CHAPTER 3

TOOLS USED

In this section we are discussing in details about the different tools that are used for design, development, implementation and testing of PReWebD and PReWebN.

3.1 The Framework

The framework is a collection of prefabricated software blocks which can be utilized, extended, or customized to suite programmers/developers application. It is an object oriented reusable mechanism that allows the developer to decompose an application into a set of interacting objects. The framework is a reusable design that describes the interfaces implemented by the components. It also provides information about the flow of control between these components, relations between the components and the system. Its standard interfaces and interactions make it possible to mix and match existing components. With frameworks, developers need not have to start from scratch each time they build an application. In this world of constantly evolving business environment, the flexibility provided with the framework enables for fast development and deployment of software applications.

A framework makes it easy to quickly build sophisticated Web applications. A framework allows a developer to build entire system, with secure, high-performance database access via an object-to-relational mapping without the need to embed SQL in the code. The framework enables the background job queuing, sophisticated event notifications and logging to facilitate management of the completed application. A framework provides a platform that enables rapid Server application development. It's open standard and open architecture approach allows enterprise solutions to stay in step with the latest development. New technologies can be easily incorporated into the framework's structure and work seamlessly with existing components and services. As technology evolves, any solution that incorporates the framework can evolve with it. A framework provides the tools for creating complex applications while mitigating the risks, resulting from the velocity of business and the heterogeneity of system
architectures. It achieves this goal by providing a combination of software subsystems, referred to as services. The services can be reused and shared to assemble applications tailored to specific business applications. It has the capability to reuse services, as well as the flexibility to add new services, that addresses the quick time-to-market concerns. In addition, the modular assembly of an application from services provides a means for integrating heterogeneous systems [30].

3.2 The Microsoft Visual Studio 2005

In the implementation of the PReWebD, we have used the Microsoft Visual Studio 2005. It is an IDE developed by the Microsoft Corporation [31]. It is used to develop console and GUI applications along with windows forms, Web sites, Web applications, and WS in both native code together with managed code for all platforms supported by the Microsoft Windows, Windows Mobile, Windows CE (Consumer Electronics or Compact Edition), the .NET Framework, the .NET Compact Framework and Microsoft Silverlight. It supports plug-ins that enhance the functionality for all kind of developments.

The Visual Studio supports languages by means of language services, provided a language-specific service exists. Built-in language support include C/C++, VB.NET (Visual Basic.NET), and C#. The support for other languages e.g. F#, M, Python, and Ruby among others is available via language services installed separately. Visual Studio also supports XML/eXtensible Stylesheet Language (XSL), HTML/XHTML, JavaScript and Cascaded Style Sheet (CSS) language. The language-specific versions of Visual Studio also exist which provide more limited language services to the user. These individual packages are called Microsoft Visual Basic, Visual J#, Visual C# and Visual C++.

3.2.1 The Architecture of Visual Studio 2005

The architecture of Visual Studio 2005 is shown in fig.3.1 along with .NET framework architecture. It does not support any programming language, solution or tool intrinsically. Instead, it allows various functionality to be plugged in. The specific functionality is coded as a Visual Studio Package (VSPackage). When installed, the functionality is available as a Service. The IDE provides three services: (i) the
SVsSolution, which provides the ability to enumerate projects and solutions; (ii) the SVsUIShell, which provides windowing and UI functionality (including tabs, toolbars and tool windows) and (iii) the SVsShell, which deals with registration of VSPackages. In addition, the IDE is also responsible for coordinating and enabling communication between services.

3.2.2 The Features

The important features include (a) the Code Editor, (b) the Debugger, (c) the Designer, (d) the Tools, (e) the Extensibility etc.

The Code Editor supports syntax highlighting and code completion using IntelliSense for variables, functions, methods, loops and queries. Autocompletion suggestions are popped up in a modelless list box, overlaid in the top of the code editor. The integrated debugger works as source-level debugger as well as machine-level debugger, which is capable of debugging any language available in the Visual Studio. The Visual Studio includes a host of visual designers to aid in the development of applications. Among them includes: (i) the Windows Forms Designers, (ii) the Windows Presentation Foundation (WPF) Designer, (iii) the Web designer/development, (iv) the Class Designer, (v) the Data Designer and (vi) the Mapping Designer. The tools available include, Open Tab Browser, Properties Editor, Object Browser, Solution Explorer, Team Explorer, Data Explorer, Server Explorer, Dotfuscator Software Services and Community Edition. The Visual Studio allows developers to extend the capabilities of the IDE. These extensions "plug into" Visual Studio and extend its functionality. The extensions come in the form of macros, packages and add-ins [32].

3.3 The Microsoft .NET Framework

The architecture of .NET framework is shown in fig.3.1. The Microsoft .NET Framework is a software framework with a large library of coded solutions to common programming problems [33]. A virtual machine runs inside the framework that manages the execution of programs written specifically for the framework. The main objectives behind the designing of .NET framework are:
(a) to provide a consistent object oriented programming environment irrespective of whether the object code is stored and executed locally or in some remote environment.

(b) to provide a code-execution environment that minimizes software and versioning conflicts.

(c) to provide a code execution environment that promotes safe execution of code, including code created by an unknown or semi-trusted third party.

(d) to provide a code-execution environment that eliminates the performance problems of scripted or interpreted environments.

(e) to make the developer experience consistent across widely varying types of applications, such as Windows-based applications and Web-based applications.

3.3.1 The Features of .NET

The main features of .NET framework are discussed below:

3.3.1.1 The Assembly

The assembly is either a Dynamic Link Library (.DLL) or Executable (.EXE) that forms a part of the application. Other important points related to assembly are: (i) it is the unit on which permissions are granted, (ii) every assembly contains a version, (iii) the assembly contains interfaces and classes and (iv) the assembly contains assembly metadata, which contains information about assembly. The Common Language Runtime (CLR) uses this information at the time of executing this assembly, (v) Assembly may be either private, which are used only by the application to which they belong or Global assemblies, which are used by any application in the system, and (vi) Two assembly with the same name but with different versions can run side by side in the same system allowing applications that depend on a specific version to use assembly of that version.

3.3.1.2 The Common Type System

Common Type System (CTS) specifies the rules related to data types that languages must follow. The data types in all languages being converted to certain
Fig. 3.1: The .NET framework architecture
standard data types, the programs written in all languages are ultimately converted to Microsoft Intermediate Language (MSIL).

### 3.3.1.3 The Cross Language Interoperability

One of the important features of .NET is the support for language interoperability. Some set of rules called Cross Language Specification (CLS) must be followed to enable a program to be used with other languages [34].

The two main components of .NET Framework are: (a) the CLR and (b) the .NET Framework class library (FCL) as discussed below [35].

### 3.3.2 The CLR

The CLR is an execution environment. It can be termed as the foundation of the .NET framework. It is a language independent runtime environment that works as a layer between the operating systems and the .NET application. During the execution of the application, the CLR manages code, memory, thread execution, garbage collection, exception handling, CTS, code safety verification and other system services. The CTS being the standard type and can be used by all .NET programming languages. All .NET languages use the same representation for common data types. The CLR is also termed as a managed environment [36]. The main features of CLR are:

(i) It manages memory, thread execution, code execution, compilation of code safety verification, and other system services.

(ii) For security reasons, managed code is provided with varying levels of trust based on origin. This prevents or allows the managed component from performing file access operations, registry access operations or other sensitive functions even within the same active application.

(iii) The Runtime enforces code robustness by implementing strict type and code verification infrastructure, called the CTS. The CTS ensures that all managed code is self describing and all Microsoft or third party language compiler generated codes use the same representation for common data types.
(iv) It also eliminates many common software issues like handling of object layout, references to objects and garbage clearance. This type of memory management prevents memory leaks and invalid memory references.

(v) It accelerates developer productivity. The programmer is free to choose the language of the application without worrying about compatibility and integration issues. He is also enabled to take advantage of the runtime and the class library of the .NET Framework and also harvest components from other applications written in different languages by different developers. This implicitly eases the process of migration.

(vi) Though CLR aims to be futuristic software, it lends support to existing applications. The interoperability between the managed and unmanaged codes makes this process extremely simple.

(vii) The design of the CLR is geared towards enhancing performance. The Just-in-time (JIT) compiling enables the managed code to run in the native machine language of the system executing it. During the process, the memory manager removes the possibilities of fragmented memory and increases memory locality-of-reference to enhance performance.

(viii) Finally, Server side applications can host runtime. High performance Servers like Microsoft SQL Server and Internet Information Services (IIS) can host this CLR and the product infrastructure can be used to write business logic while enjoying the best benefits of enterprise Server support [37].

3.3.3 The Architecture of CLI

The architecture of CLI is shown in fig.3.2. It can be termed as the foundation of .NET technology, is a European Computer Manufacturers Association (ECMA) standard that describes the executable code and runtime environment which form the core of the Microsoft .NET framework. The block diagram of CLI is shown in fig. 3.2. It allows applications to be written in a variety of high-level programming languages and executed in different system environments [38]. It is an international standard which is the basis for creating execution and development environments in which languages and libraries work seamlessly. The CLI specifies a virtual execution system
Fig. 3.2: The Common Language Infrastructure

- .NET compatible languages compile to a second platform neutral language called Common Intermediate Language (CIL).
- The platform-specific Common Language Runtime (CLR) compiles CIL to machine readable code that can be executed on the current platform.
that insulates CLI compliant programs from the underlying operating system [39]. The CLI specifies a Common Language Specification (CLS), a common set of data types that any language must support, an introduction to the component structure, the way machine state is managed, and the way exceptions are handled [40].

3.3.4 The .NET FCL

The .NET FCL provides the core functionality of the .NET Framework architecture. It is divided into two parts, namely FCL and Base Class Library (BCL) which contains a huge collection of reusable classes, interfaces and value types that optimize the development process and provide access to system functionality. The .NET FCL is organized in a hierarchical tree structure and is divided into Namespaces. The Namespaces is a logical grouping of types for the purpose of identification. The FCL provides the consistent base types that are used across all .NET enabled languages. The Classes are accessed by namespaces, which reside within assemblies. The system namespace is the root for types in the .NET framework. The .NET FCL classes are managed classes which provide access to system services. These classes are object oriented and easy to use in program developments. Also, third party components can integrate with the classes in this framework [41]. One part of the FCL is the Base Class Library (BCL) that provides the most fundamental functionality, which includes classes in Namespaces System, System.CodeDom, System.Collections, System.Diagnostics, System.Globalization, System.IO, System.Resources, System.Text and System.Text.RegularExpressions etc. [42].

The BCL is included in .NET framework in order to encapsulate a large number of common functions, such as file reading and writing, graphic rendering, database interaction and XML document manipulation which makes the programmers life easier [43].

3.3.5 The ADO.NET

The architecture of ActiveX Data Object.NET is shown in fig.3.3. The ADO.NET is a part of BCL and a set of computer software components which are used to access data and data services. It is a disconnected data-access approach i.e. data source connection is kept open in need, used to access and modify the data stored in the
Fig. 3.3: The ADO.NET architecture
relational database system [44]. In multi tier architecture, disconnected approach helps for better scalability. Since today's Web application model uses XML to encode data and HTTP as the communication medium between tiers so maintaining state between request is mandatory. The ADO.NET, which understands XML, designed for this new programming model comprises of disconnected data architecture, tight integration with XML, and Common Data Representation (CDR). So, the main attraction behind the ADO.NET working model is disconnected data and the ADO.NET solution for n-tier model in dataset. The Key components in ADO.NET are Dataset and Data Provider as discussed below.

3.3.5.1 The Dataset

The Dataset handles data access. It is an in-memory representation of data. It can be used with multiple and differing data sources such as XML file or stream of data which are local to application. The Dataset can be used if an application that meets any of the following requirements. (a) Remote data access, (b) Multiple and different data sources, (c) Cache data local to the application, (d) Provide hierarchical XML view of a relational data, (e) use of eXtensible Stylesheet Language (XSL), eXtensible path (XPath tools), and (f) Disconnected data access.

3.3.5.2 The Data Provider

The data in source is retrieved using various components of data provider. It also provides data to an application and then updates changes back to the database. The ADO.NET classes are in System.data namespace. The components of data provider are: (i) the Connection, (ii) the Command, Data adapter and (iii) the Data Reader.

3.3.5.2.1 The Connection Object

It establishes connection to the data source. In Microsoft SQL Server 7.0 or later, SQL Connection object is used. Properties of connection object are:

(a) The connection String: It is a primary property. The connection string provides information like, data provider, data source, database name, integrated security and so on.
(b) The state Property: It provides information about current state of the connection object. State property value zero specifies connection closed and one specifies connection is opened.

3.3.5.2.2 The Command Object

The data from data source is obtained by executing commands using command objects. Properties of command object are:

(a) The Command Text Property: It possesses SQL statements or name of the stored procedure to perform the command.

(b) The Command Type property: It contains stored procedure setting or text setting or table direct. The table direct returns entire table. The command object provides several execute methods like, execute scalar, execute nonquery, and execute reader.

3.3.5.2.3 The Data Adapter

It acts as a bridge between data source and data set to retrieve data and reconciles data back to the database. It exchanges data between single data source and single data table object in the data set. For multiple tables in dataset multiple data adapter should be used. To populate a table in dataset, data adapter can be called which execute Sql Statement or Stored Procedure. The data Adapter supports Select Command, Insert Command, Update Command and Delete Command Properties.

3.3.5.2.4 The Data Reader

The Data Reader can be used to increase the performance of the application which works in read-only, forward-only mode [45].

3.3.6 The life cycle of ASP.NET page

The lifecycle of an ASP.NET page is as discussed below [46]. When a specific page of the Web application is requested, the application proceeds through the following life-cycle events:
3.3.6.1 The Page Request

When the user requests for a specific page, the ASP.NET determines whether the page needs to be parsed and compiled, or whether a cached version of the page can be sent in response to the user request without running the page. The response and request objects are then set and the page lifecycle starts.

3.3.6.2 The Start

In the start stage, the Request and Response properties of the page are set. At this stage, the page also determines, whether the request is a postback or new request, and the IsPostBack property is set. The page also sets the UICulture property.

3.3.6.3 The Initialization

In this stage, the Server controls on the page are available and each controls UniqueID property is set. The master page is also applied to the page. If the current request is a postback, the postback data are not load yet and control property values are not been restored from the view state.

3.3.6.4 The Load

During the load, if the current request is a postback, control properties are loaded with information recovered from view state and control state.

3.3.6.5 The Validation

During the validation, the validate method of all the validator controls are called, and sets the IsValid property of the individual validator controls of the page.

3.3.6.6 The PostBack event handling

If the request is a postback, control event handlers are called. The validate methods of validator controls are called, which sets the IsValid property of individual validator controls of the page.
3.3.6.7 The Rendering

Before the rendering, view state is saved for the page, and all controls. During the rendering stage, the page calls the Render method for each control and the output is displayed.

3.3.6.8 The Unload

The Unload event is raised after the page has been fully rendered and is ready to be discarded. The page properties such as Response and Request are unloaded and cleanup work, such as closing open files and database connections is performed.

3.3.7 The Web Server

A Web Server is a computer program that delivers content, like Web pages, using the HTTP and the WWW. It is a computer with some kind of Server software installed to it. The main objective of a Web Server is to deliver Web pages to clients. A Web page is generally an HTML document which can consist of images, style sheets, client-side scripting languages etc.

To illustrate it in simple language, let us assume that there is a client, who is initiating for a request resides in another computer. This computer is acting as Server, holding the information. The Server will simply receive the request, process it and send back a set of codes which the client browser can understand. The browser will then display the information on the computer screen. The history of evolution of Web Servers revolves around Tim Berners-Lee, who proposed a new project to his employer CERN in 1989 [47]. The project had the goal of easing the exchange of information between scientists using a hypertext system.

Two leading Web Servers are Apache HTTP Server which is the most widely used Web Server and Microsoft's IIS [48]. Other examples include Sun Java System Web Server, IBM HTTP Server, SunOne Iplanet etc.

3.3.8 The Microsoft SQL Server 2005

The Microsoft SQL Server 2005 Express Edition is easy-to-use, lightweight, embeddable version of SQL Server 2005. The SQL Server Express being integrated with Microsoft Visual Studio 2005, this combination makes it easy to develop
powerful, secure, data-driven applications and deploy them quickly. The developers can design schemas, add data, and query of local databases all inside the Visual Studio 2005 environment [49]. It includes many important features which are helpful to developers. It uses the advanced query optimization engine of SQL Server 2005 to deliver fast performance. The full text search is one of the key features of the SQL Server Express. Some other key features are:

(i) easy to install and manage. The installation is silent when it is embedded with Visual Studio 2005.

(ii) support for distributed transaction.

(iii) support for XML. It supports native XML datatype, structures and semi-structured data, XQuery and XML schema.

(iv) supports rich database functionality such as, stores procedures, views, triggers, cursors, extended indexes, snapshot isolation, advances query optimizer etc.

(v) embedded with Microsoft Visual Studio 2005, Visual Studio Data explorer is used for designing schemas, adding data, and querying local databases. The CLR is hosted inside SQL Server Express. In-process data access facility is provided with ADO.NET.

(vi) security is provided with default settings [50].

3.3.9 The IIS

The IIS is a Web Server application created by Microsoft Corporation for use in Microsoft Windows. It is a group of Internet Servers consisting of a Web Server, a File Transfer Protocol (FTP) Server etc. [51]. It has got additional capabilities for Microsoft’s Windows NT and Windows 2000 Server operating systems [52]. A recent survey showed that it serves 24.47% of all Websites on the Internet. Important features of IIS include some modules, also called extensions, whose functions are discussed below:

3.3.9.1 The HTTP Modules

It performs the tasks specific to HTTP. Such as responding to information sent in client headers, returning HTTP errors, and redirecting requests.
3.3.9.2 The Security Modules

It performs the tasks specific to security e.g. specifying authentication schemes, performing URL authorization, and filtering requests.

3.3.9.3 The Content Modules

It performs the tasks related to the content e.g. processing requests to static files, returning a default page when a client does not specify a resource in a request, and listing the contents of a directory.

3.3.9.4 The Compression Modules

It performs the tasks related to compression e.g. the compressing responses, applying Gzip, compression transfer coding to responses and performing pre-compression of static content.

3.3.9.5 The Caching Modules

It performs the tasks related to caching e.g. storing processed information in memory on the Server and using cached content in subsequent requests for the same resource.

3.3.9.6 The Logging and Diagnostics Modules

It performs the tasks related to logging and diagnostics e.g. passing information and processing status to HTTP.sys for logging, reporting events, and tracking requests currently executing in worker processes [53].

3.4 The NetBeans IDE

The NetBeans IDE is an open-source IDE. It is free for commercial as well as non-commercial use. It is a computing platform framework for applications developing with Java, JavaScript, PHP, Python, Ruby, Groovy, C, C++, Scala and Clojure. It is written in Java and runs everywhere where a JVM is installed. It runs in any operating systems including Windows, Mac, Linux and Solaris.
3.4.1. The History of Evolving

The NetBeans began in the year 1996 as Xelfi (word play on Delphi), a Java IDE student project under the guidance of the Faculty of Mathematics and Physics at Charles University in Prague. In the year 1997 Roman Staněk formed a company around the project and produced commercial versions of the IDE. The Sun Microsystems bought the IDE in the year 1999. The Sun open-sourced the IDE in June 2000. The NetBeans community has since continued to grow. The individuals and companies who are using it, contributed to the project.

3.4.2 The Version Used

In the implementation of the PReWebN, we have used the NetBeans IDE 6.5.1. It comes with support for developing IDE modules and rich client applications based on the NetBeans platform, a Java Swing GUI builder (formerly known as "Project Matisse"), improved Concurrent Versions System (CVS) support, Weblogic 9 and JBoss 4 support and many editor enhancements.

3.4.3 The NetBeans Platform

The NetBeans platform is a reusable framework which simplifies the development of Java applications. The Java standard edition (SE) of NetBeans IDE contains the resources needed to develop NetBeans plugins and NetBeans applications. No additional Software Development Kit (SDK) is required.

When an application is run, the main class of the application is executed. The modules available are located, placed in an in-memory registry, and the modules startup tasks are executed. The modules' code is loaded in the memory only when it is required. The platform offers reusable services, common to desktop applications, allowing developers to focus on the logic specific to their application. It comes with dozens of tools which makes programming easy. For example, it makes easy to connect to any database with a Java Database Connectivity (JDBC) driver, it provides tools for browsing database tables and views. It enables interactive execution of SQL commands. The main features of the platform are:

(i) the UI management through menus and toolbars.

(ii) user settings management.
3.4.4 The modules of NetBeans IDE

Different modules of the IDE is discussed below.

3.4.4.1 The NetBeans Profiler

The NetBeans Profiler is a tool for monitoring Java applications. It helps in finding memory leaks and optimize speed. It is integrated to the core IDE since version 6.0. The NetBeans Profiler is based on a Sun Laboratories research project that was named as JFluid. The project uncovered specific techniques which can be used to lower the overhead of profiling a Java application. One of those techniques is dynamic bytecode instrumentation, which is particularly useful for profiling large Java applications. Using dynamic bytecode instrumentation and additional algorithms, the NetBeans Profiler is able to obtain runtime information on applications that are too large or complex for other profilers. It also supports profiling points that lets a user profile precise points of execution and measure execution time.

3.4.4.2 The GUI Design Tools

The GUI design tool enables a developer to prototype and design swing GUIs by dragging and positioning different GUI components. The GUI builder also has built-in support for Java Specification Request (JSR) 296 (Swing Application Framework), and JSR 295 (Beans Binding technology).

3.4.4.3 NetBeans JavaScript Editor

The NetBeans JavaScript Editor provides extended support for JavaScript, Ajax and Cascading Style Sheet (CSS). JavaScript editor features comprise syntax highlighting, refactoring, code completion for native objects and functions, generation of JavaScript class skeletons, generation of Ajax callbacks from a template and automatic browser compatibility checks. The CSS editor features comprise code
completion for styles names, quick navigation through the navigator panel, displaying the CSS rule declaration in a List View and file structure in a Tree View etc.

3.4.5 The NetBeans IDE download bundles

Different download bundles of the IDE are as discussed below.

3.4.5.1 The NetBeans IDE Bundle for Web & Java EE

The NetBeans IDE Bundle for Web & Java Enterprise Edition (JEE) provides complete tools for all the latest JEE 6 standards, including the new JEE 6 Web Profile, EJBs, Servlets, Java Persistence Application Programming Interface (API), WS, and annotations. The NetBeans also supports the JSF 2.0 (Facelets), JSP, Hibernate, Spring, and Struts frameworks, and the J5EE and J2EE 1.4 platforms.

3.4.5.2 The NetBeans IDE Bundle for Ruby

The NetBeans IDE Bundle for Ruby includes a Ruby editor, debugger, and full support for the Ruby on Rails framework. It supports the development with Ruby and JRuby, as well as Rails for these two implementations of Ruby.

3.4.5.3 The NetBeans IDE Bundle for Java Micro Edition

The NetBeans IDE Bundle for Java Micro Edition (JME) is a tool for developing applications that run on mobile devices, in general the mobile phones, but this also includes entry-level Personal Digital Assistant (PDA), and Java Card, among others.

3.4.5.4 The NetBeans IDE Bundle for C/C++

The NetBeans IDE Bundle for C/C++ adds support for C/C++ developers to NetBeans IDE. This pack permits the C/C++ developers to use their specified set of compilers and tools in conjunction with IDE to build native applications for Windows, Mac OS X, Linux, and Solaris. The pack makes the editor language-aware for C/C++ and provides project templates, a dynamic class browser, makefile support, profiler and debugger functionality. The NetBeans C/C++ bundle does not include a C/C++ compiler.
3.4.5.5 The NetBeans IDE Bundle for PHP

The NetBeans supports PHP since version 6.5. The bundle for PHP includes: basic editing, syntax highlighting, code completion, occurrence highlighting, semantic analysis with highlighting of parameters and unused local variables, PHP code debugging with xdebug, PHP Unit testing with PHPUnit and Selenium, Code coverage, Symfony framework support (since version 6.8), Zend Framework support (since version 6.9), PHP 5.3 namespace and closure support (since version 6.8).

3.4.5.6 The NetBeans IDE Bundle for JavaFX

An IDE bundle is also available for JavaFX since NetBeans 6.5. Additional to full Java Standard Edition (JSE) support, it provides a JavaFX editor and a JavaFX composer. It allows to create, preview and profile JavaFX Desktop and Java mobile applications.

In a nutshell, the IDE provides features: NetBeans Base IDE, JSE, JavaFX, Web & Java EE, Java ME, Ruby, C/C++, PHP (Version 6.5 and later), GlassFish, Apache Tomcat [54].

3.4.6 The Utilities of the IDE

Using NetBeans for developing applications saves time by managing windows, settings and data. In addition, the IDE can store repetitive tasks through macros and abbreviations. The drag-and-drop features enables to create GUI components or accessing databases easily. The highlighted code and debugging features alert developers about errors in the code [55]. The main advantages of using the IDE are: (a) extensible and easy modular design, (b) vast API of commonly used tasks, (c) easy to design swing API, (d) pre-defined update center and (e) automatic update notification [56]. The IDE helps to construct ready made windows forms with all required buttons, labels, text boxes etc. to ease users as well as developers. The database integration is one of the important advantages by providing a Create, Read, Update and Delete (CRUD) application shell.
3.4.7 The Features of NetBeans IDE 6.5.1

Although the NetBeans IDE has got some common features present in other IDEs, it is very well designed and facilitates the creation of desktop as well as Web and mobile applications. The main features of NetBeans IDE 6.5.1 are:

(i) Java Desktop Applications: It makes developers task simple by building rock solid Java Swing desktop applications with the NetBeans framework.

(ii) Connected Developer Integration: Open-source projects can be created and host them on kenai.com.

(iii) JEE and Web Applications: Web applications can be built using CSS, JavaScript and JSP. The framework also supports JSF (Facelets), Struts, Spring, Hibernate. Full set of tools for J6EE, GlassFish v3, EJB, and Web services development.

(iv) Dynamic Languages: Powerful editor for JavaFX, JavaScript, CSS, PHP and Symfony, Groovy and Grails, Ruby and Ruby on Rails, and Python. Includes semantic and syntactic highlighting, smart code completion, documentation pop-ups, and debugging integration.

(v) Visual Mobile Development: GUI applications which run on mobile phones, set-top boxes and Personal Digital Assistant’s (PDA) can be created tested and debugged using NetBeans IDE.

(vi) C and C++ development: Full featured C, C++ editor, profiler, debugger, project templates, and support for multiple project configurations, remote development, and packaging of completed projects [57, 58].

3.4.8 The JSF Overview

The architecture of the JSF is shown in fig.3.4. The JSF is a standard UI framework for Java Web Applications. The rapid Web application development is promoted by easily assembling UI components, plumbing them to the back end BL components, and wiring UI generated component events to Server-side event handlers. The JSF API is composed of the following:

(i) a set of classes to represent standardized UI components, manage state, perform event handling, and input validation.

(ii) custom tag libraries for accessing UI components from JSP.
Fig. 3.4: The Architecture of the JSF
A JSF application typically runs on the Web Server and renders the UI back to the client. A typical JSF application development team will primarily consist of:

(a) UI Component Developers: They create UI components either directly from components classes, or by extending them from UI components provided by the framework.

(b) Application Developers: They are essentially the integrators, responsible for connecting the UI components to event handlers, creating model objects to represent their state, and programming page navigation and error handling in the application.

(c) Web Page Authors: They express the UI components in the presentation layer. They will use either the standard HTML tag library provided by the framework for a custom tag library, provided by the component developers. They use these tags to reference the UI components in JSP and attach model objects and validators to them.

### 3.4.8.1 The different layers of JSF Architecture

It is composed of six different layers as discussed below [59, 60]:

(i) **The UI Component**

The UI component model primarily defines the functionality of the components. It consists of Java Beans representing various models of input, selection, and grouping capabilities.

(ii) **The Rendering**

The rendering model defines the presentation aspects of a UI component. The same UI component can be rendered in a different way by creating multiple renderers.

(iii) **The Event Model**

The JSF event model is similar to JavaBeans event design. It defines listeners and event classes that a Web application can use to handle events generated by UI components. The JSF implementation supports two
different kinds of events: (a) Components value change events and (b) Action events.

(iv) The Validation Framework
The validation framework provides a mechanisms by which validators can be registered with UI components.

(v) The Page Navigation Framework
The page navigation Framework provides an easy declarative mechanism to specify the sequence of pages to be loaded without requiring any special code in the application.

(vi) The Internationalization Framework
The internationalization framework provides an easy mechanism for localizing static data, dynamic data, and messages in applications. The static data can be localized using the standard tag library internationalization tags and providing resource bundles (properties files containing messages for different languages tied to key values) and associating specific data in JSP pages with keys.

3.4.8.2 The JSF page lifecycle
The block diagram of JSF life cycle is shown in fig.3.5. The standard lifecycle for a typical JSF page can be described in the following sequences.

3.4.8.2.1 The Reconstruct Component Tree
When a page request is submitted to the Server, the Server creates a hierarchical tree of the UI components constituting the page. The JSF engine parses this component tree and all declared validators and event handlers are wired to the specific components. This information is finally persisted in a FacesContext and is utilized in all the phases of the lifecycle until the response is rendered.
Fig.3.5: The Life cycle of JSF
3.4.8.2.2 The Apply Request Values

In this phase, the JSF engine applies the values from the parameters to the respective components in the component tree stored in the FacesContext. These values are stored locally in the UI components. If any event handlers were associated with any component, they are notified to the relevant events.

3.4.8.2.3 The Perform validations

In the validation phase, all registered validators are applied to the relevant UI components. The locally stored values are run through these validators and if any errors occur, they are queued in the FacesContext. If validation errors occur in this phase, the page enters the render response phase when the JSF engine skips the other phases of the lifecycle.

3.4.8.2.4 The Synchronize Model

In this phase, any model objects associated with the UI components are synchronized to the local component value. If any conversion errors occur in this phase, the errors are queued in the FacesContext, and the page enters the render response phase of the lifecycle.

3.4.8.2.5 The Invoke Application Logic

In this phase, the JSF engine handles all application level events, such as linking to another pages, performing updates on backend services etc. When processing an application level event, a default ActionListener determines the outcome of the action and passes the outcome to the NavigationHandler. The NavigationHandler then looks up the response to be generated from the navigation rule defined in the application configuration file. The component tree for the next page is generated and the cycle enters the last phase.

3.4.8.2.6 The Render Response Phase

In this phase, the JSF engine renders the UI components in the component tree persisted in the FacesContext.
3.4.9 The Server

The Apache Tomcat or Jakarta Tomcat or simply Tomcat is an open source servlet container developed by the Apache Software Foundation (ASF). The Tomcat implements the Java Servlet and the JSP specifications from Sun Microsystems and provides a pure Java HTTP Web Server environment for Java code to run [61, 62]. The Apache Tomcat includes tools for configuration and management, but can also be configured by editing XML configuration files [63]. It has been used as the Web container which allows to run servlet and JSP based Web applications. It also provides per default a HTTP connector on port 8080, e.g. Tomcat can also be used as HTTP Server even though the performance of Tomcat is not as good as the performance of the Apache HTTP Server [64].

3.4.9.1 The Components of Apache Tomcat

The components present in Apache tomcat are: Jasper, Catalina and Coyote.

The Jasper is Tomcat’s JSP Engine. It parses JSP files to compile them into Java code as Servlets. At runtime, it is able to automatically detect JSP file changes and recompile them.

The Catalina is Tomcat’s Servlet container. It implements Sun Microsystems' specifications for servlet and JSP.

The Coyote is Tomcat’s HTTP connector component that supports the HTTP 1.1 protocol for the Web Server or application container. It listens for incoming connections on a specific TCP port on the Server and forwards the request to the Tomcat Engine to process the request and send back a response to the requesting client.

3.5 The MySQL Database Server

The MySQL is an open source relational database management system (RDBMS). The SQL is the base of MySQL, which is used to insert, delete and modify information from the database tables. It is owned by Sun Microsystems, a subsidiary of Oracle Corporation [65]. Any dynamic Web application can use the MySQL in the sense that the content of the page changes as the page loads depending on some conditions. Those Web applications are often referred to as database-driven Web applications. The NetBeans and MySQL both being open source and so these two can
be a good combination for building database driven applications [66]. Over four millions installation of MySQL has done so far, including production systems at companies like, Yahoo!, Cisco, Google, the US Census Bureau, National Aeronautic and Space Administration (NASA), Motorola, Texas Instruments, Silicon Graphics and others. Though it seems that free software must be inferior to commercial products, MySQL has been recognizing as a true enterprise RDBMS. It is reviewed by PC Magazine as one of the top five databases.

The factors for selection criteria of MySQL databases are: Cost, Ease of Use, Security, Functionality, Job Market Appeal, and Compatibility [67].

3.6 The Testing Tools

The tools used for testing PReWebD and PReWebN are discussed below:

3.6.1 The Mercury LoadRunner

The Mercury LoadRunner (Version 8.0) is used for testing and evaluation of PReWebD and PReWebN. It is an automated performance and load testing product for examining system behavior and performance, while generating actual load. It can emulate hundreds of thousands of concurrent users to put the application through the rigidity of real life user loads, while collecting information from key infrastructure components (Web Servers, database Servers etc). The results can then be analyzed in details, to explore the reasons for particular behavior [68].

The Mercury LoadRunner contains the following components:

(i) The Virtual User Generator (VuGen), which is used to emulate the steps of real human users. It captures end users business processes and creates an automated performance testing scripts, which is known as a virtual user script. It creates the load by running virtual users.

(ii) Once a script is prepared in VuGen, it runs through the LoadRunner controller. The Controller organizes drives, manages and monitors the scenarios that are run during the load test. It can simulate load for thousand of virtual users. The LoadRunner controller enables the concerned user to control several aspects of the performance test, like,
virtual user scripts, number of virtual users per scripts, ramp up time, ramp
down time, execution time etc [69].

(iii) The Analysis helps in viewing, dissecting, and comparing the performance
results. The analysis tool takes the result from the completed scenario and
prepares graphs and results which are used to correlate system information
and identify bottlenecks and performance issues. These results can then be
prepared in the form of HTML or Microsoft Word format.

(iv) The Launcher provides a single point of access for all of the LoadRunner
components [70].

The next chapter of the thesis describes the design aspects and implementation
of the PReWebD and PReWebN by using different tools as discussed in this chapter.