

BUDGET CONTROLLING SYSTEM

A Mini Project Report Submitted to the Faculty of

Computer Science and Engineering

Geethanjali College of Engineering & Technology

(Affiliated to J.N.T.U.H, Approved by AICTE, NEW DELHI)



In partial fulfillment of the requirements

For the award of degree of

BACHELOR OF TECHNOLOGY

IN

COMPUTER SCIENCE AND ENGINEERING

By

KOMPELLA KAUSHIK (10R11A0527)

N.CHARAN DAS (10R11A0539)

V. PRASHANTH (10R11A0556)

Under the Esteemed Guidance of

A.BIXAPATHI

Assistant Professor

Department of Computer Science and Engineering

Geethanjali College of Engineering & Technology

Cheeryal (V), Keesara (M), R.R. Dist. Hyderabad-501301, A.P

2013-2014

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

Geethanjali College of Engineering & Technology

(Affiliated to J.N.T.U.H, Approved by AICTE, NEWDELHI.)



CERTIFICATE

This is to certify that the Mini Project report entitled “BUDGET CONTROLLING SYSTEM” is a bonafide work done by Kompella Kaushik (10R11A0527), N.Charandas (10R11A0539), V.Prashanth (10R11A0556), in partial fulfillment of the requirement of the award for the degree of Bachelor of Technology in “Computer Science and Engineering” from Jawaharlal Nehru Technological University, Hyderabad during the year 2013-2014.

Internal Guide

A. Bixapathi

Assistant Professor,

Dept of C.S.E

H.O.D

Dr. D.Ratna Deepthi

Professor& Head,

Dept of C.S.E

External Examiner

ACKNOWLEDGEMENT

We are greatly indebted to the authorities of Geethanjali College of Engineering and Technology, Cheeryal, R.R Dist, for providing us the necessary facilities to successfully carry out this mini project work titled “BUDGET CONTROLLING SYSTEM”.

Firstly, we thank and express our sincere gratitude to Prof. Dr. D.RATNADEEPTHI, HOD, CSE department, Geethanjali College of Engineering and Technology, for his invaluable help and support which helped us a lot in successfully completing our mini project.

Secondly, we express our gratitude to Assistant Prof. Mr. A. BIXAPATHI, Internal guide, Geethanjali College of Engineering and Technology for his suggestions and encouragement which helped us in the successful completion of our mini project.

We would like to express our sincere gratitude to our Principal Dr. S.UDAYA KUMAR for providing the necessary infrastructure to complete our project.

Finally, we would like to express our heartfelt thanks to our parents who were very supportive both financially and mentally and for their encouragement to achieve our set goals.

Kompella Kaushik (10R11A0527)

N. Charan Das (10R11A0539)

V. Prashanth (10R11A0556)

ABSTRACT

This project is aimed at developing a system by which the employees in the organization submit the bills to their managers. The bills could of various types and also of various amounts. The employee after submitting the bill will automatically provide the manager's name to which the bill will be submitted. The bill will pass through a workflow process and the owner of the bill can view the status of the bill at any time. An action on bills will be Store in to the concerned table, let them know about the status of the bill.

LIST OF FIGURES/DIAGRAMS

S.No.	Figure	Description	Page No.
1	Figure 4.1.1	System Architecture Business Flow Diagram	11
2	Figure 4.1.2	System Architecture Technical Flow Diagram	11
3	Figure 4.2.1	Use case Diagram	13
4	Figure 4.2.2	Sequence Diagram	14
5	Figure 4.2.3	Activity Diagram	15
6	Figure 4.2.4	Class Diagram	16
7	Figure 4.2.5	Component Diagram	17
8	Figure 7.1	Login Page	26
9	Figure 7.2	Registration Page	27
10	Figure 7.3	New Expense Page	28
11	Figure 7.4	Budget Amount Page	29
12	Figure 7.5	Action on Expense Page	30

LIST OF TABLES

S.No.	Table	Description	Page No.
1	Table 2.1	Software Requirements	4
2	Table 2.2	Hardware Requirements	5
3	Table 6.1	Test Cases	25

TABLE OF CONTENTS

S. No.	Contents	Page No.
1.	Introduction.....	1
	1.1 About the Project.....	1
	1.2 Objective.....	1
2.	System Analysis.....	2
	2.1 Existing System.....	2
	2.2 Proposed System.....	2
	2.3 Scope of the Project.....	2
	2.4 Modules Description.....	3
	2.4.1 Admin.....	3
	2.4.2 Manager.....	3
	2.4.3 Employee.....	3
	2.5 System Configuration.....	4
	2.5.1 Software Requirements.....	4
	2.5.2 Hardware Requirements.....	5
	2.5.3 Feasibility Study.....	5
	2.5.3.1 Economic Feasibility.....	5
	2.5.3.2 Technical Feasibility.....	6
	2.5.3.3 Operational Feasibility.....	6

3.	Literature Overview.....	7
3.1	Technology Used.....	7
3.2	Java Tools.....	8
3.2.1	JSP.....	8
3.2.2	JDBC.....	8
3.2.2.1	Single-Tier Architecture.....	8
3.2.2.2	Two-Tier Architecture.....	9
3.2.2.3	Three-Tier and N-Tier Architecture.....	9
3.2.3	J2EE.....	9
3.2.4	SERVLETS.....	9
3.2.5	HTML.....	10
3.2.6	XML.....	10
4.	System Design.....	11
4.1	System Architecture.....	11
4.1.1	Business Flow.....	11
4.1.2	Technical Flow.....	11
4.2	UML Diagrams.....	12
4.2.1	Use Case Diagram.....	13
4.2.2	Sequence Diagram.....	14
4.2.3	Activity Diagram.....	15
4.2.4	Class Diagram.....	16
4.2.5	Component Diagram.....	17
4.3	Database Design.....	17
5.	Sample Code.....	18
5.1	Coding.....	18
5.1.1	Object Oriented Programming and Java.....	18
5.1.2	The Object Oriented Approach.....	18
5.1.3	Characteristics of Object-Oriented Languages.....	18
5.1.3.1	Objects.....	18

	5.1.3.2 Abstraction.....	19
	5.1.3.3 Encapsulation.....	19
	5.1.3.4 Inheritance.....	19
	5.1.3.5 Polymorphism.....	20
	5.1.4 Sample Code.....	20
6.	Testing.....	22
	6.1 Testing.....	22
	6.1.1 White Box Testing.....	22
	6.1.2 Black Box Testing.....	22
	6.2 Types of Testing.....	23
	6.2.1 Unit Testing.....	23
	6.2.2 Integration Testing.....	23
	6.2.3 Validation Testing.....	23
	6.2.4 System Testing.....	23
	6.2.5 Functional Testing.....	24
	6.2.6 Performance Testing.....	24
	6.2.7 Regression Testing.....	24
	6.2.8 Alpha Testing.....	24
	6.2.9 Beta Testing.....	24
	6.3 Test Cases.....	25
7.	Output Screens.....	26
	7.1 Login Page.....	26
	7.2 Registration Page.....	27
	7.3 New Expense Page.....	28
	7.4 Budget Amount Page.....	29
	7.5 Action on Expense Page.....	30
8.	Conclusion.....	31
	8.1 Conclusion.....	31
	8.2 Further Enhancements.....	31

9.	Bibliography.....	32
	9.1 Books References.....	32
	9.2 Websites References.....	32
	9.3 Technical Publication References.....	32

1. INTRODUCTION

1.1 ABOUT THE PROJECT

This project is aimed at developing a system by which the employees in the organization submit the bills to their managers. The bills could of various types and also of various amounts. The employee after submitting the bill will automatically provide the manager's name to which the bill will be submitted. The bill will pass through a workflow process and the owner of the bill can view the status of the bill at any time. An action on bills will be Store in to the concerned table, let them know about the status of the bill.

1.2 OBJECTIVE

A budgetary controlling system is a method of monitoring and controlling income and expenditure, for managing the demands for cash and minimizing borrowings. It can be applied in a business context or by an individual in relation to his or her personal finances. In a business environment it is most valuable as a tool to control the flow of cash. Additionally, such a system would also ensure that cash will always be available for essential business purposes like buying raw materials.

2. SYSTEM ANALYSIS

2.1 EXISTING SYSTEM

In the existing system, Companies currently maintain all Account Transactions and Budget Information manually. To overcome the limitations of manual system, computerized Account and Budget Control System has been introduced, which will solve the problems of the existing system.

2.2 PROPOSED SYSTEM

The information of the allocated budget amount and the expenses made by the company on budgeted account will be readily available. This feature will allow Company to monitor the expenses of its branches under different budget/account head. One of the strongest benefits of this Budget Controlling System over current manual system is the generation of most up-to-date trial balance at any time. The generation of trial balance in current system takes considerable time and effort. This application can be used by the server from any media like computer, mobile, touch pad etc. User can get the report in different chart formats like pie chart, bar chart etc. User can set the alert facility if the expenditure is meeting his stop value.

2.3 SCOPE OF THE PROJECT

This project is aimed at developing a system by which the employees in the organization submit the bills to their managers. It can be accessed easily. Updates are easy, it is platform independent, bud fixings are easy, it is easy to use and security is also maintained. One of the strongest benefits of this Budget Controlling System over current manual system is the generation of most up-to-date trial balance at any time. The generation of trial balance in current system takes considerable time and effort. This application can be used by the server from any media like computer, mobile, touch pad etc.

2.4 MODULES DESCRIPTION

Based on the Budget Controlling System project has been analyzed and divided into three main modules they are follows:

1. Admin Module
2. Manager Module
3. Employee Module

2.4.1 ADMIN

The Admin can login with his User-id and Password. He can create a new employee. Admin can create new Expense and submit it to the manager. He can able to view to all actions taken by the Finance manager on Expenses. He can able to view his reports.

2.4.2 MANAGER

He has a separate User-id and Password with which he can maintain the transactions of the manager. Manager will decide the annual budget amount which is displayed on Home Page of Employee and Admin. Manager will take the decision on all types of expenses whether they may Approve, Hold or Reject. Manager also views reports.

2.4.3 EMPLOYEE

He is also having a separate username with which he can maintain the transactions of the employees. Employee can enter an expense and submit to the manager. Employee can view the reports to know the actions taken by the manager on expenses.

2.5 SYSTEM CONFIGURATION

The system configuration defines how the system should be configured in order to run the project being developed. The system should be properly configured in order to avoid nasty resource conflict problems and other strange errors. System Configuration includes the software requirements of the project, the hardware requirements of the project and the feasibility study. The feasibility study shows how well the project can work under limited resources and limited time.

2.5.1 SOFTWARE REQUIREMENTS

The Software requirements enlist all the requirements and the environment that are necessary for the project development. The software requirements primarily include the front end and the back end of the project. The front end application is the one with which the users interact directly. The back end application serves indirectly in support of the front end services by communicating with the required resource. The intermediate program mediates front end and back end activities.

Below is a list of Software Requirements for the development of “BUDGET CONTROLLING SYSTEM”.

Java	Version 1.6
Technologies	Java, Jdbc, servlets, jsp
Web technologies	Html, CSS, JavaScript,Ajax
Browser	Mozilla Firefox
Web application server	Tomcat 6.0
Database	MySQL 5.1

Table 2.1: Software requirements

2.5.2 HARDWARE REQUIREMENTS

The hardware requirements represent all the usage requirements for the project. This shows the minimum and recommended requirements in the hardware of the system to run the project efficiently.

Below is a list of Hardware Requirements for the development of “BUDGET CONTROLLING SYSTEM”.

Hard Disk	20GB
Ram	128MB
Processor	Intel Processor IV

Table 2.2: Hardware requirements

2.5.3 FEASIBILITY STUDY

A feasibility study evaluates the project's potential for success. All projects are feasible under unlimited resources and infinite time. But the development of software is plagued by the scarcity of resources and difficult delivery rates. It is both necessary and prudent to evaluate the feasibility of a project at the earliest possible time. Three key considerations are involved in the feasibility analysis.

2.5.3.1 ECONOMIC FEASIBILITY

This procedure is to determine the benefits and savings that are expected from a candidate system and compare them with costs. If benefits outweigh costs, then the decision is made to design and implement the system. Otherwise, further justification or alterations in proposed system will have to be made if it is to have a chance of being approved. This is an ongoing effort that improves in accuracy at each phase of the system life cycle.

2.5.3.2 TECHNICAL FEASIBILITY

A large part of determining resources has to do with assessing technical feasibility. It considers the technical requirements of the proposed project. The technical requirements are then compared to the technical capability. The project is considered technically feasible if the internal technical capability is sufficient to support the project requirements. Technical feasibility centers on the existing computer system (hardware, software, etc.,) and to what extent it can support the proposed addition. If the budget is a serious constraint, then the project is judged not feasible.

2.5.3.3 OPERATIONAL FEASIBILITY

Operational feasibility is dependent on human resources available for the project and involves projecting whether the system will be used if it is developed and implemented. Operational feasibility is a measure of how well a proposed system solves the problems, and takes advantage of the opportunities identified during scope definition and how it satisfies the requirements identified in the requirements analysis phase of system development.

3. LITERATURE OVERVIEW

3.1 TECHNOLOGY USED

The technology used in this system is **JAVA**.

Java was conceived by James Gosling, Patrick Naughton, Chris Warth, Ed Frank and Mike Sheridan at SUN Microsystems Incorporation in the year 1991. It took 18 months to develop the 1st working version. This language was initially called “OAK”, but was renamed “JAVA” in 1995, many more contributed to the design and evolution of the language.

Java is a powerful but lean object-oriented programming language. It has generated a lot of excitement because it makes it possible to program for Internet by creating Applets, programs that can be embedded in web page. The context of an applet can be an animation with sound, an interactive game or a ticker tape. Applets can be just little decorations to liven up web page, or they can be serious applications like Word processor or Spreadsheet.

Java is being used more for writing standalone applications as well. It is becoming so popular that many people believe it will become standard language for both general purpose and Internet programming.

Java builds on the strength of C++. It has taken the best features of C++ and discarded the more problematic and error prone parts. To this lean core, it has added garbage collection (automatic memory management), multithreading (the capacity for one program to do more than one thing at a time), security capabilities. This result is that Java is simple, elegant, and powerful and easy-to-use. This system uses various JAVA tools.

3.2 JAVA TOOLS

Given below are the various Java Tools used for the development of the “BUDGET CONTROLLING SYSTEM”.

3.2.1 JSP

Java Server Pages or JSP for short is Sun's solution for developing dynamic web sites. JSP provides excellent server side scripting support for creating database driven web applications. JSP enables the developers to directly insert java code into jsp file, this makes the development process very simple and its maintenance also becomes very easy. JSP pages are efficient, it loads into the web server's memory on receiving the request very first time and the subsequent calls are served within a very short period of time. JavaServer Pages are text files that combine standard HTML and new scripting tags. JSPs look like HTML, but they get compiled into Java servlets the first time they are invoked. The resulting servlet is a combination of HTML from the JSP file and embedded dynamic content specified by the new tags.

3.2.2 JDBC

JDBC (Java Database connectivity) is a front-end tool for connecting to a server to ODBC in that respect, however JDBC can connect only java client and it uses ODBC for the connectivity. JDBC is essentially a low-level API since any data manipulation, storage and retrieval has to be done by the program itself. Based on number of intermediate server through the request should go it is named as single tier, two tier and multi tier architecture

3.2.2.1 SINGLE TIER ARCHITECTURE

In a single tier the server and client are the same, in the sense that a client program that needs information (client) and the source of this type of architecture is also possible in java, in case flat files are used to store the data. However this is useful only in case of

small applications. The advantage with this is the simplicity and portability of the application developed.

3.2.2.2 TWO TIER ARCHITECTURE

In two tier architecture the database resides in one machine the network. In this type of architecture a database management takes control of the database and provides access to clients in a network. This software bundle is also called as the server. Software in different machines, requesting for information are called as the clients.

3.2.2.3 THREE TIER AND N-TIER ARCHITECTURE

In the three-tier architecture, any number servers can access the database that resides on server, which in turn serve clients in a network. The intermediate server acts as a two-way communication channel also. This is the information or data from the database is passed on to the applet that is requesting it. This can be extended to make n tiers of servers, each server carrying to specific type of request from clients; however in practice only 3 tiers architecture is popular.

The four types of JDBC Drivers are, JDBC-ODBC BRIDGE PLUS ODBC DRIVER, NATIVE API PARTLY-JAVA DRIVER, JDBC-NET ALL-JAVA DRIVER, and NATIVE PROTOCOL ALL-JAVA DRIVER.

3.2.3 J2EE

The platform provides an API and runtime environment for developing and running enterprise software, including network and web services, and other large-scale, multi-tiered, scalable, reliable, and secure network applications. The platform incorporates a design based largely on modular components running on an application server.

3.2.4 SERVELETS

Servlets provide a Java-Based solution used to address the problems currently associated with doing server side programming, including inextensible scripting

solutions, platform specific APIs, and incomplete interfaces. Servlets are objects that conform to a specific interface which can be plugged into a Java-based server. Servlets are to the server-side where as applets are client-side-object byte codes that can be dynamically loaded off the net. They differ from applets in that they are faceless objects (without graphics or a GUI component). They serve as platform independent, dynamically loadable, plug gable helper byte code objects on the server side that can be used to dynamically extend server-side functionality.

3.2.5 HTML

The hypertext markup language (HTML) is a simple markup language used to create hypertext documents that are portable from one platform to another. HTML documents are SGML (Standard generalized markup language) documents with generic semantics that are appropriate for representing information from a wide range of applications. An HTML document consists of text, which comprises the content of the document and tags, which, defines the structure, and appearance of the document. Each document has a head and body delimited by the <HEAD> and <BODY> tag. The head is where we give our HTML document a title and where we indicate other parameters the browser may use when displaying the document.

3.2.6 XML

Extensible Markup Language (XML) is a markup language that defines a set of rules for encoding documents in a format that is both human-readable and machine-readable. The design goals of XML emphasize simplicity, generality, and usability over the Internet. It is a textual data format with strong support via Unicode for the languages of the world. Although the design of XML focuses on documents, it is widely used for the representation of arbitrary data structures, for example in web services.

4. SYSTEM DESIGN

4.1 SYSTEM ARCHITECTURE

CONVENTIONS STANDARDS FOLLOWED

Security- Username and Password are required to access the system.

Quality- By keeping the interface simple and direct, quality is maintained high.

The following are the different ways of representing System Architecture.

4.1.1 BUSINESS FLOW

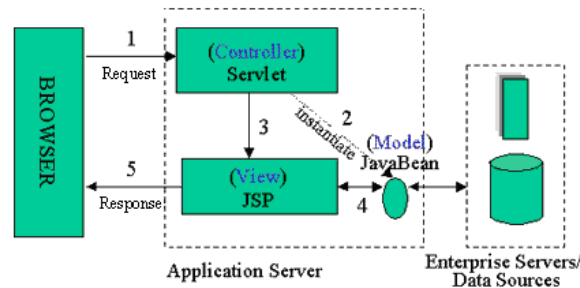


Figure 4.1.1: System Architecture Business Flow Diagram

4.1.2 TECHNICAL FLOW

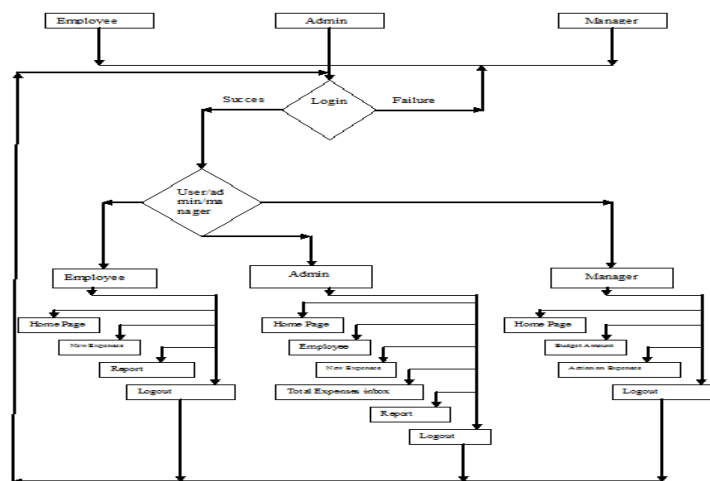


Figure 4.1.2: System Architecture Technical Flow Diagram

4.2 UML DIAGRAMS

The Unified Modeling Language (UML) is a standard visual modeling language intended to be used for modeling business and similar processes and to analysis, design, and implement a software-based systems.

Unified Modeling Language (UML) combines techniques from data modeling (entity relationship diagrams), business modeling (work flows), object modeling, and component modeling. It can be used with all processes, throughout the software development life cycle, and across different implementation technologies.

The Unified Modeling Language (UML) offers a standard way to visualize a system's architectural blueprints, including elements such as:

1. Activities
2. Actors
3. Business processes
4. Database schemas
5. (Logical) components
6. Programming language statements
7. Reusable software components

UML is a common language for business analysts, software architects and developers used to describe, specify, design, and document existing or new business processes, structure and behavior of artifacts of software systems.

UML can be applied to diverse application domains (e.g., banking, finance, internet, etc.) It can be used with all major object and component software development methods and for various implementation platforms (e.g., J2EE,). The various UML diagrams are listed below.

4.2.1 USECASE DIAGRAM

A Use Case Model describes the proposed functionality of a new system. A Use Case represents a discrete unit of interaction between a user (human or machine) and the system. This interaction is a single unit of meaningful work, such as Create Account or View Account Details. Each Use Case describes the functionality to be built in the proposed system, which can include another Use Case's functionality or extend another Use Case with its own behavior.

The USECASE diagram for BUDGET CONTROLLING SYSTEM is given as

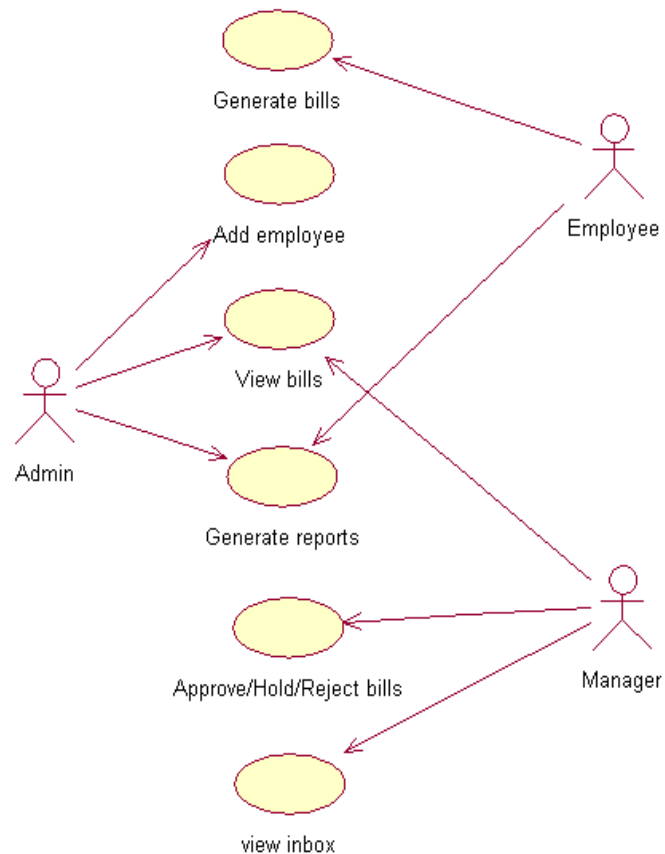


Figure 4.2.1 Use Case Diagram

4.2.2 SEQUENCE DIAGRAM

Sequence diagrams provide a graphical representation of object interactions over time. These typically show a user or actor, and the objects and components they interact with in the execution of a use case. One sequence diagram typically represents a single Use Case 'scenario' or flow of events. Sequence diagrams are an excellent way of documenting usage scenarios and both capturing required objects early in analysis and verifying object use later in design. The diagrams show the flow of messages from one object to another, and as such correspond to the methods and events supported by a class/object.

The SEQUENCE diagram for BUDGET CONTROLLING SYSTEM is given as

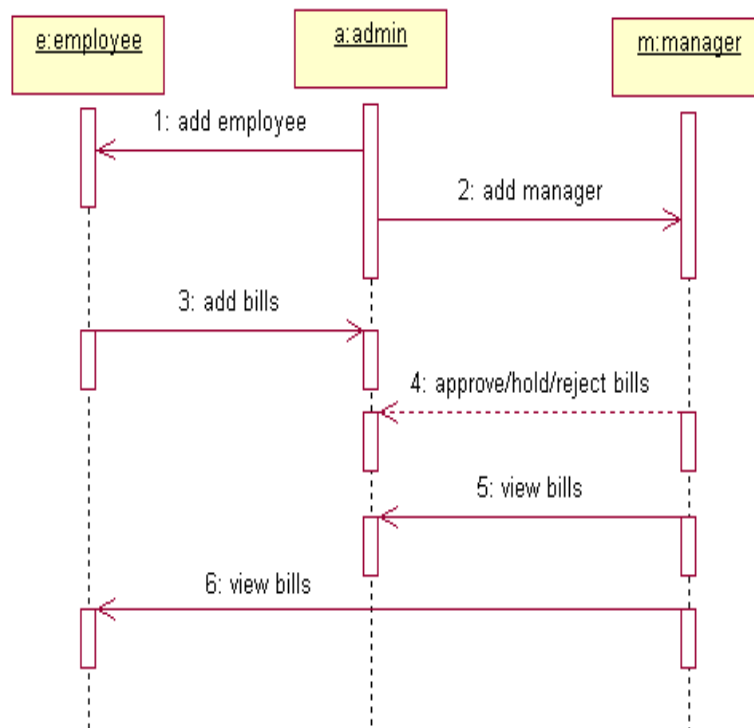


Figure 4.2.2: Sequence Diagram

4.2.3 ACTIVITY DIAGRAM

Activity diagram is another important diagram in UML to describe dynamic aspects of the system. Activity diagram is basically a flow chart to represent the flow from one activity to another activity. The activity can be described as an operation of the system. So the control flow is drawn from one operation to another. This flow can be sequential, branched or concurrent. Activity diagrams deal with all types of flow control by using different elements like fork, join etc.

The ACTIVITY DIAGRAM for the BUDGET CONTROLLING SYSTEM is given as

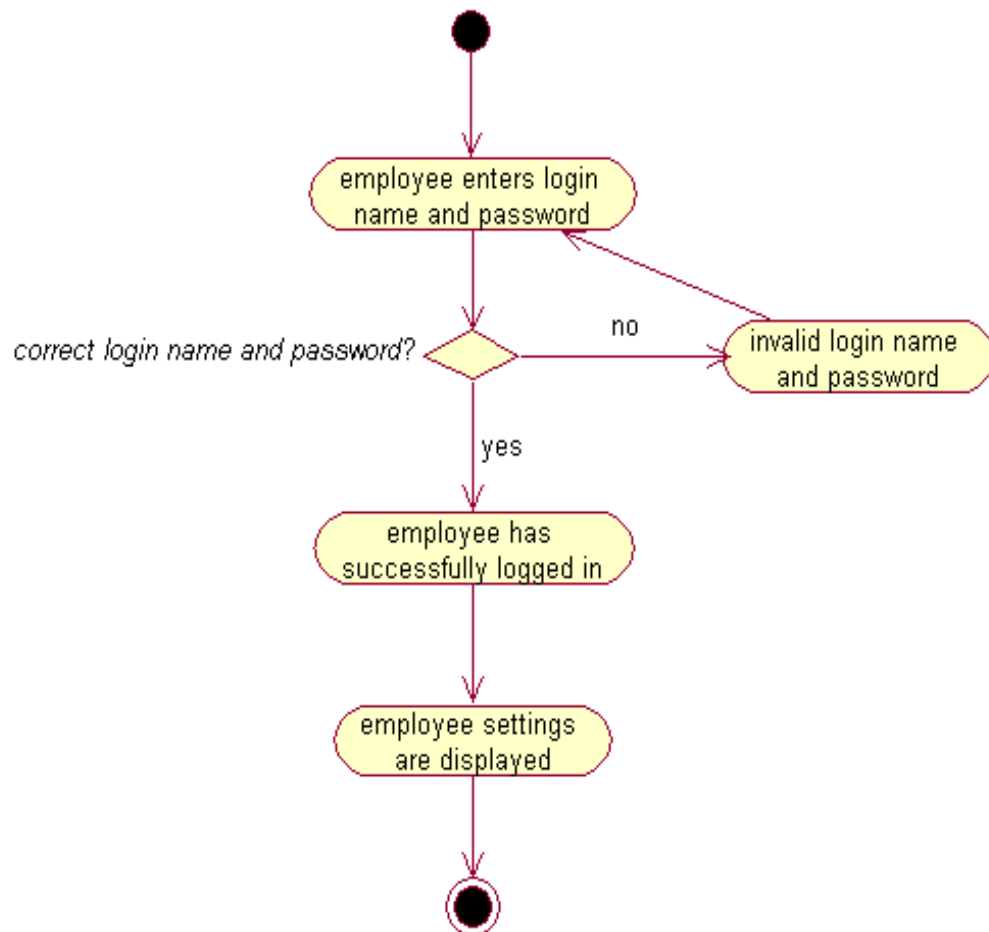


Figure 4.2.3: Activity Diagram

4.2.4 CLASS DIAGRAM

The class diagram is a static diagram. It represents the static view of an application. The class diagram describes the attributes and operations of a class and also the constraints imposed on the system. The class diagram shows a collection of classes, interfaces, associations, collaborations and constraints. It is also known as a structural diagram. The purpose of the class diagram is to Analyze and design the static view of an application, Describe responsibilities of a system; provide base for component and deployment diagrams, forward and reverse engineering.

The CLASS DIAGRAM of the BUDGET CONTROLLING SYSTEM is given as

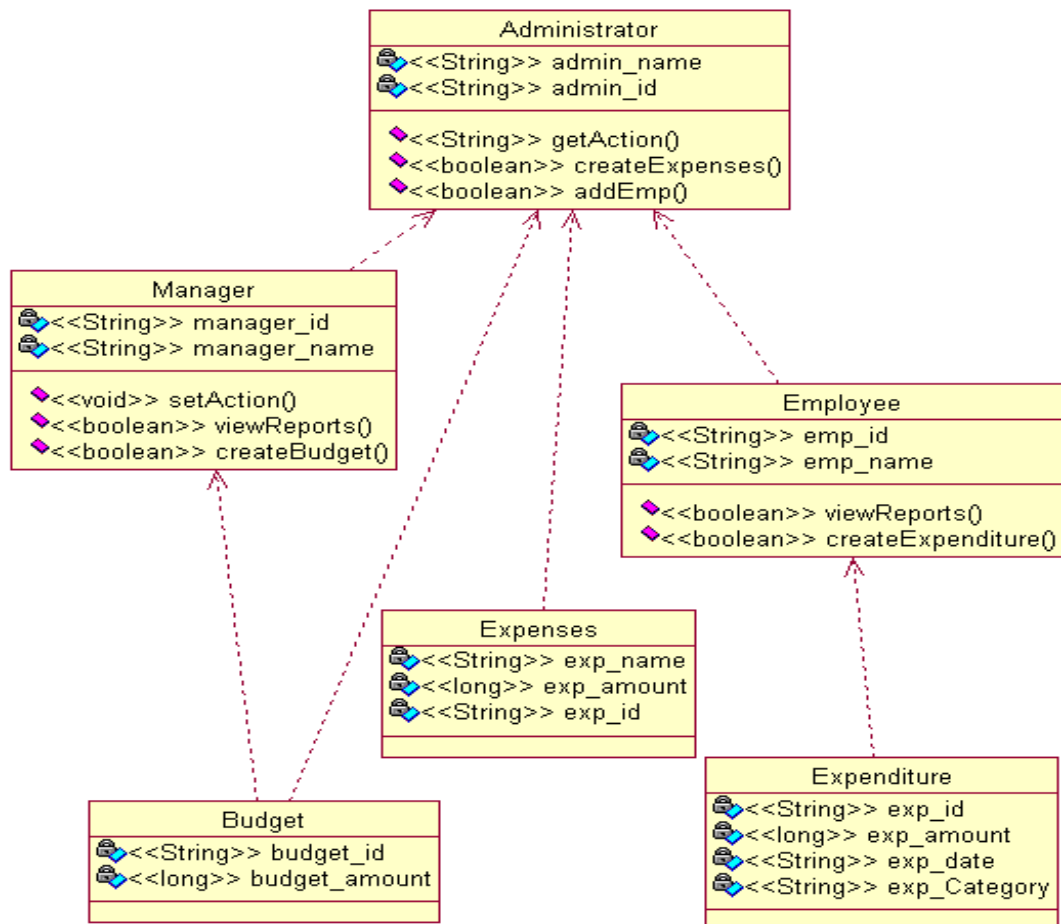


Figure 4.2.4: Class Diagram

4.2.5 COMPONENT DIAGRAM

Component diagrams are different in terms of nature and behavior. They are used to model physical aspects of a system. The purpose of the component diagram is to visualize the components of a system, Construct executables by using forward and reverse engineering, Describe the organization and relationships of the components.

The COMPONENT DIAGRAM of the BUDGET CONTROLLING SYSTEM is given as

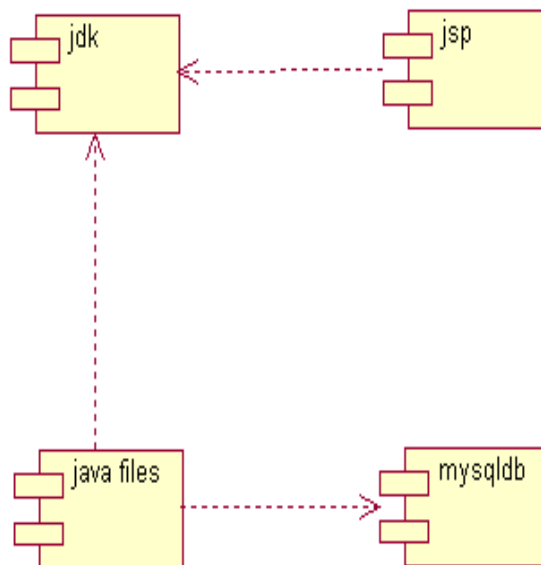


Figure 4.2.5: Component Diagram

4.3 DATABASE DESIGN

In the BUDGET CONTROLLING SYSTEM, MySQL is used to store the data. MySQL is a database system used on the web. Basically, a MySQL database allows us to create a relational database structure on a web-server somewhere in order to store data or automate procedures.

5. SAMPLE CODE

5.1 CODING

5.1.1 OBJECT ORIENTED PROGRAMMING AND JAVA

Object-oriented Programming was developed because of limitations found in earlier approaches of programming. To appreciate what OOP does, we need to understand what these limitations are and how they arose from traditional programming.

5.1.2 THE OBJECT ORIENTED APPROACH

The fundamental idea behind object-oriented languages is to combine into a single unit both data and the functions that operate on that data. Such a unit is called an object.

An object's functions, called member methods in Java, typically provide the only way to access its data. If you want to read the item and return the value to you, you call a member function in the object. It will read the item and return the value to you. You can't access the data directly. The data is hidden, so it is safe from accidental modification. Data and its functions are said to be encapsulated into a single entity. Data encapsulation and data hiding are key terms in the description of object oriented languages. No other functions can access the data. This simplifies writing, debugging, and maintaining the program. A Java program typically consists of a number of objects, which communicate with each other by calling one another's members functions.

5.1.3 CHARACTERISTICS OF OBJECT-ORIENTED LANGUAGES

5.1.3.1 OBJECTS

When you approach a programming problem in an object oriented language, you no longer ask how the problem will be divided into functions, but how it will be divided into objects. Thinking in terms of objects, rather than functions, has a surprisingly helpful effect on how easily programs can be designed and objects in the real world.

5.1.3.2 ABSTRACTION

An essential element of object-oriented programming is abstraction. Humans manage complexity through abstraction. For example, people do not think of a car as a set of tens of thousands of individual parts. They think of it as a well-defined object with its own unique behavior. This abstraction allows people to use a car to drive to the grocery store without being overwhelmed by the complexity of the parts that form the car. They can ignore the details of how the engine, transmission, and braking systems work. Instead they are free to utilize the object as a whole.

5.1.3.3 ENCAPSULATION

Encapsulation is the mechanism that binds together code and the data it manipulates, and keeps both safe from outside interference and misuse. One way to think about encapsulation is as a protective wrapper that prevents the code and data from being arbitrarily accessed by other code defined outside the wrapper. Access to the code and data inside the wrapper is tightly controlled through a well-defined interface. To relate this to the real world, consider the automatic transmission on an automobile. It encapsulates hundreds of bits of information about your engine, such as how much you are accelerating, the pitch of the surface you are on, and the position of the shift lever.

5.1.3.4 INHERITANCE

Inheritance is the process by which one object acquires the properties of another object. This is important because it supports the concept of hierarchical classification. As mentioned earlier, most knowledge is made manageable by hierarchical (that is, top-down) classifications. Without the use of hierarchies, each object would need to define all of its Characteristics explicitly. However, by use of inheritance, an object need only define those qualities that make it unique within its class. It can inherit its general attributes from its parent. Thus, it is the inheritance mechanism that makes it possible for one object to be a specific instance of a more general case.

5.1.3.5 POLYMORPHISM

Polymorphism (from the Greek, meaning “many forms”) is a feature that allows one interface to be used for a general class of actions. The specific action is determined by the exact nature of the situation. Consider a stack (which is a last-in, first-out list). You might have a program that requires three types of stack. One stack is used for integer values, one for floating-point values, and one for characters. The algorithm that implements each stack is the same, even though the data being stored differs. In a non-object-oriented language, you would be required to create three different sets of stack routines, with each set using different names. However, because of polymorphism, in Java you can specify a general set of stack routines that all share the same names.

5.1.4 SAMPLE CODE

```
package com.mat.DB;

import java.sql.*;

public class DBConnection {

    /**
     * @param args
     * @return
     */

    public static Connection getConnection() {

        Connection con = null;

        try{

            Class.forName("com.mysql.jdbc.Driver");

            System.out.println("<<<<<Driver is loaded>>>>>");
```

```
con=DriverManager.getConnection("jdbc:mysql://localhost:3306/bcs","root","");

System.out.println("((((((((((((Connection Success))))))))))");

}

catch(Exception e){

e.printStackTrace();

}

return con;

}

}
```

6. TESTING

6.1 TESTING

Once the source code has been generated, the software must be tested to uncover as many errors as possible before delivery to the customer. Software testing is a critical element of software quality assurance and represents the ultimate review of specification design and code generation.

There are two basics of software testing, WHITE BOX TESTING and BLACK BOX TESTING.

6.1.1 WHITE-BOX TESTING

This is performed knowing the internal workings of a product. Tests are conducted to ensure that “all gears mesh”, that is, that internal operation performs according to specification and all internal components have been adequately exercised. This can be done on close examination of procedural detail. Using the white-box testing we can derive test cases that,

1. Guarantee that all independent paths within a module have been exercised at least once.
2. Exercise all logical decisions on their true and false sides.
3. Execute all loops at their boundaries and within their operational bounds and
4. Exercise internal data structures to assure their validity.

6.1.2 BLACK-BOX TESTING

Knowing the specified function that a product has been designed to perform tests can be conducted that demonstrates each function is fully operational, at the same time searching for errors in each function. It enables us to derive sets of input conditions that will fully exercise all function requirements for documents. It attempts to find errors in,

1. Incorrect or missing functions
2. Interface errors
3. Errors in data structures or external database access
4. Performance errors
5. Initialization and termination errors

6.2 TYPES OF TESTING

Given Below are the various types of testing.

6.2.1 UNIT TESTING

Unit testing focuses verification effort on the smallest unit of software design (i.e.), the module. Unit Testing exercise specific paths in a module's control structure to ensure complete coverage and maximum error detection. This test focuses on each module individually; ensure that it functions properly as a unit. Hence, the name is unit testing.

6.2.2 INTEGRATION TESTING

Integration Testing addresses the issues associated with the dual problems of verification and program construction. After the software has been integrated a set of High-order tests are conducted. The main objectives in this testing process are to take unit-tested modules and build a program structure that has been dictated by design. Top-down integration and bottom-up integration are the two different types of integration testing.

6.2.3 VALIDATION TESTING

At the end of Integration Testing, software is completely assembled as a package, interfacing errors have been uncovered and correction testing begins.

6.2.4 SYSTEM TESTING

System testing is series of different tests whose primary purpose is to fully exercise the computer based system. Although each test has a different purpose, all the work should

verify that all system elements have been properly integrated and perform allocated functions.

6.2.5 FUNCTIONAL TESTING

Functional testing is the testing to ensure that the specified functionality required in the system requirements works. It falls under the class of black box testing.

6.2.6 PERFORMANCE TESTING

Performance testing is the testing to assess the speed and effectiveness of the system and to make sure it is generating results within a specified time as in performance requirements. It falls under the class of black box testing.

6.2.7 REGRESSION TESTING

Regression testing is the testing after modification of a system, component, or a group of related units to ensure that the modification is working correctly and is not damaging or imposing other modules to produce unexpected results. It falls under the class of black box testing.

6.2.8 ALPHA TESTING

Alpha testing is a preliminary software field test carried out by a team of users in order to find bugs that were not found previously through other tests. The main purpose of alpha testing is to refine the software product by finding (and fixing) the bugs that were not discovered through previous tests.

6.2.9 BETA TESTING

Beta testing is the testing which is done by end users, a team outside development, or publicly releasing full pre-version of the product which is known as beta version. The aim of beta testing is to cover unexpected errors. It falls under the class of black box testing.

6.3 TESTCASES

Test cases are specific executable tests that examine all aspects including inputs and outputs of a system and then provide a detailed description of the steps that should be taken, the results that should be achieved, and other elements that should be identified. Steps explained in a test case include all details even if they are assumed to be common knowledge. Test cases are used as a technical explanation and reference guide for systems.

Given below is a list of various test cases of the “BUDGET CONTROLLING SYSTEM”.

S.No	Field	Input	Expected Result	Actual Result	Status Pass/Fail	Remarks
1	User ID	NULL	Enter username	Enter username	Pass	-
2	User ID	510	No error	No error	Pass	-
3	Password	NULL	Enter password	Enter password	Pass	-
4	Password	ram	No error	No error	Pass	-
5	Name	NULL	Enter username	Enter username	Pass	-
6	Name	Ram	No error	No error	Pass	-

Table 6.1: Test cases

7. OUTPUT SCREENS

7.1 LOGIN PAGE

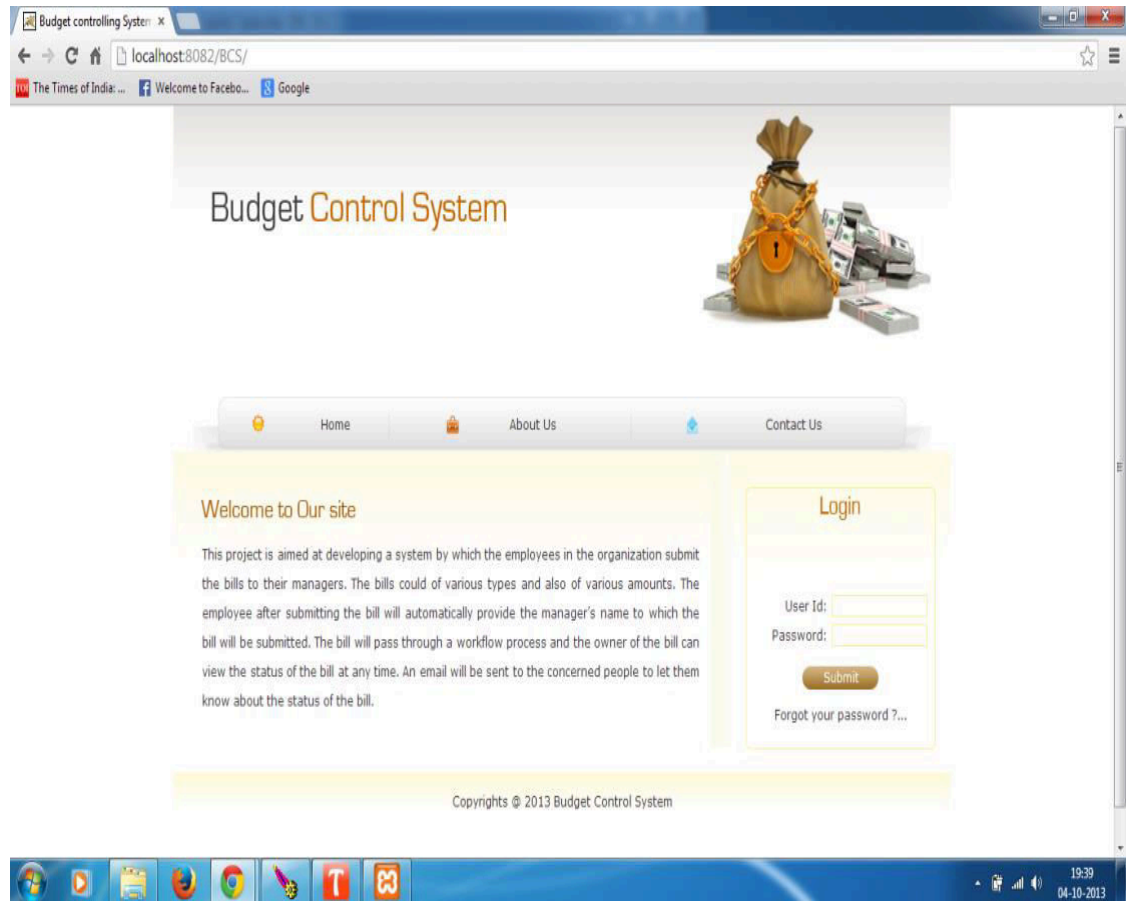


Figure 7.1: Login Page

This is the Login Page of the “BUDGET CONTROLLING SYSTEM”. From this Login Page the User can log into the system by entering his User Id and Password and clicking the “Submit” button. If the User forgets his Password, he can click on the link “Forgot your password?” to retrieve his Password. This page displays a default message on the screen. The menu bar contains three buttons, Home, About Us and Contact Us. The user can visit any of the pages by clicking their respective buttons in the Login Page.

7.2 REGISTRATION PAGE

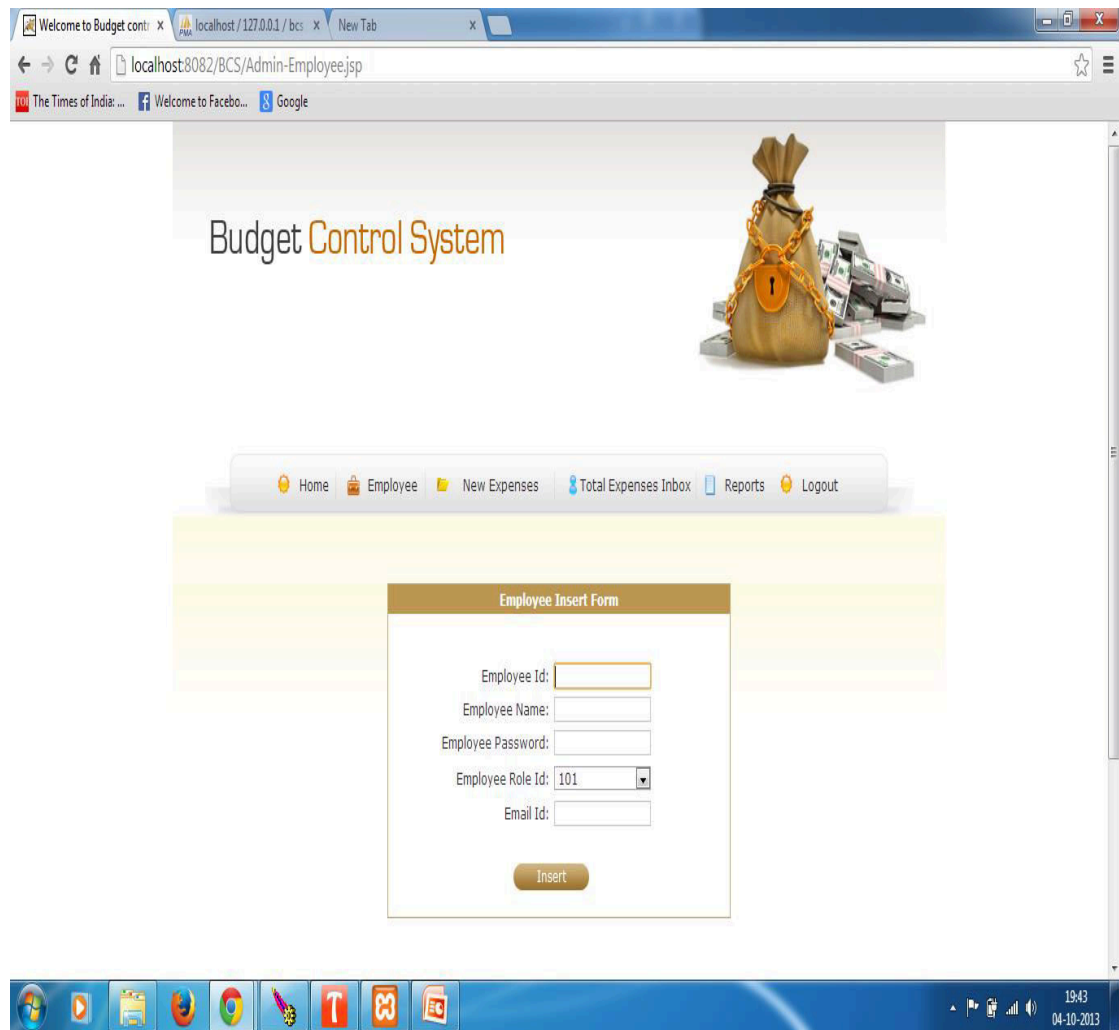
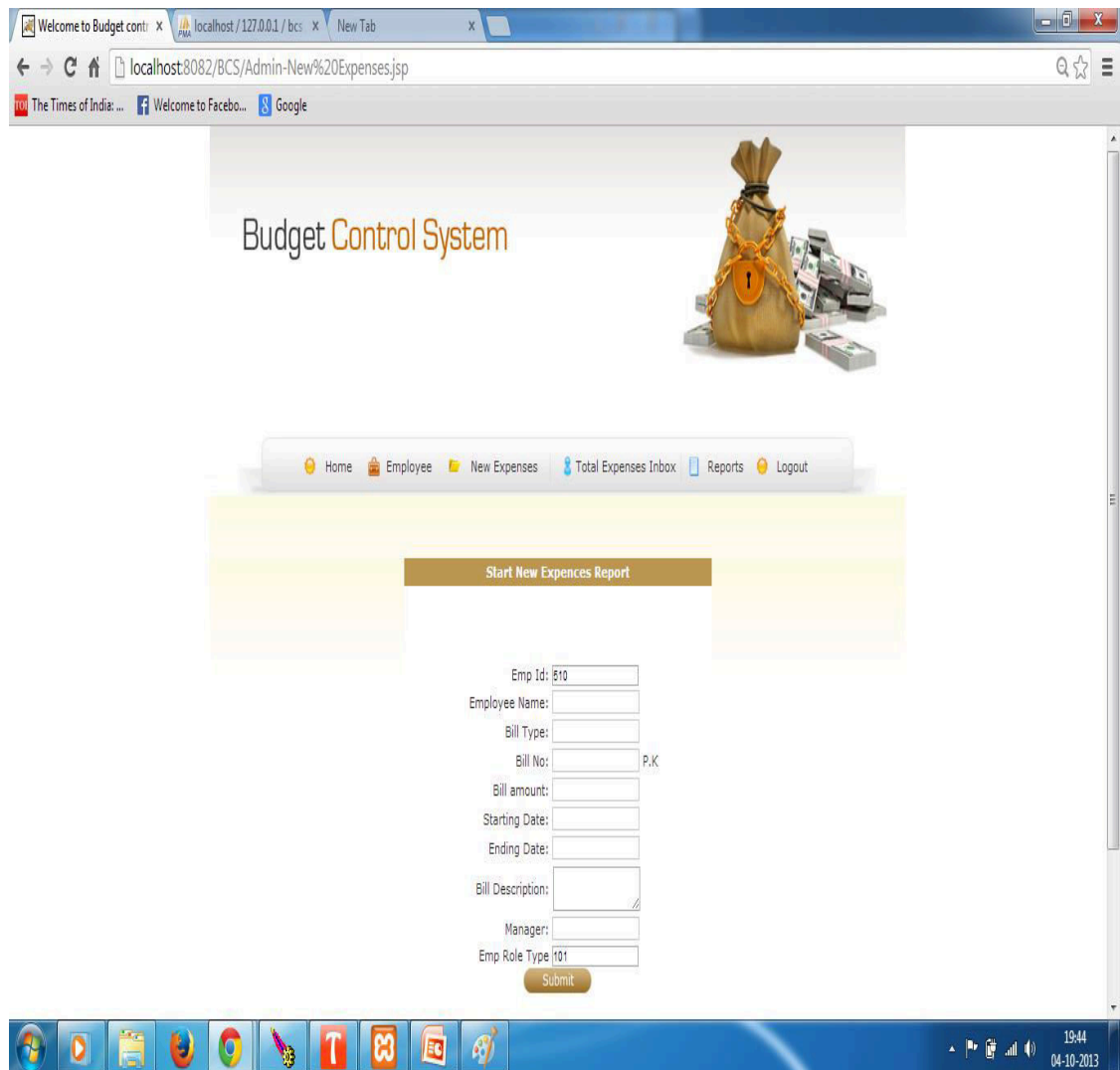


Figure 7.2: Registration Page

The User can get registered with the “BUDGET CONTROLLING SYSTEM” using the Registration Page. To register the User has to fill his details in a form as shown in the screenshot. He has to enter his Id, Name, Password, Role Id and Email Id. After entering the details the User has to click the “Insert” button. The registration process finishes. By clicking the respective buttons in the menu bar, the user can add new expenses, he can view reports, he can see total expenses and he can logout from the system as well.

7.3 NEW EXPENSE PAGE



The screenshot shows a web browser window with the URL `localhost:8082/BCS/Admin-New%20Expenses.jsp`. The page features a header with the text "Budget Control System" and an illustration of a money bag. Below the header is a navigation bar with links: Home, Employee, New Expenses, Total Expenses Inbox, Reports, and Logout. The main content area is titled "Start New Expenses Report" and contains a form with the following fields:

- Emp Id:
- Employee Name:
- Bill Type:
- Bill No:
- Bill amount:
- Starting Date:
- Ending Date:
- Bill Description:
- Manager:
- Emp Role Type:

A "Submit" button is located at the bottom of the form. The Windows taskbar at the bottom shows the date and time as 19:44 on 04-10-2013.

Figure 7.3: New Expense Page

The New Expense Page of the “BUDGET CONTROLLING SYSTEM” helps the User to create new expenses. The User has to enter his Id, Name, the Type of Bill, Bill Number, Bill Amount, Starting Date and Ending dates of the Expenses, Description of the Bill in brief, Manager and his Role Type; and then click on the button “Submit”. In this way the User can add a particular expense and report it to the Manager.

7.4 BUDGET AMOUNT PAGE

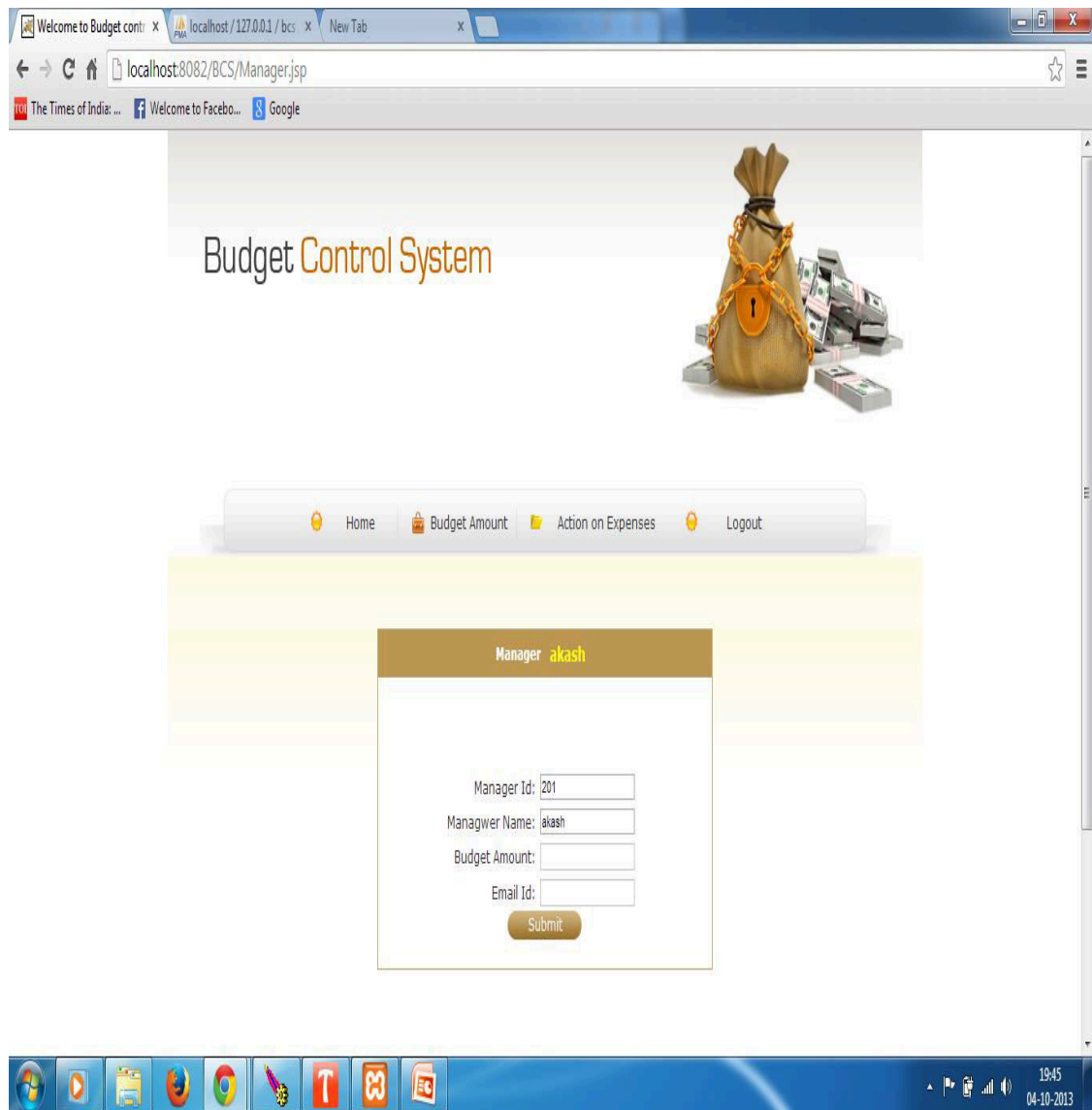


Figure 7.4: Budget Amount Page

The Budget Amount Page of the “BUDGET CONTROLLING SYSTEM” allows the Manager to view the Budget Amount. All he has to do is enter his Id, Name, Budget Amount and Email Id; and then click on the button “Submit”. The Manager can thus check the entire Budget Amount of the System.

7.5 ACTION ON EXPENSE PAGE

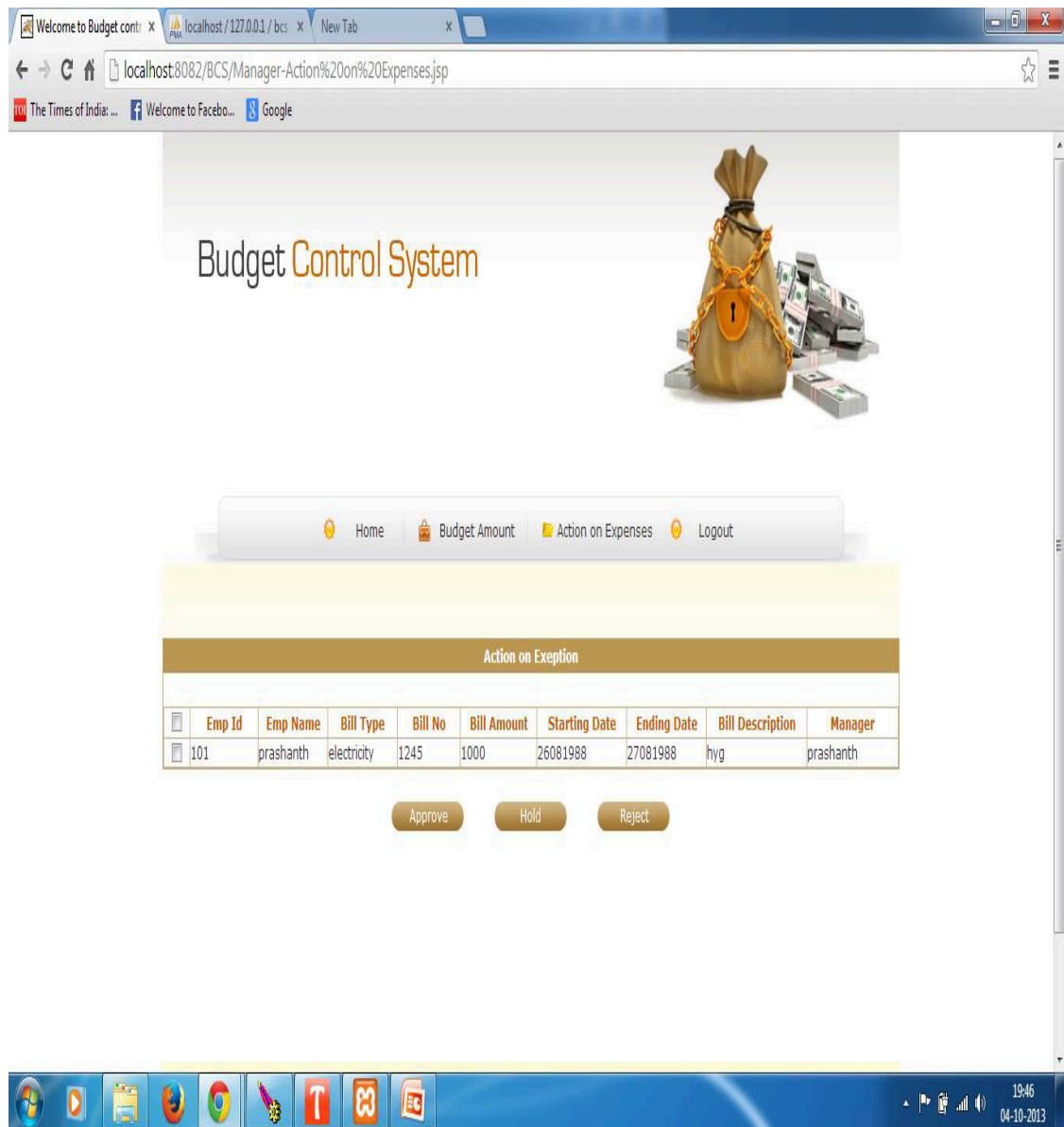


Figure 7.5: Action on Expense Page

The Action on Expense Page allows the Admin to take actions on the Expenses. He can Approve, Hold or Reject the Expenses. The particulars of the Expenses are displayed as shown in the screenshot and the Admin can take necessary actions on it.

8. CONCLUSION

8.1 CONCLUSION

By using the “BUDGET CONTROLLING SYSTEM”, the user can monitor and control the income and expenditure. The flow of cash can be controlled which is very essential in any business environment.

8.2 FURTHER ENHANCEMENTS

The BUDGET CONTROLLING system can be made a mobile application thereby making it mobile in use, frequently usable, customizable, robust and to provide a better connection to users.

9. BIBLIOGRAPHY

9.1 BOOKS REFERENCES

1. Visual FoxPro By Satish Jain
2. Using Visual Foxpro 6 By Menachem Bazian
3. Java Programming by Examples By Rajiv Sharma, Vivek Sharma
4. Java Developers Guide By Jamie Jaworski
5. Java Fundamental Classes Reference By Mark Grand, Jonathan Knudsen
6. Java In A Nutshell 2nd Edition David Flanagan

9.2 WEBSITES REFERENCES

1. www.avantetech.com
2. www.metrokc.gov
3. www.premiereelections.com
4. www.wikipedia.com
5. <http://www.freejavaguide.com/corejava.htm>
6. <http://www.learnjavaonline.org>
7. <http://www.w3schools.com>

9.3 TECHNICAL PUBLICATION REFERENCES

1. Java, Object Oriented Analysis and Design, UML By Addison-Wesley Professional
2. Writing “Real” Java-Not Just C in Java By Oracle
3. Building Secure Software in Java By Aonix
4. Real-Time Core Extensions for the Java Platform By NewMonics