

Project Ditto

by

“Anurag Kumar”

Roll No. 1326453

“August 2015”

A 6 weeks training project submitted in partial fulfilment of the requirements
for the degree of Bachelor of Science in Computer Science.



The Computer Science Department
Swami Vivekanand Institute of Engineering & Technology
chandigarh-patiala Highway, Sector-8, Ramnagar, Banur
“2015”

I hereby certify that this 6 weeks training project, submitted by “Anurag Kumar,” conforms to acceptable standards and is fully adequate in scope and quality for the degree of Bachelor of Science in Computer Science.

Dr. Lalita Bhutani
Head Of Department (HOD) of Computer Science

Date



The Computer Science Department
Swami Vivekanand Institute of Engineering & Technology
chandigarh-patiala Highway, Sector-8, Ramnagar, Banur
“2015”

Acknowledgements

I always thank my God as I remember you in my prayers, Philemon 1:4

I would like to acknowledge the assistance of Mr. Gaurav Kwatra, who was my tutor and mentor during my 6 weeks training program on Java Web based Application Development at the company, Kyrion Networking Solutions Pvt. Ltd., as he helped me to understand the programming languages and frameworks that I implemented in this project.

Course Objectives

At the conclusion of Project Ditto, I will:

- ☐ showcase my knowledge and command of computer science principles by completing an acceptable project.
- ☐ appreciate the *grand ideas* of computer science by understanding fundamental computer science concepts.
- ☐ enhance programming skills through actual program synthesis.
- ☐ enhance technical communication skills by doing written documentation of my project.

Index

Process Documentation

Project Proposal	1
Resources Used	12
Project Directory Hierarchy	13
Design Documentation	14
Application Screen-shots.	15
Github Link containing Project and Code.	16

Project Proposal

This project is a light hearted **socialisation** project named “**Project Ditto**”. The word Ditto is used in this project as the main concern that arises in this web based application is to share any random act of humour or “**Silly Deed**” performed by one of the users, which of course will be visible to every user on the page itself, and if any other user has also done that deed or took it as a challenge and perform that deed after reading can increment the ditto or “**Me Too**” count of the deed itself.

Users could also use the idea of **challenging other users** to perform the task as an act of social service, they can put a selfless deed on the list of tasks and then whosoever performs the deed can acknowledge the greatness of deed and of course **increment the ditto count at a vary fast rate**. Or users could just use this application to add random acts of humour that they would like others to perform.

The **User Interface or UI** of this project is very lite as no pre-rendered style sets are used in this project. Also improving the relationship between a user and the application itself. So that it can be easily used at a very heavy amount of frequent user interactions without any technical glitch. I have tried to reduce the amount of button clicks used to interact with the application for the sake of creating memorable **User Experience or UX** for the account holders on the web page.

The below standard operations are available for any user to interact with the application:

- ☐ Signing up (highly recommended).
- ☐ Logging in.
- ☐ Publishing a Post or a Silly Deed.
- ☐ Edit any Post created by it's respective user only.
- ☐ Pushing the “Ditto Me” button for any deed to acknowledge that user has performed that deed himself / herself.
- ☐ Viewing the list of all the deeds uploaded on the web application ordered according to the publishing date (feature provided on all viewing pages).
- ☐ Deleting any published deed by it's respective user only.
- ☐ Logging out of the application (not recommended).

Resources Used

Below is the list of resources I used to develop this project (including programming languages, IDE and all the frameworks):

- **Spring Tool Suite (RELEASE 3.1.11):** STS is an eclipse based spring framework oriented IDE for J2EE development.

STS provide a customized all-in-one Eclipse based distribution that makes application development easy. The tool suites provide ready-to-use combinations of language support, framework support, and runtime support, and combine them with the existing Java, Web and Java EE tooling from Eclipse. An integrated dashboard in the distribution makes it trivial to install further extensions that add support for technologies such as Pivotal Cloud Foundry, Gradle or Pivotal tc Server. Support for these distributions is always available through their issue tracker or StackOverflow tag.

- **Pivotal Virtual local-host server:** as a web application is deployed on a server, rather than using a real server that could be very expensive, I used a virtual local-host generated server for deploying the application.

Pivotal Software, Inc. (Pivotal) is a software company based in Palo Alto, California that provides software and services for the development of custom applications for data and analytics based on cloud computing technology. Pivotal Software is a spin-out and joint venture of EMC Corporation and its subsidiary VMware that combined software products, employees, and lines of businesses from the two parent companies including Greenplum, Cloud Foundry, Spring, Pivotal Labs, GemFire and other products from the VMware vFabric Suite.

In July 2012 a GigaOM blog entry speculated on a possible spin out of VMware and EMC that would consolidate some of their cloud computing projects into a new division. The companies confirmed speculation in December of the same year, announcing the initiative with existing technology, people and programs from both companies focused on big data and cloud application platforms under one organization. On April 1, 2013, *The New York Times* reported that Pivotal was official and positioned as a competitor to Amazon Web Services.

At an official event held on April 24, 2013, the organization announced both a \$105 million investment from General Electric and its PaaS offering, PivotalCF, a cloud-enabled application platform for private cloud initiatives and public cloud providers.

Paul Maritz became Pivotal's chief executive officer. Maritz joined EMC in February 2008 when Pi Corporation, a company he co-founded, was acquired. In October 2013 Pivotal acquired consulting firm Xtreme Labs for an estimated \$65 million.

Rob Mee became chief executive officer on August 18, 2015.

- **Oracle Express Edition:** it is a well known database management tool used for back-end database support for any application that requires to save any user data externally and further retrieval of that data even after the application is rebooted.

Oracle Database (commonly referred to as **Oracle RDBMS** or simply as **Oracle**) It is abbreviated as **Oak Ridge Automatic Computer and Logical Engine** and It is an object-relational database management system produced and marketed by Oracle Corporation. Larry Ellison and his two friends and former co-workers, Bob Miner and Ed Oates, started a consultancy called Software Development Laboratories (SDL) in 1977. SDL developed the original version of the Oracle software. The name *Oracle* comes from the code-name of a CIA-funded project Ellison had worked on while previously employed by Ampex.

An Oracle database system—identified by an alphanumeric system identifier or SID—comprises at least one instance of the application, along with data storage. An instance—identified persistently by an instantiation number (or activation id: SYS.V_\$DATABASE.ACTIVATION#)—comprises a set of operating-system processes and memory-structures that interact with the storage. (Typical processes include PMON (the process monitor) and SMON (the system monitor).) Oracle documentation can refer to an active database instance as a "shared memory realm".

Users of Oracle databases refer to the server-side memory-structure as the SGA (System Global Area). The SGA typically holds cache information such as data-buffers, SQL commands, and user information. In addition to storage, the database consists of online redo logs (or logs), which hold transactional history. Processes can in turn archive the online redo logs into archive logs (offline redo logs), which provide the basis (if necessary) for data recovery and for the physical-standby forms of data replication using Oracle Data Guard.

If the Oracle database administrator has implemented Oracle RAC (Real Application Clusters), then multiple instances, usually on different servers, attach to a central storage array. This scenario offers advantages such as better performance, scalability and redundancy. However, support becomes more complex, and many sites do not use RAC. In version 10g, grid computing introduced shared resources where an instance can use (for example) CPU resources from another node (computer) in the grid.

The Oracle DBMS can store and execute stored procedures and functions within itself. PL/SQL (Oracle Corporation's proprietary procedural extension to SQL), or the object-oriented language Java can invoke such code objects and/or provide the programming structures for writing them.

The Oracle RDBMS stores data logically in the form of tablespaces and physically in the form of data files ("datafiles"). Tablespaces can contain various types of memory segments, such as Data Segments, Index Segments, etc. Segments in turn comprise one or more extents. Extents comprise groups of contiguous data blocks. Data blocks form the basic units of data storage.

A DBA can impose maximum quotas on storage per user within each tablespace.

- **Spring Framework:** Spring framework is used to solve a special purpose that is to introduce dependency injection in the project. In this project, spring is used to create a hibernate framework configuration model and then use it in different services provided by the application through auto-wiring that model to those services.

The **Spring Framework** is an application framework and inversion of control container for the Java platform. The framework's core features can be used by any Java application, but there are extensions for building web applications on top of the Java EE platform. Although the framework does not impose any specific programming model, it has become popular in the Java community as an alternative to, replacement for, or even addition to the Enterprise JavaBeans (EJB) model. The Spring Framework is open source.

The first version was written by Rod Johnson, who released the framework with the publication of his book *Expert One-on-One J2EE Design and Development* in October 2002. The framework was first released under the Apache 2.0 license in June 2003. The first milestone release, 1.0, was released in March 2004, with further milestone releases in September 2004 and March 2005. The Spring 1.2.6 framework won a Jolt productivity award and a JAX Innovation Award in 2006. Spring 2.0 was released in October 2006, Spring 2.5 in November 2007, Spring 3.0 in December 2009, Spring 3.1 in December 2011, and Spring 3.2.5 in November 2013. Spring Framework 4.0 was released in December 2013. Notable improvements in Spring 4.0 included support for Java SE 8, Groovy 2, some aspects of Java EE7, and WebSocket.

Spring Framework 4.1.7 was released on 30 June 2015 and was immediately upgraded to the current latest version Spring Framework 4.2.0, which was released on 31 July 2015. It is *"compatible with Java 6, 7 and 8, with a focus on core refinements and modern web capabilities"*.

Release candidate for next version of Spring Framework 4.3 is expected in March 2016. It *"will be the final generation within the general Spring 4 system requirements (Java 6+, Servlet 2.5+), getting prepared for an extended 4.3.x support life until 2019"*.

The Spring Framework includes several modules that provide a range of service:

- Spring Core Container: This is the base module of Spring and provides spring containers (BeanFactory and ApplicationContext).
- Aspect-oriented programming: enables implementing cross-cutting concerns.
- Authentication and authorization: configurable security processes that support a range of standards, protocols, tools and practices via the Spring Security sub-project (formerly Acegi Security System for Spring).
- Convention over configuration: a rapid application development solution for Spring-based enterprise applications is offered in the Spring Roo module
- Data access: working with relational database management systems on the Java platform using JDBC and object-relational mapping tools and with NoSQL databases
- Inversion of control container: configuration of application components and lifecycle management of Java objects, done mainly via dependency injection

- **Messaging:** configurative registration of message listener objects for transparent message-consumption from message queues via JMS, improvement of message sending over standard JMS APIs
- **Model–view–controller:** an HTTP- and servlet-based framework providing hooks for extension and customization for web applications and RESTful Web services.
- **Remote access framework:** configurative RPC-style marshalling of Java objects over networks supporting RMI, CORBA and HTTP-based protocols including Web services (SOAP)
- **Transaction management:** unifies several transaction management APIs and coordinates transactions for Java objects
- **Remote management:** configurative exposure and management of Java objects for local or remote configuration via JMX
- **Testing:** support classes for writing unit tests and integration tests

- **Hibernate Framework:** Hibernate framework is used to efficiently improve the back-end database connectivity. It is most commonly used to create a model-to-table relation that is to connect a custom class model used in the application and create a table in the database with the same configurations as the model itself. It also creates secure connections to the back-end.

Hibernate ORM (Hibernate in short) is an object-relational mapping framework for the Java language, providing a framework for mapping an object-oriented domain model to a traditional relational database. Hibernate solves object-relational impedance mismatch problems by replacing direct persistence-related database accesses with high-level object handling functions. Hibernate is free software that is distributed under the GNU Lesser General Public License.

Hibernate's primary feature is mapping from Java classes to database tables (and from Java data types to SQL data types). Hibernate also provides data query and retrieval facilities. It generates SQL calls and relieves the developer from manual result set handling and object conversion. Applications using Hibernate are portable to supported SQL databases with little performance overhead.

Mapping Java classes to database tables is accomplished through the configuration of an XML file or by using Java Annotations. When using an XML file, Hibernate can generate skeleton source code for the persistence classes. This is unnecessary when annotations are used. Hibernate can use the XML file or the annotations to maintain the database schema.

Facilities to arrange one-to-many and many-to-many relationships between classes are provided. In addition to managing associations between objects, Hibernate can also manage reflexive associations where an object has a one-to-many relationship with other instances of its own type.

Hibernate supports the mapping of custom value types. This makes the following scenarios possible:

- Overriding the default SQL type that Hibernate chooses when mapping a column to a property.
- Mapping Java Enum to columns as if they were regular properties.
- Mapping a single property to multiple columns.

Definition: Objects in a front-end application follow OOP principles, while objects in the back-end follow database normalization principles, resulting in different representation requirements. This problem is called "object-relational impedance mismatch". Mapping is a way of resolving the impedance mismatch problem.

Mapping tells the ORM tool which java class object to store in which database table.

Hibernate provides an SQL inspired language called Hibernate Query Language (HQL) which allows SQL-like queries to be written against Hibernate's data objects. *Criteria Queries* are provided as an object-oriented alternative to HQL. Criteria Query is used to modify the objects and provide the restriction for the objects.

Hibernate provides transparent persistence for Plain Old Java Objects (POJOs). The only strict requirement for a persistent class is a no-argument constructor, not necessarily *public*. Proper behavior in some applications also requires special attention to the *equals()* and *hashCode()* methods.

Collections of data objects are typically stored in Java collection objects such as Set and List. Java generics, introduced in Java 5, are supported. Hibernate can be configured to lazy load associated collections. Lazy loading is the default as of Hibernate 3.

Related objects can be configured to *cascade* operations from one to the other. For example, a parent Album object can be configured to cascade its save and/or delete operation to its child Track objects. This can reduce development time and ensure referential integrity. A *dirty checking* feature avoids unnecessary database write actions by performing SQL updates only on the modified fields of persistent objects.

- **Maven repository:** maven is used for automatically generating a standard project directory structure for a vast amount of varying applications and automate the process of adding dependencies for those projects. As we provide the details for any dependency that we want to be added to the project, maven finds those dependencies online and import them to our projects.

Maven is a build automation tool used primarily for Java projects. The word *maven* means 'accumulator of knowledge' in Yiddish. Maven addresses two aspects of building software: First, it describes how software is built, and second, it describes its dependencies. Contrary to preceding tools like Apache Ant, it uses conventions for the build procedure, and only exceptions need to be written down. An XML file describes the software project being built, its dependencies on other external modules and components, the build order, directories, and required plug-ins. It comes with pre-defined targets for performing certain well-defined tasks such as compilation of code and its packaging. Maven dynamically downloads Java libraries and Maven plug-ins from one or more repositories such as the Maven 2 Central Repository, and stores them in a local cache. This local cache of downloaded artifacts can also be updated with artifacts created by local projects. Public repositories can also be updated.

Maven can also be used to build and manage projects written in C#, Ruby, Scala, and other languages. The Maven project is hosted by the Apache Software Foundation, where it was formerly part of the Jakarta Project.

Maven is built using a plugin-based architecture that allows it to make use of any application controllable through standard input. Theoretically, this would allow anyone to write plugins to interface with build tools (compilers, unit test tools, etc.) for any other language. In reality, support and use for languages other than Java has been minimal. Currently a plugin for the .NET framework exists and is maintained, and a C/C++ native plugin is maintained for Maven 2.

Alternative technologies like gradle and sbt as build tools do not rely on XML, but keep the key concepts Maven introduced. With Apache Ivy, a dedicated dependency manager was developed as well that also supports Maven repositories.

A central feature in Maven is dependency management. Maven's dependency-handling mechanism is organized around a coordinate system identifying individual artifacts such as software libraries or modules. The POM example above references the JUnit coordinates as a direct dependency of the project. A project that needs, say, the Hibernate library simply has to declare Hibernate's project coordinates in its POM. Maven will automatically download the dependency and the dependencies that Hibernate itself needs (called transitive dependencies) and store them in the user's local repository. Maven 2 Central Repository is used by default to search for libraries, but one can configure the repositories to be used (e.g., company-private repositories) within the POM.

There are search engines such as The Central Repository Search Engine which can be used to find out coordinates for different open-source libraries and frameworks.

Projects developed on a single machine can depend on each other through the local repository. The local repository is a simple folder structure that acts both as a cache for downloaded dependencies and as a centralized storage place for locally built artifacts. The Maven command `mvn install` builds a project and places its binaries in the local repository. Then other projects can utilize this project by specifying its coordinates in their POMs.

- **JAVA-J2EE programming language:** the heart of this project lies with JAVA, it is a very trending programming language, the most common use of JAVA includes web applications and Android platform application development. And J2EE include the web application part of JAVA.

Java Platform, Enterprise Edition or **Java EE** is a widely used enterprise computing platform developed under the Java Community Process. 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. Java EE extends the Java Platform, Standard Edition (Java SE), providing an API for object-relational mapping, distributed and multi-tier architectures, and web services. The platform incorporates a design based largely on modular components running on an application server. Software for Java EE is primarily developed in the Java programming language. The platform emphasizes convention over configuration and annotations for configuration. Optionally XML can be used to override annotations or to deviate from the platform defaults.

The platform was known as *Java 2 Platform, Enterprise Edition* or *J2EE* until the name was changed to *Java Platform, Enterprise Edition* or *Java EE* in version 5. The current version is called *Java EE 7*.

- J2EE 1.2 (December 12, 1999)
- J2EE 1.3 (September 24, 2001)
- J2EE 1.4 (November 11, 2003)
- Java EE 5 (May 11, 2006)
- Java EE 6 (December 10, 2009)
- Java EE 7 (May 28, 2013, but April 5, 2013 according to spec document. June 12, 2013 was the planned kickoff date)
- Java EE 8 (expected first half of 2017)

Java EE is defined by its specification. As with other Java Community Process specifications, providers must meet certain conformance requirements in order to declare their products as *Java EE compliant*.

Java EE includes several API specifications, such as RMI, e-mail, JMS, web services, XML, etc., and defines how to coordinate them. Java EE also features some specifications unique to Java EE for components. These include Enterprise JavaBeans, connectors, servlets, JavaServer Pages and several web service technologies. This allows developers to create enterprise applications that are portable and scalable, and that integrate with legacy technologies. A Java EE application server can handle transactions, security, scalability, concurrency and management of the components it is deploying, in order to enable developers to concentrate more on the business logic of the components rather than on infrastructure and integration tasks.

- **HTML:** html is used to create static web page layouts.

HyperText Markup Language, commonly referred to as **HTML**, is the standard markup language used to create web pages. Web browsers can read HTML files and render them into visible or audible web pages. HTML describes the structure of a website semantically along with cues for presentation, making it a markup language, rather than a programming language. HTML elements form the building blocks of all websites. HTML allows images and objects to be embedded and can be used to create interactive forms. It provides a means to create structured documents by denoting structural semantics for text such as headings, paragraphs, lists, links, quotes and other items.

The language is written in the form of HTML elements consisting of *tags* enclosed in angle brackets (like `<html>`). Browsers do not display the HTML tags and scripts, but use them to interpret the content of the page.

HTML can embed scripts written in languages such as JavaScript which affect the behavior of HTML web pages. Web browsers can also refer to Cascading Style Sheets (CSS) to define the look and layout of text and other material. The World Wide Web Consortium (W3C), maintainer of both the HTML and the CSS standards, has encouraged the use of CSS over explicit presentational HTML since 1997.

In 1980, physicist Tim Berners-Lee, who was a contractor at CERN, proposed and prototyped ENQUIRE, a system for CERN researchers to use and share documents. In 1989, Berners-Lee wrote a memo proposing an Internet-based hypertext system. Berners-Lee specified HTML and wrote the browser and server software in late 1990. That year, Berners-Lee and CERN data systems engineer Robert Cailliau collaborated on a joint request for funding, but the project was not formally adopted by CERN. In his personal notes from 1990 he listed "some of the many areas in which hypertext is used" and put an encyclopedia first.

The first publicly available description of HTML was a document called "HTML Tags", first mentioned on the Internet by Berners-Lee in late 1991. It describes 18 elements comprising the initial, relatively simple design of HTML. Except for the hyperlink tag, these were strongly influenced by SGMLguid, an in-house Standard Generalized Markup Language (SGML)-based documentation format at CERN. Eleven of these elements still exist in HTML 4.

HTML is a markup language that web browsers use to interpret and compose text, images and other material into visual or audible web pages. Default characteristics for every item of HTML markup are defined in the browser, and these characteristics can be altered or enhanced by the web page designer's additional use of CSS. Many of the text elements are found in the 1988 ISO technical report TR 9537 *Techniques for using SGML*, which in turn covers the features of early text formatting languages such as that used by the RUNOFF command developed in the early 1960s for the CTSS (Compatible Time-Sharing System) operating system: these formatting commands were derived from the commands used by typesetters to manually format documents. However, the SGML concept of generalized markup is based on elements (nested annotated ranges with attributes) rather than merely print effects, with also the separation of structure and markup; HTML

has been progressively moved in this direction with CSS.

Berners-Lee considered HTML to be an application of SGML. It was formally defined as such by the Internet Engineering Task Force (IETF) with the mid-1993 publication of the first proposal for an HTML specification: "Hypertext Markup Language (HTML)" Internet-Draft by Berners-Lee and Dan Connolly, which included an SGML Document Type Definition to define the grammar. The draft expired after six months, but was notable for its acknowledgment of the NCSA Mosaic browser's custom tag for embedding in-line images, reflecting the IETF's philosophy of basing standards on successful prototypes. Similarly, Dave Raggett's competing Internet-Draft, "HTML+ (Hypertext Markup Format)", from late 1993, suggested standardizing already-implemented features like tables and fill-out forms.

After the HTML and HTML+ drafts expired in early 1994, the IETF created an HTML Working Group, which in 1995 completed "HTML 2.0", the first HTML specification intended to be treated as a standard against which future implementations should be based.

Further development under the auspices of the IETF was stalled by competing interests. Since 1996, the HTML specifications have been maintained, with input from commercial software vendors, by the World Wide Web Consortium (W3C). However, in 2000, HTML also became an international standard (ISO/IEC 15445:2000). HTML 4.01 was published in late 1999, with further errata published through 2001. In 2004 development began on HTML5 in the Web Hypertext Application Technology Working Group (WHATWG), which became a joint deliverable with the W3C in 2008, and completed and standardized on 28 October 2014.

- **CSS:** css is used for styling of the html web page.

Cascading Style Sheets (CSS) is a style sheet language used for describing the look and formatting of a document written in a markup language. Although most often used to change the style of web pages and user interfaces written in HTML and XHTML, the language can be applied to any kind of XML document, including plain XML, SVG and XUL. Along with HTML and JavaScript, CSS is a cornerstone technology used by most websites to create visually engaging webpages, user interfaces for web applications, and user interfaces for many mobile applications. CSS is designed primarily to enable the separation of document content from document presentation, including elements such as the layout, colors, and fonts. This separation can improve content accessibility, provide more flexibility and control in the specification of presentation characteristics, enable multiple HTML pages to share formatting by specifying the relevant CSS in a separate .css file, and reduce complexity and repetition in the structural content, such as semantically insignificant tables that were widely used to format pages before consistent CSS rendering was available in all major browsers. CSS makes it possible to separate presentation instructions from the HTML content in a separate file or style section of the HTML file. For each matching HTML element, it provides a list of formatting instructions. For example, a CSS rule might specify that "all heading 1 elements should be bold", leaving pure semantic HTML markup that asserts "this text is a level 1 heading" without formatting code such as a **<bold>** tag indicating how such text should be displayed.

This separation of formatting and content makes it possible to present the same markup page in different styles for different rendering methods, such as on-screen, in print, by voice (when read out by a speech-based browser or screen reader) and on Braille-based, tactile devices. It can also be used to display the web page differently depending on the screen size or device on which it is being viewed. Although the author of a web page typically links to a CSS file within the markup file, readers can specify a different style sheet, such as a CSS file stored on their own computer, to override the one the author has specified. If the author or the reader did not link the document to a style sheet, the default style of the browser will be applied. Another advantage of CSS is that aesthetic changes to the graphic design of a document (or hundreds of documents) can be applied quickly and easily, by editing a few lines in one file, rather than by a laborious (and thus expensive) process of crawling over every document line by line, changing markup.

The CSS specification describes a priority scheme to determine which style rules apply if more than one rule matches against a particular element. In this so-called *cascade*, priorities (or *weights*) are calculated and assigned to rules, so that the results are predictable.

The CSS specifications are maintained by the World Wide Web Consortium (W3C). Internet media type (MIME type) `text/css` is registered for use with CSS by RFC 2318 (March 1998). The W3C operates a free CSS validation service for CSS documents.

Before CSS, nearly all of the presentational attributes of HTML documents were contained within the HTML markup; all font colors, background styles, element alignments, borders and sizes had to be explicitly described, often repeatedly, within the HTML. CSS allows authors to move much of that information to another file, the style sheet, resulting in considerably simpler HTML.

For example, headings (h1 elements), sub-headings (h2), sub-sub-headings (h3), etc., are defined structurally using HTML. In print and on the screen, choice of font, size, color and emphasis for these elements is *presentational*.

Before CSS, document authors who wanted to assign such typographic characteristics to, say, all h2 headings had to repeat HTML presentational markup for each occurrence of that heading type. This made documents more complex, larger, and more error-prone and difficult to maintain. CSS allows the separation of presentation from structure. CSS can define color, font, text alignment, size, borders, spacing, layout and many other typographic characteristics, and can do so independently for on-screen and printed views. CSS also defines non-visual styles such as the speed and emphasis with which text is read out by aural text readers. The W3C has now deprecated the use of all presentational HTML markup.

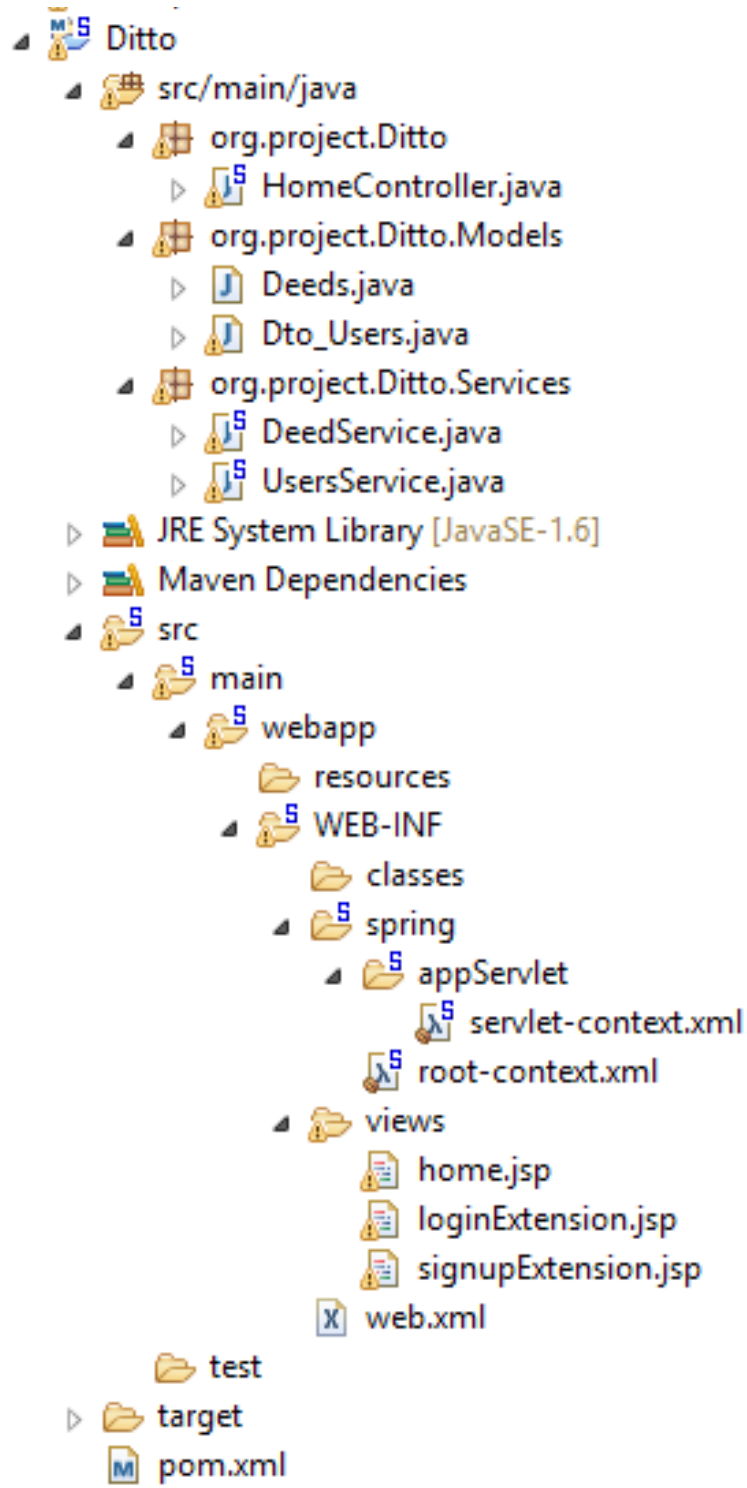
CSS information can be provided from various sources. These sources can be the web browser, the user and the author. The information from the author can be further classified into inline, media type, importance, selector specificity, rule order, inheritance and property definition. CSS style information can be in a separate document or it can be embedded into an HTML document. Multiple style sheets can be imported. Different styles can be applied depending on the output device being used; for example, the screen version can be quite different from the printed version, so that authors can tailor the presentation appropriately for each medium.

The style sheet with the highest priority controls the content display. Declarations not set in the highest priority source are passed on to a source of lower priority, such as the user agent style. This process is called *cascading*.

One of the goals of CSS is to allow users greater control over presentation. Someone who finds red italic headings difficult to read may apply a different style sheet. Depending on the browser and the web site, a user may choose from various style sheets provided by the designers, or may remove all added styles and view the site using the browser's default styling, or may override just the red italic heading style without altering other attributes.

Project Directory Hierarchy

Below is the image of project directories and files in hierarchical presentation.



Design Documentation

Most of the **user interface** introduced in this project are **custom style-sheets** designed by myself. I rejected the idea of using pre-rendered style sheets as the project wasn't huge enough to be worried about implementation of graphical interfaces and style-sheets css most commonly used like Twitter Bootstrap css libraries. I took this initiative for more personnel reasons including learning to create custom style-sheets and to improve the user experience by removing add-on libraries as they tend to over weight the project, they should only be used if necessary.

The whole web application interface is written in **JSP (JAVA Server Page)**, that is an essential part of JAVA web application as it provides the freedom of creating a **dynamically rendered page**, a JSP can get data from the browser and produce a layout as per the requirement and the provided data during runtime, which is the essence of JAVA web application development.

List of **basic elements of html used in JSP** for this project are as:

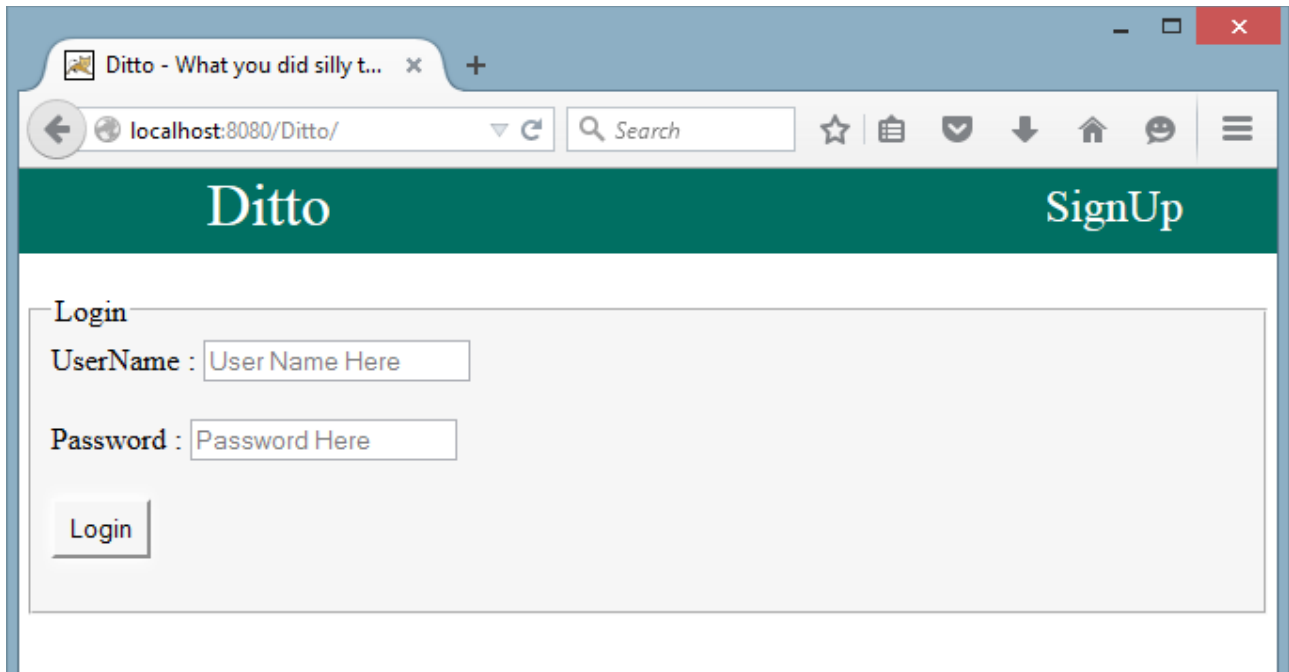
- ☐ Field-Sets
- ☐ Text-Fields
- ☐ Text-Areas
- ☐ Buttons

I have implemented least button clicks needed as much as possible, as it is an essential part of user-to-application interactions. The more clicks and entries that user require to do, the less users revisit. This procedure is well used in the industry for creating habit forming applications. Improving the **overall user experience**.

All the layout implementation is done in a **JSP format as HTML tags**, that is i have used the JSP to dynamically provide the necessary layout of the web application according to user needs, but the actual layout is still hard-coded in HTML only. All the extended custom makeover, or view style-sheets design is written in CSS language.

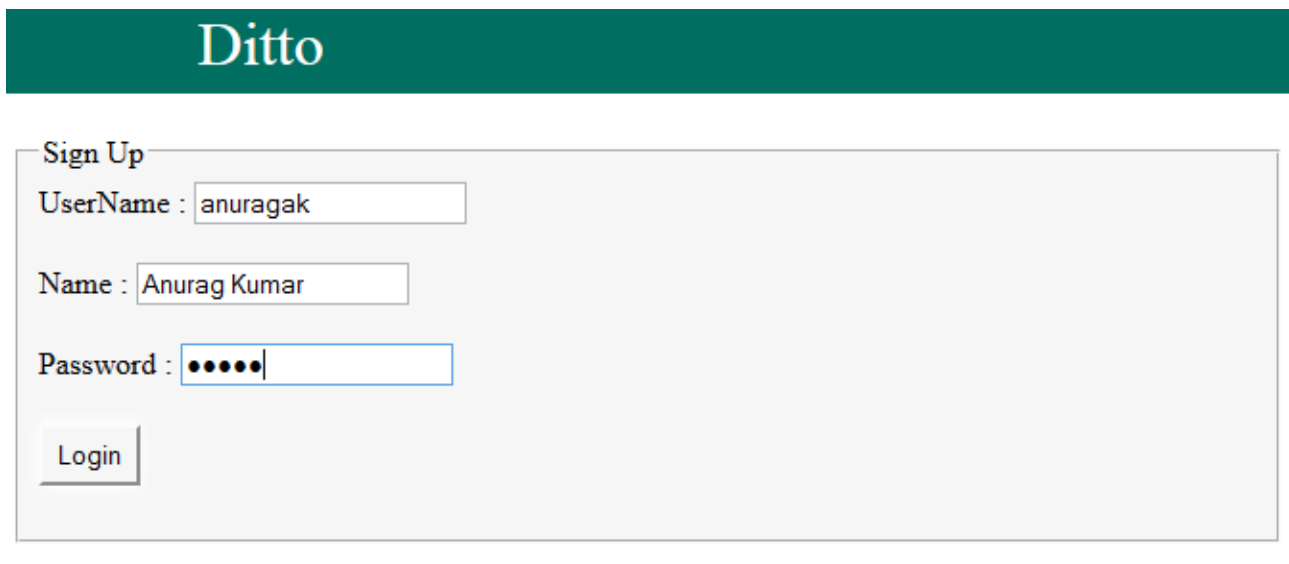
It also include some nice animations like if hover over the visible tile of deed on the page, little extension of that deed pops up below the tile offering the user to view its ditto count, edit or delete the deed and increment the ditto count. This is performed using the hover options of CSS itself, very handy feature of css.

Application Screen-shots



The screenshot shows a web browser window with the address bar displaying 'localhost:8080/Ditto/'. The page has a dark green header with the word 'Ditto' on the left and 'SignUp' on the right. Below the header, there is a 'Login' section. It contains two input fields: 'UserName : User Name Here' and 'Password : Password Here'. A 'Login' button is positioned below these fields.

The above page provides the facility of logging in to the user or signing up if the user doesn't hold any existing account for this application.



The screenshot shows the 'Sign Up' section of the application. It features three input fields: 'UserName : anuragak', 'Name : Anurag Kumar', and 'Password : •••••'. A 'Login' button is located at the bottom left of the sign-up form area.

Sign-Up Page

Here the new users can sign up for an account on the application to access the privileges that only an account holder can access.

Post Deed

Hi , What Silly you did today ?**I, Anurag Kumar, created this web based java project.**

by :Anurag Kumar

[Ditto Me](#)[Edit Deed](#)[Delete Deed](#)

Dittos : 2

Posting Deed Add-on to the user page

In the above page, any logged in user can view all the posts or deeds ever uploaded in the application, and also add a fresh new post to the list for others to see, as the user hovers over the posts, he/she will find the options for the posts visible in the screen-shot for ditto Me, Editing, Deleting the posts that can only be done by the user that created this post. And view the count of Dittos, or the number of times other users performed this deed.

Ditto

Logout

Edit Past Deed

Edit Your Deed

anuragak

I, Anurag Kumar, created this web based java project.

Post

I, Anurag Kumar, created this web based java project.

by :Anurag Kumar

Editing Deed Add-on to the user page

The above page provides the facility of editing any previous deed that the current logged in user posted. And after editing, the new deed will be visible for all the users to acknowledge.

Below is the Github link for the documented project, as I prefer to give a link from where you could also note the improves and version control with full transparency, so rather than providing a hard copy of the project files I provided the Github link here:

Project-Ditto:

<https://github.com/anuragkumarak95/Project-Ditto/>