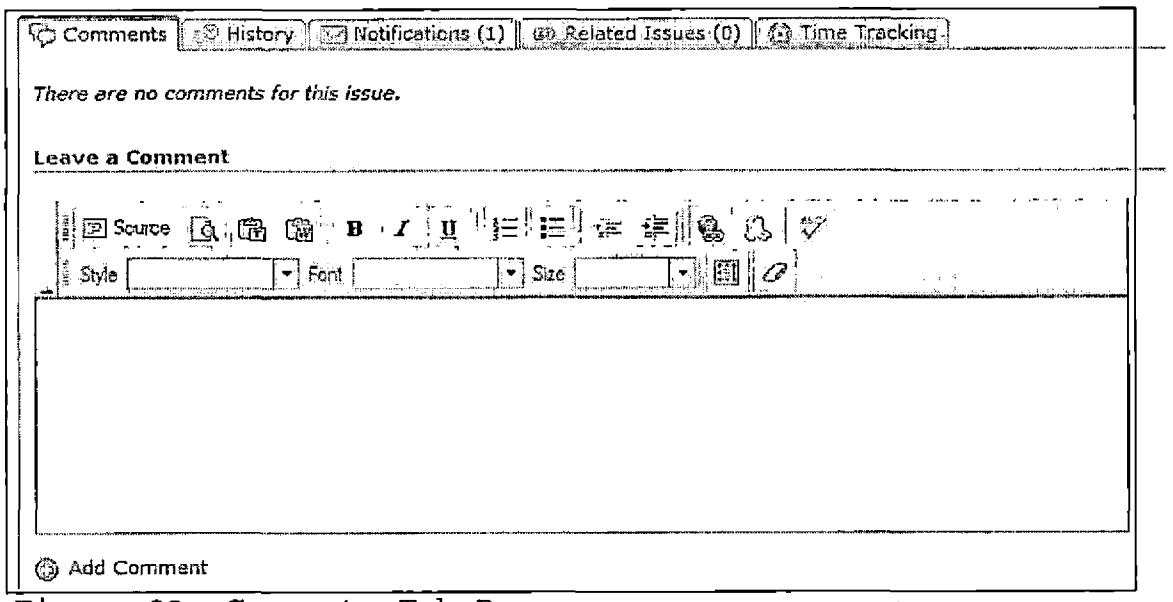


Figure 23. Comments Tab Page

## History Tab Page

This page gives a list of all the items that have been changed, their old and new values, date modified and the username.

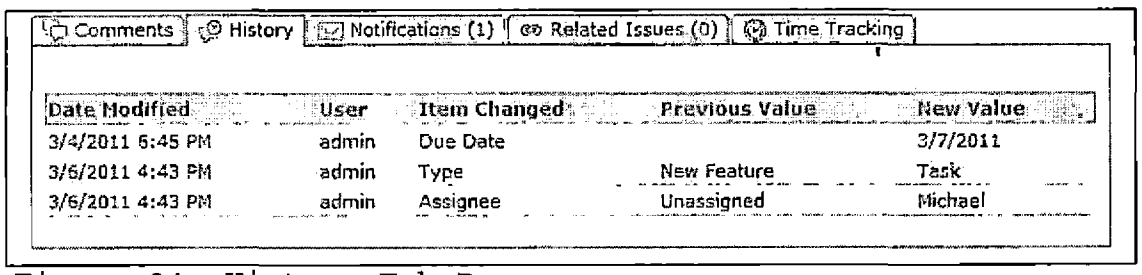


Figure 24. History Tab Page

## Notifications Tab Page

This page is used to send notification messages to the selected users in the project. The administrator and

the project managers only have the access rights to add and remove the selected users to/from the project. When the issue is updated, the logged in user will receive email notifications when "Receive Notifications" button is clicked and the notifications will not be received on clicking "Don't Receive Notifications" button.

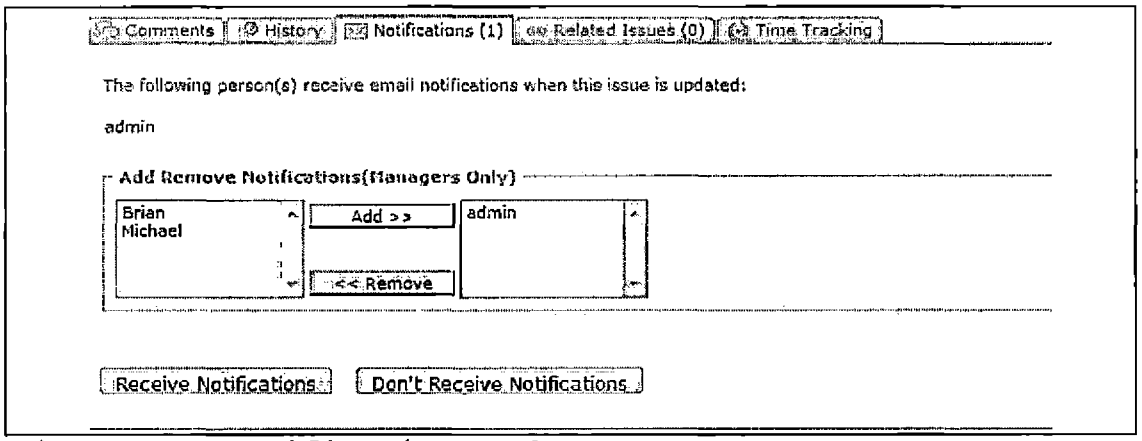


Figure 25. Notifications Tab Page.

Add Related Issue Tab Page

This page is used to add the related issues in the project. A user can add the related issue, if, and only if, the issue belongs to the same project. It gives an error message if the issue does not belong to the same project.

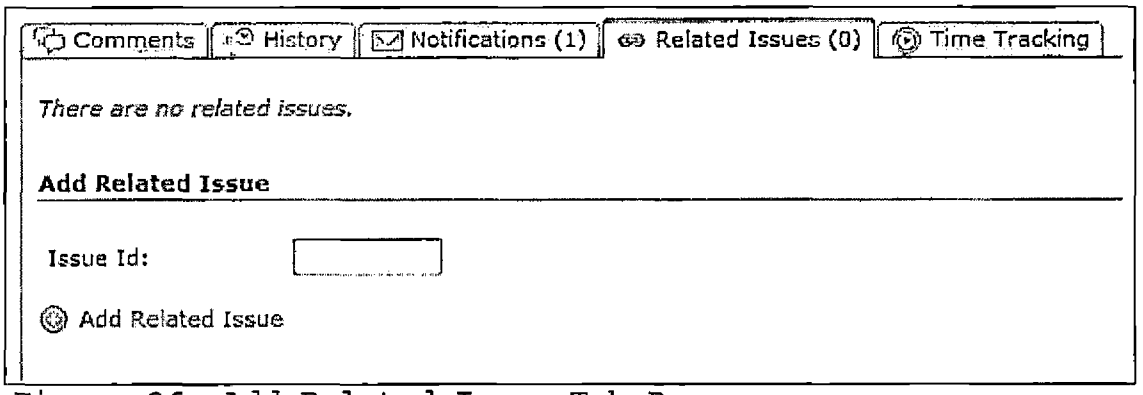

Figure 26. Add Related Issue Tab Page

## Time Tracking Tab Page

This page is used to record the time spent each day on any given task. To record the time, the user will have to type the number of hours in duration and click on "Add Time Entry" button. This will populate the time entry details under that task. To delete any time that has been previously entered, the delete link has to be clicked.

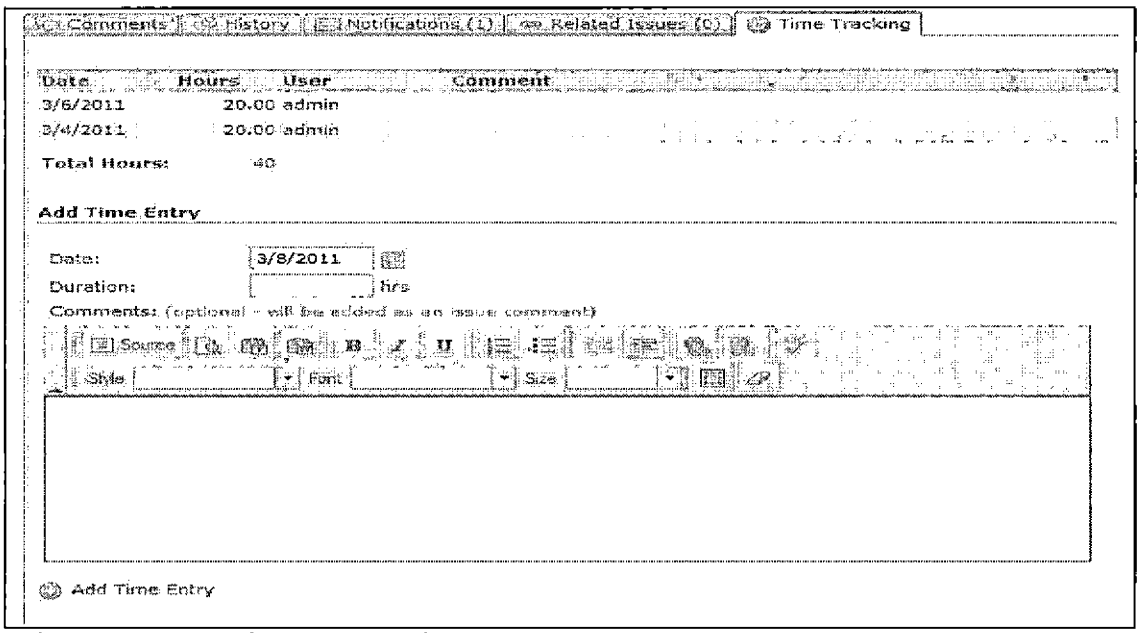

Figure 27. Time Tracking Tab Page

Project Summary Page

This page displays the summary details of the project. The links on the page, when clicked, take the users to the respective pages.

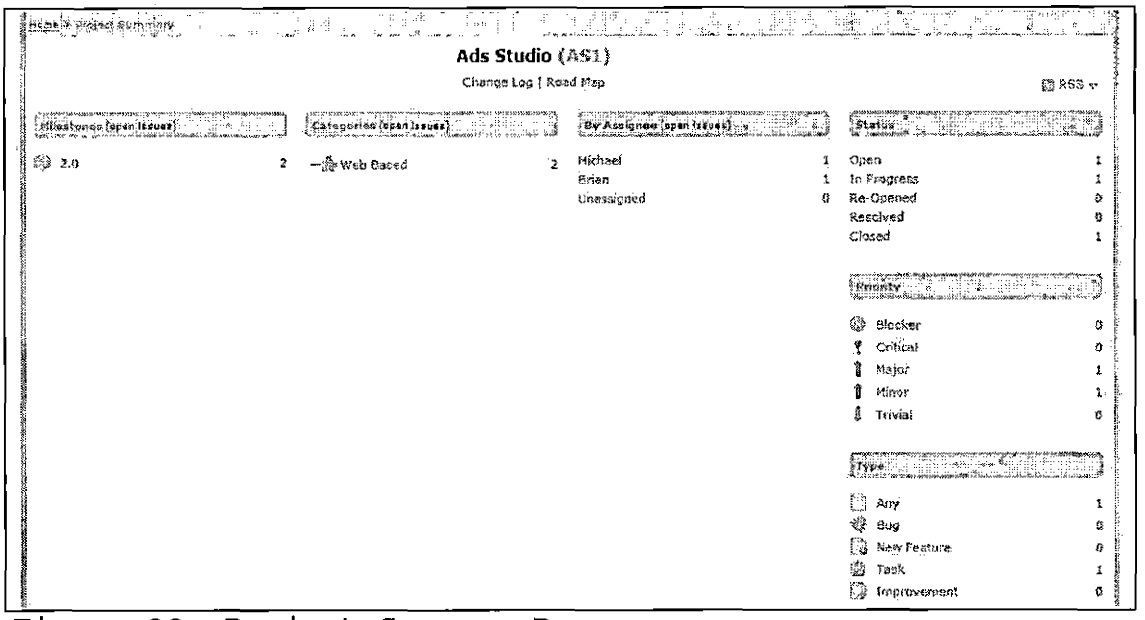


Figure 28. Project Summary Page

## Change Log Page

This page gives the list of the recently closed issues in a project. It provides details such as the category, type, project ID, summary, to whom the task has been assigned, and status.

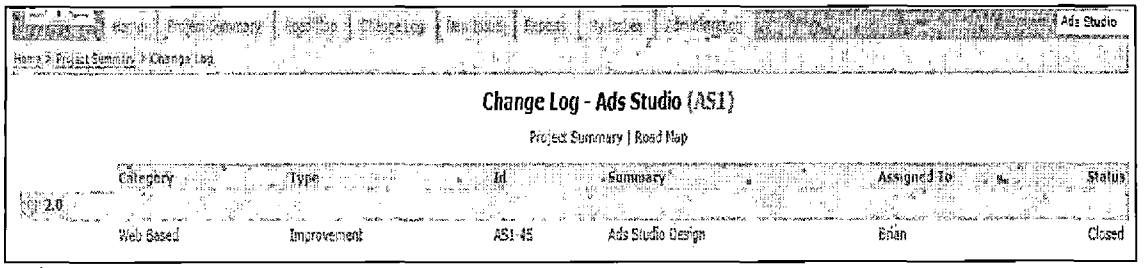


Figure 29. Change Log Page

Road Map Page

This page gives the status of the project completed in terms of percentages. It also gives the ratio of the number of issues that have been resolved and the total number of issues.

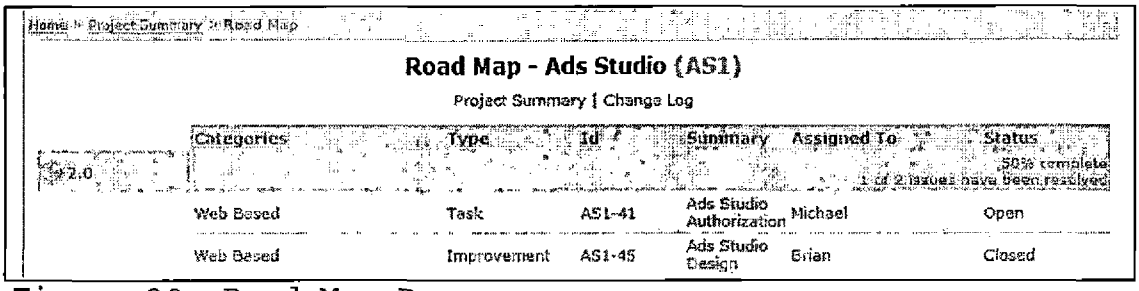

Figure 30. Road Map Page

Manage User Accounts Page

This page is visible only to the administrator and the project managers. It provides an easy interface to add, edit, and delete the users. This page also provides the ability to search for the users through usernames or Email IDs. All the users who are given access to the system are authorized by default.

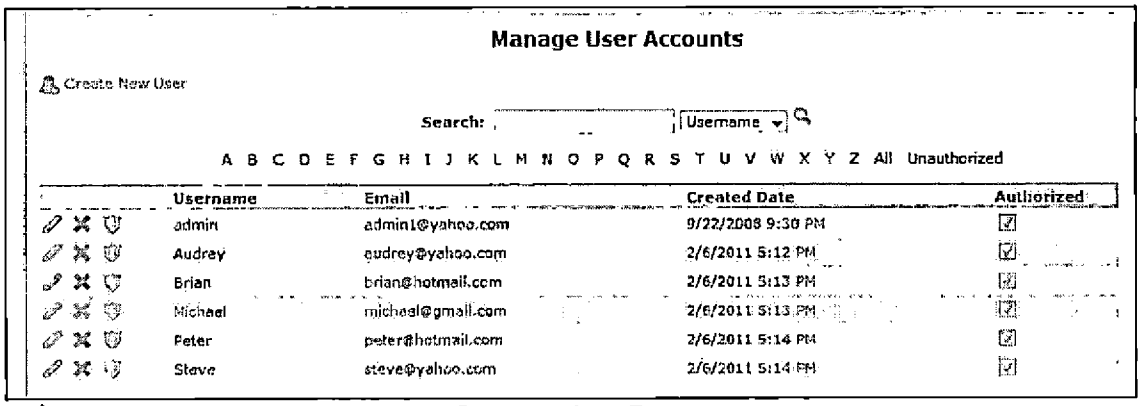


Figure 31. Manage User Accounts Page

## Create New User Page

This page is used to create a new user. The information in the textboxes below needs to be filled in to create a new user. If they are left blank, it gives an error message.

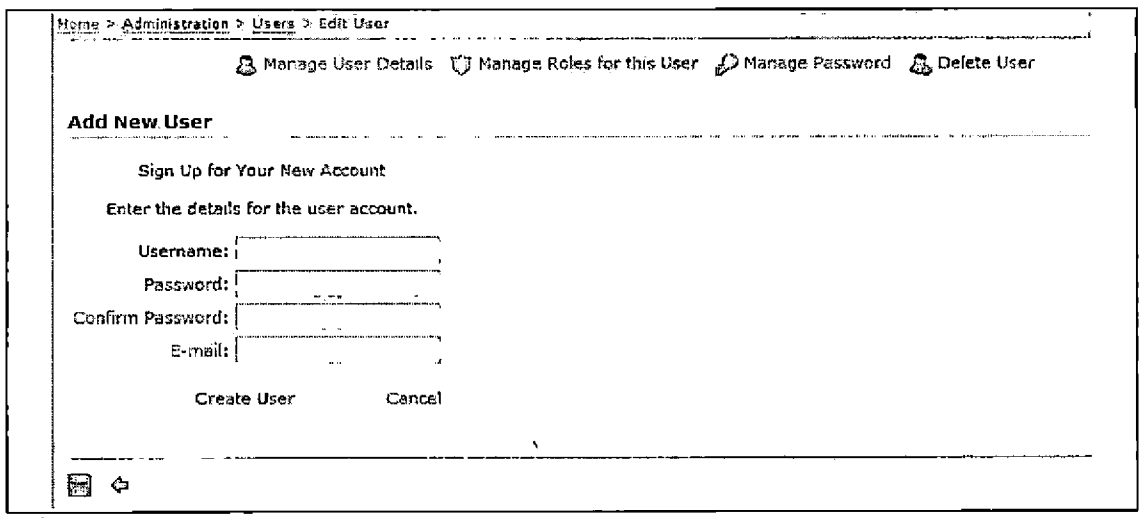


Figure 32. Create New User Page

Edit User Page

This page is used to edit the information related to the names of the users and their email IDs. It also displays the date when the ID was created, the date when the user logged in recently, and the last activity date.

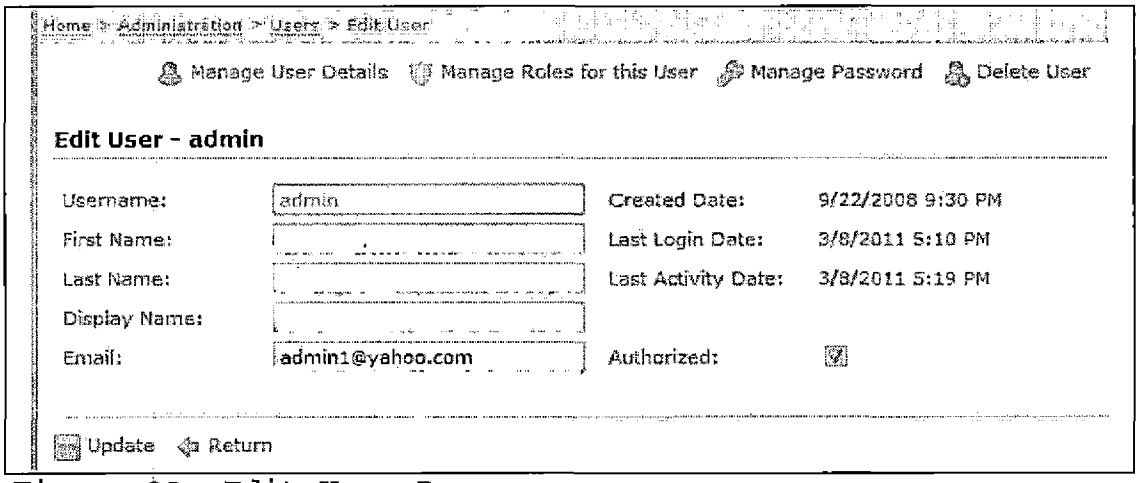


Figure 33. Edit User Page

Manage Roles Page

This page displays a list of all the roles of the user in all the projects. It can also be used to add a new role by selecting from the drop down list and clicking the "Add Role" link. The "Update" link when clicked saves the changes to the database.

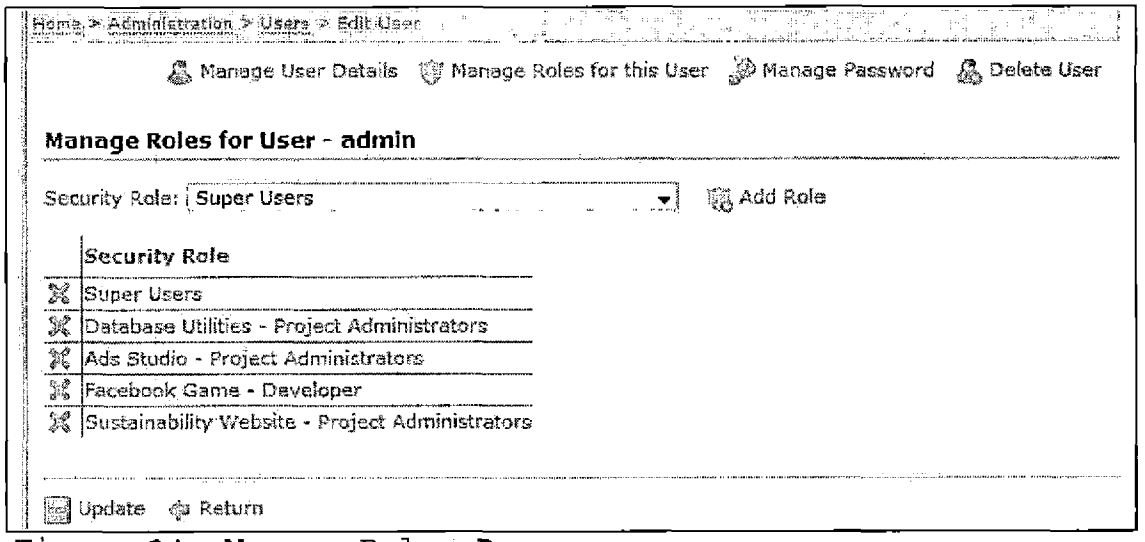


Figure 34. Manage Roles Page

## Manage Password Page

In this page, users can change their passwords after they successfully login. The page also provides the link to reset the password.

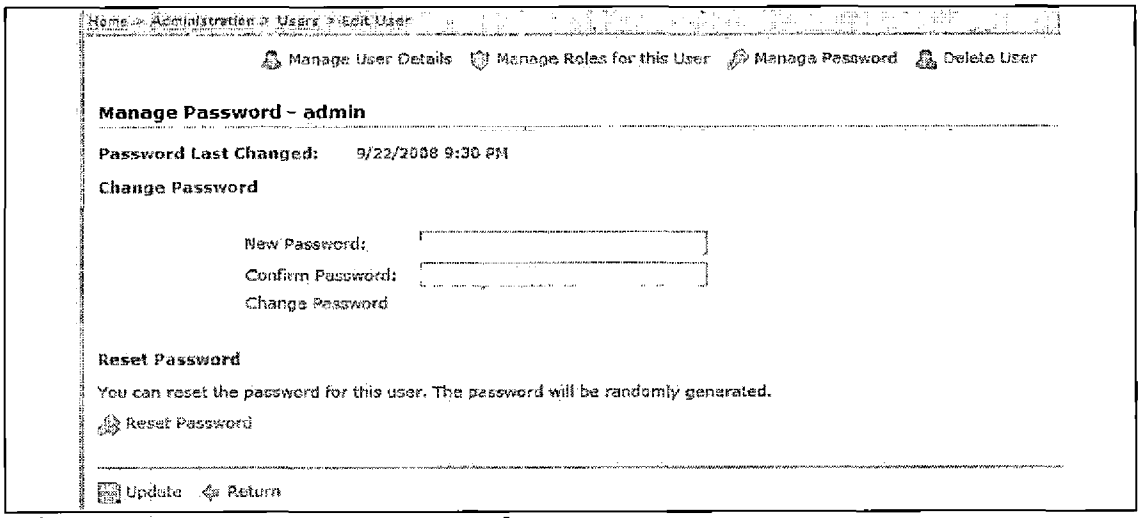


Figure 35. Manage Password Page

## Delete User Page

This page provides the administrator the link to delete the user. It also provides an option to unauthorize the user account. The screen also explains the difference between both the options.

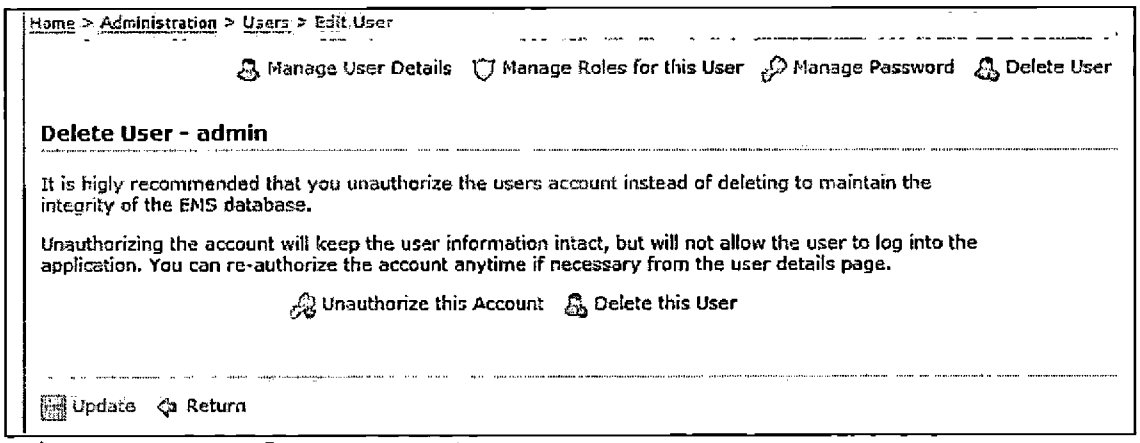


Figure 36. Delete User Page

## Application Configuration Page

This Page is used to set the title and welcome message for the application.

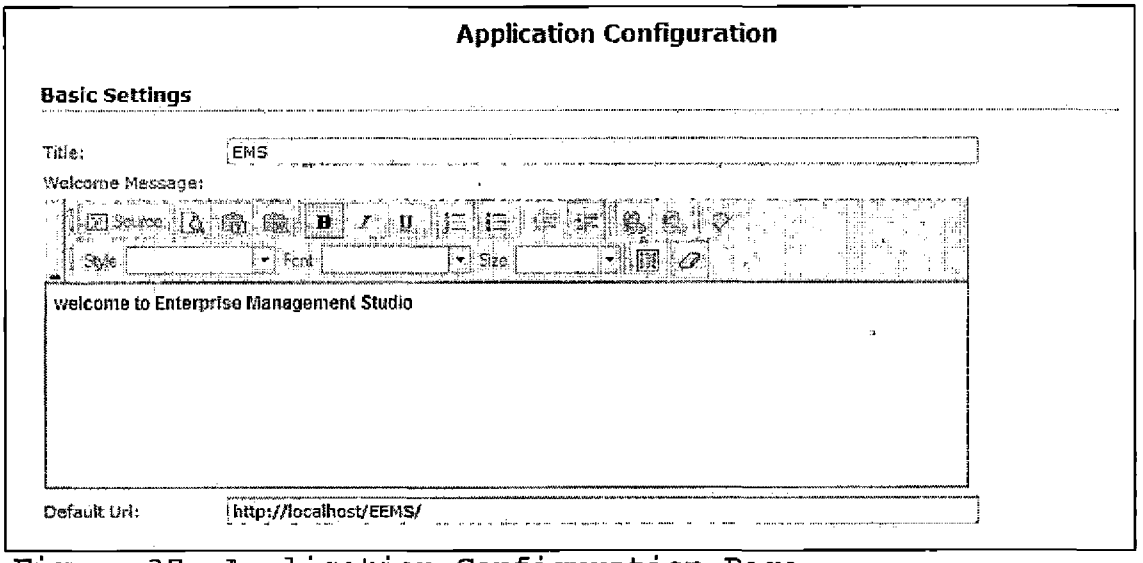


Figure 37. Application Configuration Page

## Application Log Viewer Page

This page gives any exceptions that occurred in the application. It also provides the link "Clear Log" which is used to clear the logs in the page.

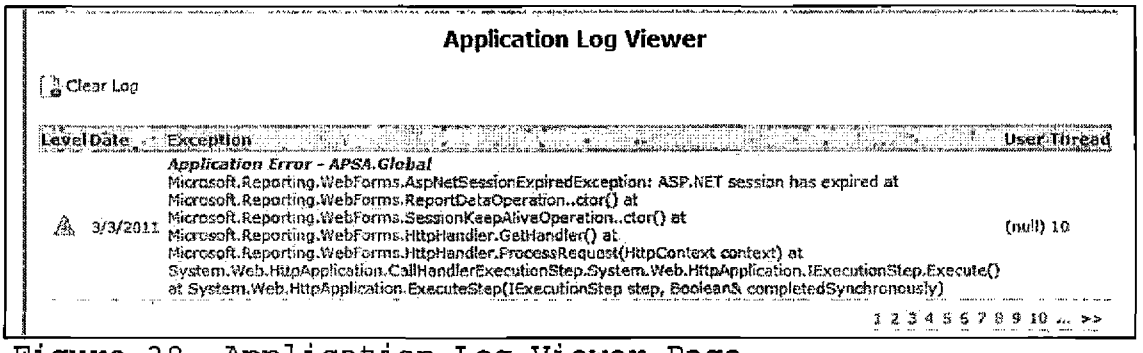


Figure 38. Application Log Viewer Page

Status Report Page

This page is used to view and print the status report. There are two different types of formats that can be used to generate the report. One is the MS Word format and the second one is the Acrobat PDF file format. Also the report can be exported on the desktop using either of the formats.

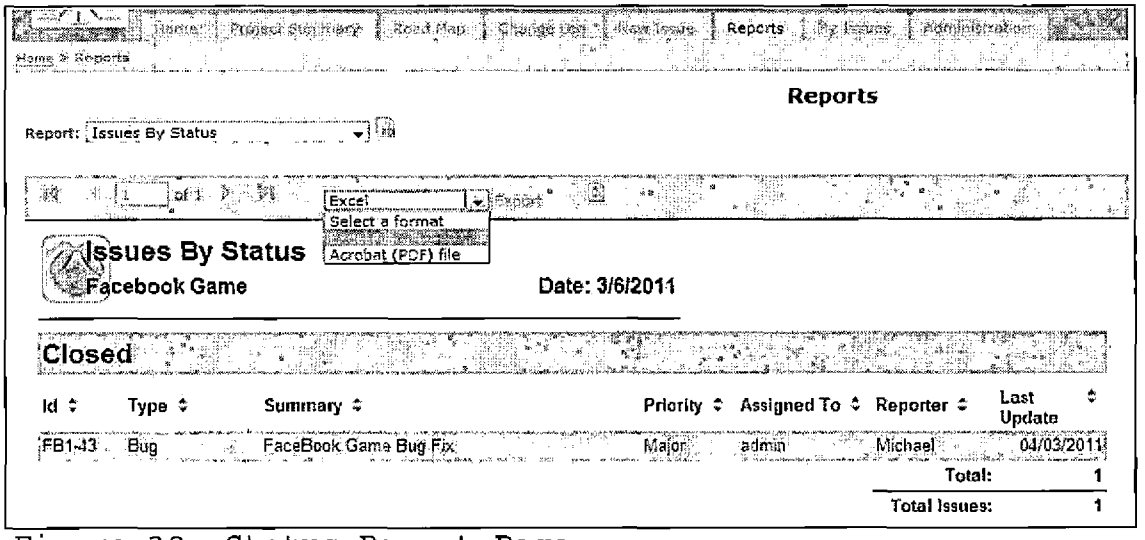


Figure 39. Status Report Page

Issues by Status Chart Page

To view the charts, the user has to select "Issues by Status Chart" from the drop down and click the green arrow next to it. This action will generate the summary chart for "Issues by Status".

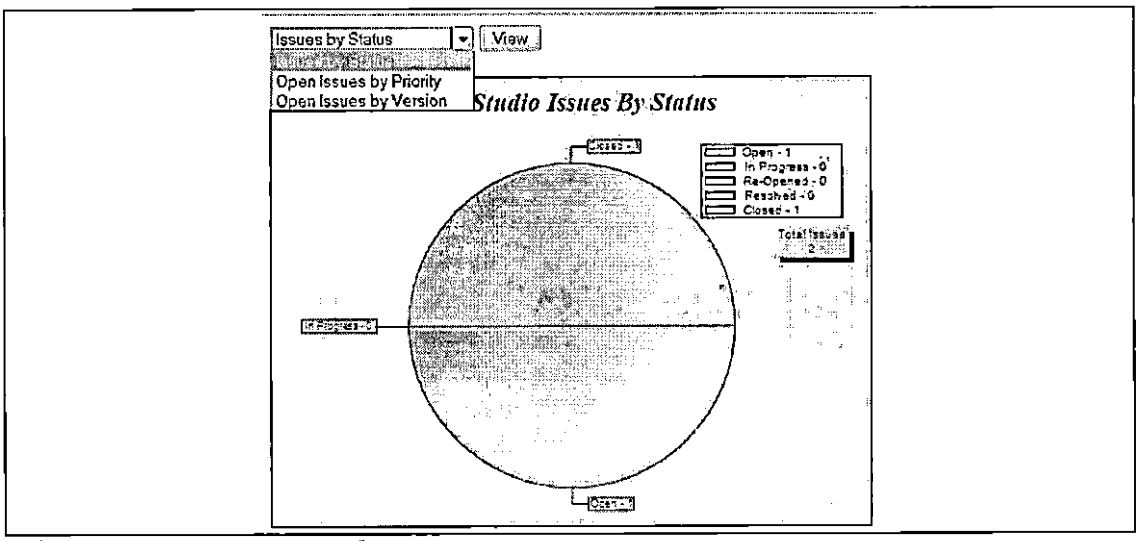

Figure 40. Issues by Status Page

## Open Issues by Priority Chart Page

To view the Priority Page chart with open issues, the user has to select "Open Issues by Priority Page" from the drop down and click the green arrow next to it.

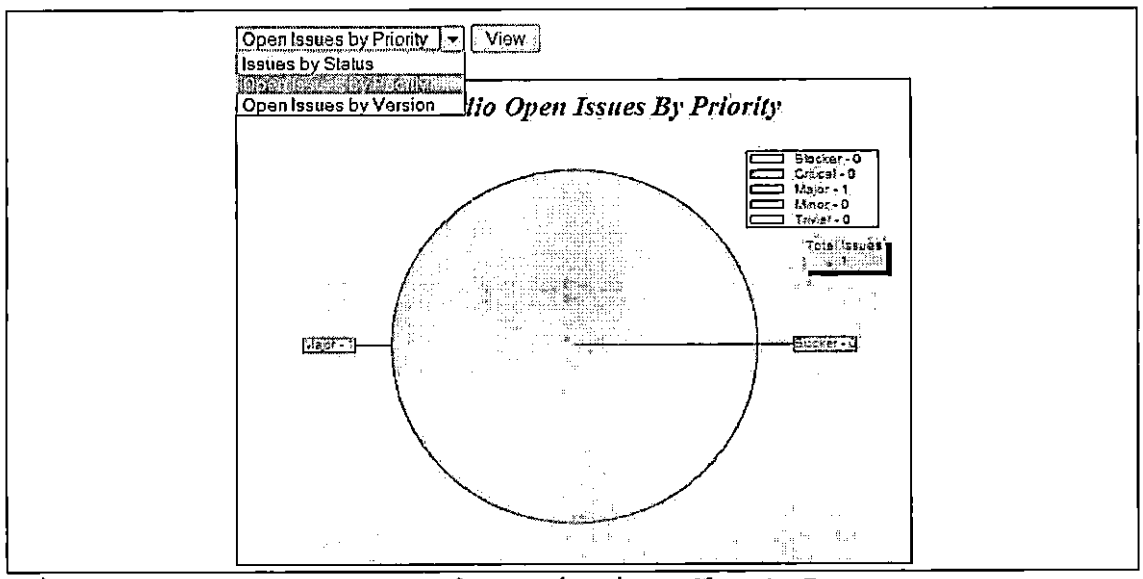

Figure 41. Open Issues by Priority Chart Page

Open Issues by Version Chart Page

To view the Version Page chart with open issues, the user has to select "Open Issues by Version Page" from the drop down and click the green arrow next to it. This action will generate the chart.

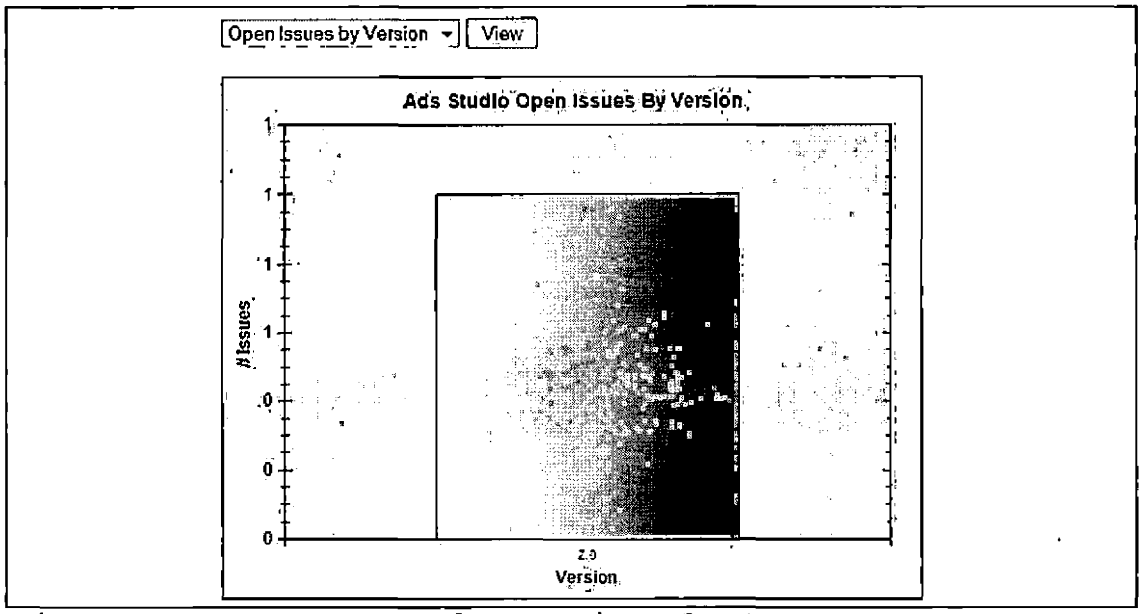


Figure 42. Open Issues by Version Chart Page

Project Time Tracking Report Page

This page generates the report of all hours spent by an employee on the project. To view the report, the user has to select the report from the drop down menu click the green arrow next to the drop down which will generate the report.

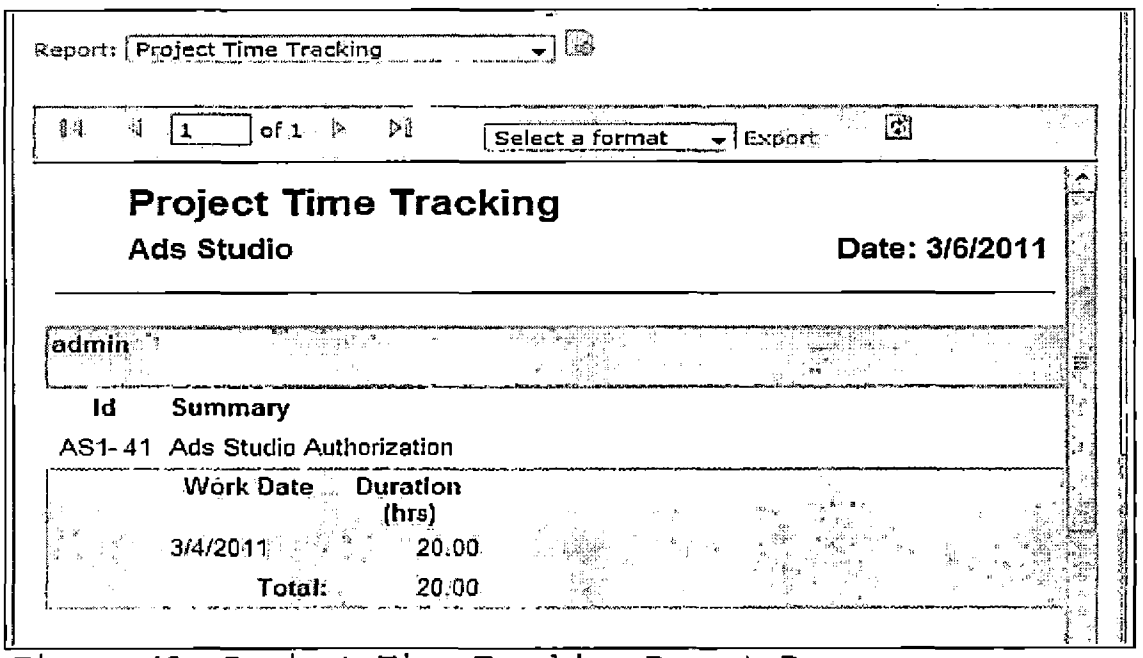


Figure 43. Project Time Tracking Report Page

Project User Time Tracking Report Page

This page generates the report of all hours spent by a particular employee on the project. To view the report, the user has to select the report from the drop down menu. This action will give another drop down menu which will prompt to select the user. The View Report button when clicked will generate the report.

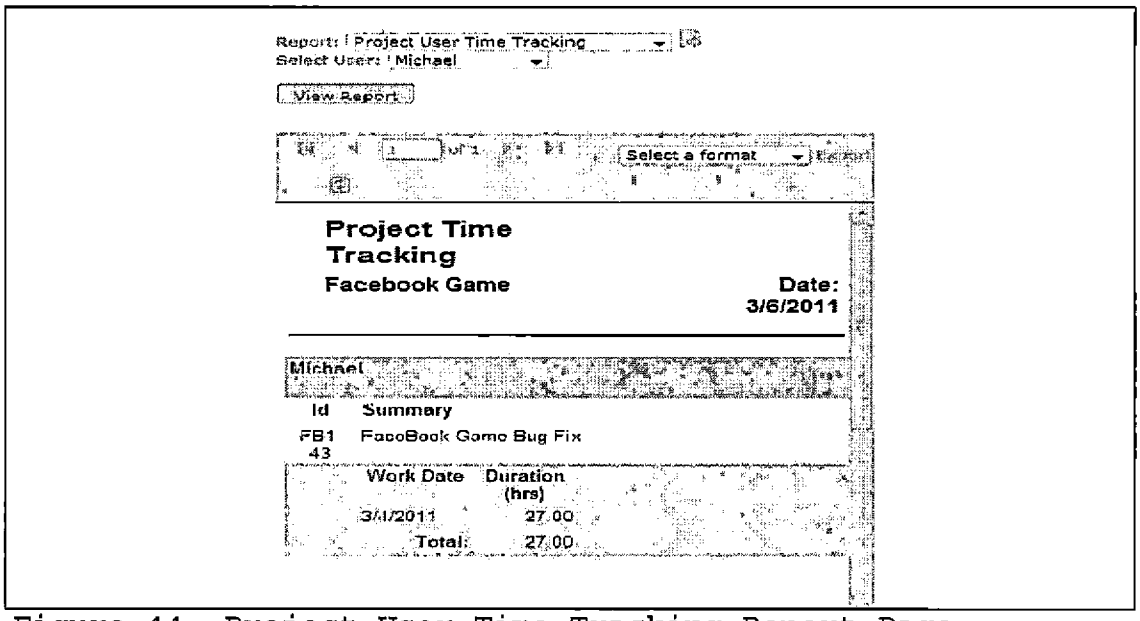


Figure 44. Project User Time Tracking Report Page

## Employee Interface

### After Login Page

This page is visible to the employees after they login to the application. The Administration tab will be disabled for the employees.

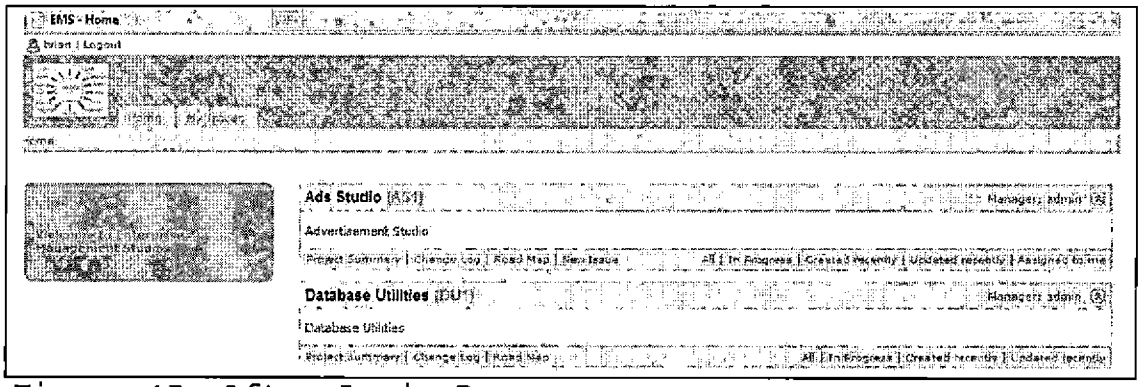


Figure 45. After Login Page

# CHAPTER FIVE

# SYSTEM VALIDATION

This chapter is meant to document the performance of EMS. The purpose of the software validation is to evaluate the attributes of the EMS and determine if it meets its required results. The following sections depict the results in different testing levels.

## Unit Testing

Unit Testing is the basic method of testing where individual units are tested to ensure that they work properly. The results of unit testing in this section are shown in Table 15.

Table 18. Unit Testing Results

| Page | Test Performed | Results |
|---|---|---|
| Login Page | • Check if the login page is redirecting to the proper page when correct authentication is typed.<br>• Redirect to an error page if login information is incorrect.<br>• Check if the forgot password link when clicked takes to forgot password page.<br>• Check the user privileges before loading the page. | Pass |
| Home Page | • Check all the menus are shown properly to the user.<br>• Check all the links work as expected. | Pass |
| Administration Page | • Verify that the appropriate page is loaded depending on the role of the page. | Pass |
| Projects Page | • Check the links work as expected<br>• Verify that projects are displayed<br>• Redirect to an access denied page if the login id does have access rights to create a project. | Pass |
| Create New Project Page | • Check if the wizard is coming up with a series of steps indicating the step number for each wizard. | Pass |
| Wizard Pages | • Check if the links and buttons in the pages are working properly as expected.<br>• Check if the error messages are displayed.<br>• Check Page Validations.<br>• Verify data is saved correctly. | Pass |

| Page | Test Performed | Results |
|---|---|---|
| New Issue Page | • Make sure the links work as expected.<br>• Check drop down menu is populated correctly.<br>• Check if the content in the Welcome Message Textbox is editable with the menu options in the FCK Editor. | Pass |
| My Issues Page | • Check if the issues are displayed properly as expected.<br>• Check if the links and images when<br>• Clicked take to the proper pages. | Pass |
| Issue Details Page | • Check if all the items are displayed in the drop down lists properly as expected.<br>• Check if the links when clicked work properly. | Pass |
| Project Summary Page | • Make sure the links work as expected.<br>• Check if the page displays the data as expected. | Pass |
| Change Log Page | • Make sure the links work as expected.<br>• Check if the page displays the data as expected. | Pass |
| Road Map Page | • Make sure the links work as expected.<br>• Check if the page displays the data as expected. | Pass |
| Manage User Accounts Page. | • Check the links work as expected.<br>• Check the images when clicked work as expected.<br>• Verify that record is correctly saved. | Pass |
| Create New User Page | • Check the Create User link works properly.<br>• Check the information is saved properly in the database. | Pass |

| Page | Test Performed | Results |
|---|---|---|
| Edit User Page | • Check the Update link works correctly.<br>• Check the information is saved properly in the database. | Pass |
| Delete User Page | • Check the links works properly.<br>• Check the information is saved properly in the database. | Pass<br>Pass |
| Manage Password Page | • Check the links works properly.<br>• Check the information is saved properly in the database. | Pass |
| Application Configuration Page | • Check if the content in the Welcome Message Textbox is editable with the menu options in the FCK Editor.<br>• Verify that record is correctly saved. | Pass |
| Log Viewer Page | • Check if the content in the Welcome Message Textbox is editable with the menu options in the FCK Editor.<br>• Verify that record is correctly saved. | Pass |

Table 19. Unit Test Results (Reports)

| Report Pages | Test Performed | Results |
|---|---|---|
| Status Report Page | • Check report is displayed properly as expected.<br>• Check drop down menu is populated correctly.<br>• Check if the report is exported to Excel and PDF formats. | Pass |
| Issues by Status Chart Page | • Check report is displayed properly as expected.<br>• Check drop down menu is populated correctly. | Pass<br>Pass |
| Open Issues by Priority Chart Page | • Check report is displayed properly as expected.<br>• Check drop down menu is populated correctly. | Pass |
| Open Issues by Version Chart Page | • Check report is displayed properly as expected.<br>• Check drop down menu is populated correctly. | Pass |
| Project Time Tracking Report Page | • Check report is displayed properly as expected.<br>• Check drop down menu is populated correctly.<br>• Check if the report is exported to Excel and PDF formats. | Pass |
| Project User Time Tracking Report Page | • Check report is displayed properly as expected.<br>• Check drop down menu is populated correctly.<br>• Check if the report is exported to Excel and PDF formats. | Pass |

## System Testing

System testing is the next step up in the testing process where all related units from a system does a certain task. Thus, the system test process is useful for

detecting interface errors from a front-end perspective and specific functions from the back end point perspective. Table 17 shows system test results in detail.

Table 20. System Testing Results

| System | Tests Performed | Results |
|---|---|---|
| Administrator | • Verify that only authenticated personnel can access Administration tab.<br>• Verify that administrator can create, edit, and delete the project information.<br>• Verify that administrator can add, edit, and delete user accounts. | Pass |
| Authorize system | • Test if error message is displayed on incorrect login.<br>• Make sure the result of the authorizing user is correct.<br>• Verify the login page redirects to the correct browsing page after the user logs in. | Pass |
| User Accounts System | • Make sure all the existing users are listed in the user list.<br>• Check if the user can update his/her own account properly. | Pass |
| Browsing System | • Check if the system checks for user privilege before showing pages.<br>• Verify the page is showing properly after the user clicks on the page link. | Pass |

## System Integration Testing

System Integration testing is the testing process that uses real data, which the system is intended to manipulate, to test the system. This testing is usually done when the system is about to be deployed in a production environment.

Table 21. System Integration Testing Results

| | |
|---|---|
| Installation of the system on IIS | PASS |
| Running of system on Internet Explorer browser | PASS |
| Running of system on Firefox browser | PASS |
| Startup database on SQL server | PASS |
| Real data test on all pages and reports | PASS |

# CHAPTER SIX

# MAINTENANCE MANUAL

Like any good system, Enterprise Management Studio requires regular maintenance. The organization that is going to use it must appoint an administrator who can be made responsible for this. The following section contains instructions for setting up and managing Internet Information server and all necessary applications for running the EMS.

## Installation

### Installing System on Internet Information Services

To install the system on IIS, the following steps must be performed.

1. Make sure that the .NET platform is installed on the computer.
2. To Start the IIS Manager, Select Start menu, Click Control Panel, Click Administrative Tools, Click Internet Information Services Manager.
3. The next step is to create a virtual directory for an existing physical directory. In the IIS Manager window, expand the local computer, and then expand Sites. Right Click Default Site and Click Add Virtual Directory from the list.

4. The Alias is the name a user will use in his/her URL to access the files in this virtual directory. Type EMS in the alias textbox.
5. In the physical path, provide the physical directory path of EMS.
6. Click Connect As and select the path-credentials as Application user (pass-through authentication). Click OK on Connect As window and finally click OK on Add Virtual Directory window. After completing this, the new virtual directory appears in IIS Manager.
7. Test for the login page using either Internet Explorer or Firefox browser.

Installing Microsoft AJAX

The link to download and install the MS Ajax Extensions for .Net Framework 2.0 is http://go.microsoft.com/fwlink/?LinkID=77296

Set up the ASP.NET AJAX Control Toolkit

To make Ajax work on EMS website on the server, download and set up the .NET AJAX Control Toolkit after installing Microsoft AJAX. The link to download the toolkit is: Http://codeplex.com/AjaxControlToolKit

Installing Database on SQL Server 2005

The following steps needs to be performed to create a database on the SQL Server.

1. Install SQL Server 2005 on the system.
2. Configure your SQL Server 2005 installation (Use the manual that comes with the SQL Server 2005 installation).
3. Create a new database in the server using SQL Manager and name it as "Bugnet".
4. Import all the data definitions from the existing database.
5. Open the web.config file and point the data source to the required server.
6. Test the connection using the login page.

Backup

Database Backup

The data must be backed up onto any external device periodically, so that it can be recovered if some data loss event occurs at any point. Follow the instructions given in the SQL manual to back up the data. If required, data can be restored from this location at a later stage.

System Backup

It is important to have a backup copy of the whole system. To do so, the administrator must make a copy of all the ASPX files, DDL and the PDB files, and save them to the external device. In case something happens to the existing system, the administrator can easily restore from the backup.

# CHAPTER SEVEN

# CONCLUSION AND FUTURE DIRECTIONS

## Conclusion

Enterprise Management Studio is an excellent tool for medium to large-scale organizations. It has many features that can be used by the organizations for doing many tasks in less time. Using this tool, the organizations can manage projects and keep better track of them. The interesting feature of EMS is its integration with AJAX. Any new request will be opened in a single page and the entire page does not get refreshed. The ability to assign weights to the milestone information will give the organization better project progress reports.

The performance of the employees can be evaluated using the Time Tracking feature. Since the system can be accessed via Internet, it can be used anytime from any location. Apart from keeping track of project progress, this tool can also be used to find the efficiency of each employee, provide access type for the projects, provide authorization to the users in the company, and also generate various reports for time tracking. The access to EMS database is fast, secure, and easy. Teams can work on a project from remote locations using EMS. Thus, EMS

offers instant and better access to the project information in less time and with less cost, and provides an improved decision making process.

The file upload/download system acts as a one-stop-place to archive and access all the files related to a given project. EMS is easy to use because the GUI is designed in a very user-friendly manner. The usage of EMS will significantly improve the efficiency of the project handling in the organization and decrease the manual efforts. As a conclusion, EMS offers services for instant access to the project information for all the users in the organization-- regardless of their physical location. The focus of the future work is the integration of EMS in a mobile device. The users would be able to receive instant notifications about the projects through mobile devices in the form of alerts.

REFERENCES


[1] R. S. Pressman, *Software Engineering: A Practitioner's Approach*, Sixth Edition, NewYork, NY: McGraw-Hill Companies, Inc., 2005.

[2] S. Maheshwari and R. Jain., *Introduction to SQL and PL/SQL*, India : Laxmi Publications, 2007.

[3] C. Darie and W. Barnett., *Build your own ASP.NET 3.5 Using C# & VB*, Collingwood, Australia : SitePoint, 2008.

[4] M. MacDonald and M. Szpuszta., *Pro ASP.NET 3.5 in C# 2008*, Second Edition, NewYork, NY: Apress, 2007.

[5] A. Demelo., REMOTE: A Complete Software Process Management Tool, Masters in computer Science Project, CSUSB, 2006.

[6] A. Dhir, Online Project Management System, Masters in computer Science Project, CSUSB, 2005.

[7] .NET Development, MSDN Library. http://msdn.microsoft .com/en-us/library/default.aspx/

[8] ASP.NET AJAX Control Toolkit, The Official Microsoft ASP.NET Site. http://www.asp.net/

[9] CKEditor. http://en.wikipedia.org/wiki/CKEditor.

[10] Wikipedia, http://www.wikipedia.org/