**ABSTRACT**

The Managed Accounts Portal will be a cutting edge platform for management of advisor-based investment products for custody business. This portal will offer an integrated end-to-end solution including proposal development, creation and maintenance of accounts, and ongoing monitoring and reporting services. The purpose of this project is to have one Hierarchal solution that will support both the existing lines and future scope of business. Character length gaps will need to be fulfilled in order to accommodate clients request. In order to accomplish this, we need to create a common grouping function that will support the existing grouping features and functions that is currently in use. In this project, Security features and User Experience is enhanced by making all the essential features available for the user with easy steps.

**TABLE OF CONTENTS**

|  |  |  |
| --- | --- | --- |
| **CHAPTER NO.** | **TITLE** | **PAGE NO.** |
|  | **ABSTRACT**  **LIST OF FIGURES** |  |
| **1** | **INTRODUCTION** |  |
| **2** | **PROBLEM STATEMENT AND DESCRIPTION** |  |
|  | **2.1 PROBLEM STATEMENT**  **2.2 PROBLEM DESCRIPTION** |  |
| **3** | **PROPOSED DESIGN**  **3.1 AUTHENTICATION**  **3.2 MULTIFACTOR AUTHENTICATION**  **3.3 USER EXPERIENCE**  **3.4 TREE VIEW**  **3.5 SESSION MANAGEMENT** |  |
| **4** | **TOOLS USED** |  |
|  | **4.1 LUCENE SEARCH** |  |
|  | **4.2 SONARQUBE** |  |
|  | **4.3 JENKINS**  **4.4 GIT** |  |
| **5** | **SYSTEM CONFIGURATION** |  |
|  | * 1. **SOFTWARE CONFIGURATION**   **5.2 HARDWARE CONFIGURATION** |  |
| **6** | **CONCLUSION** |  |
|  |  |  |
|  |  |  |

**LIST OF FIGURES**

|  |  |  |
| --- | --- | --- |
| **FIG NO** | **Title** | **PAGE NO.** |
| **3.1** | Flow Diagram For User Authentication | **21** |
| **3.2** | Flow Diagram For OTP Authentication | **22** |
| **3.3** | Screenshot Of Login Screen | **22** |
| **3.4** | Screenshot Of User’s OTP Method Selection | **26** |
| **3.5** | Screenshot Of User’s OTP Verification | **27** |
| **3.6** | Types Of User | **28** |
| **3.7** | Use Case Diagram Depicting User’s Basic Functions | **28** |
| **3.8** | Screenshot Of User’s Dashboard | **29** |
| **3.9** | Screenshot Of User’s Key Searching Dialog Box | **29** |
| **3.10** | Screenshot Of Various Menu Available For User | **30** |
| **3.11** | Screenshot Of Lucene Search | **30** |
| **3.12** | Screenshot Of Reports Generated | **32** |
| **3.13** | Flow Diagram Of Session Management | **32** |

**1. INTRODUCTION**

**1 Introduction:**

**1.1 Managed Accounts:**

In banking, a managed account is a fee-based [investment management](https://en.wikipedia.org/wiki/Investment_management) [product](https://en.wikipedia.org/wiki/Product_(business)) for [high-net-worth individuals](https://en.wikipedia.org/wiki/High-net-worth_individual). The main appeal for wealthy individuals is the access to professional money managers, a high degree of customization and greater tax efficiencies in a fee-based product. They are not to be confused with managed bank accounts such as [thinkmoney](https://en.wikipedia.org/wiki/Thinkmoney" \o "Thinkmoney), e-money accounts and basic bank accounts, all of which are consumer banking products in the UK.

Managed accounts started as [separately managed accounts](https://en.wikipedia.org/wiki/Separately_managed_account) (SMAs) and have since evolved into multiple strategy accounts (MSAs) and the rapidly emerging [unified managed accounts](https://en.wikipedia.org/wiki/Unified_managed_account) (UMAs). There is broad agreement that managed accounts provide the added benefits of greater transparency, liquidity and control.

Managed account minimums and the cost to operate managed account programs have steadily dropped as technology helps with efficiency and scale. Increasingly, managed account products are seeing interest from the "[mass affluent](https://en.wikipedia.org/wiki/Mass_affluent)" as well.

The retail managed accounts industry was sized at $1.70 trillion in 3Q 2009. Managed Accounts are typically offered by global [investment banks](https://en.wikipedia.org/wiki/Investment_bank) and specialist investment firms.

**1.2 Managed Investments:**

Managed investment schemes are also known as 'managed funds', 'pooled investments' or 'collective investments'. Generally in a managed investment scheme. People are brought together to contribute money to get an interest in the scheme. Money is pooled together with other investors (often many hundreds or thousands of investors) or used in a common enterprise.

Managed investment is a 'responsible entity' operates the scheme. Investors do not have day to day control over the operation of the scheme.

**2. PROBLEM STATEMENT AND DESCRIPTION**

**2.1 Problem Statement:**

Security features and User experience are the two major factors deciding the success of an application. This project deals with improving the security features and enhancing the experience of the user generally the client while using the application.

**2.2 Problem Definition:**

Security features of an application such as authenticating the user who logs in, granting specific permissions to access the data based on the type of user.

Enhancing User experience by showing only the results and data what the user wants to know exactly.

**3. OVERVIEW OF PROPOSED SYSTEM**

**3.1 Authentication:**

The process of identifying an individual, usually based on a [username](http://www.webopedia.com/TERM/U/username.html) and [password](http://www.webopedia.com/TERM/P/password.html). In [security systems](http://www.webopedia.com/TERM/S/security.html), authentication is distinct from [*authorization*](http://www.webopedia.com/TERM/A/authorization.html) *,* which is the process of giving individuals [access](http://www.webopedia.com/TERM/A/access.html) to system objects based on their [identity](http://www.webopedia.com/TERM/I/identity.html). Authentication merely ensures that the individual is who he or she claims to be, but says nothing about the access rights of the individual.

The General flow diagram is as follows.



**Figure 3.1** Flow Diagram for user authentication

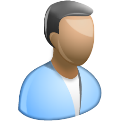
**3.2 Multi Factor Authentication:**

The user who logs in is actually authenticated by any of the following three methods. User can choose,

* + - * Phone Call
      * Text Message
      * E-Mail

User can choose any one method and OTP (One Time Password) for that login session is sent to the user on registered number or mail address. On validating the OTP sent and user entered code, user is authenticated.

Following usecase diagram shows the flow diagram of how user is being authenticated using OTP.



Invalid

Retry Logging in

ValidOTP Passcode

Valid credentials

Valid

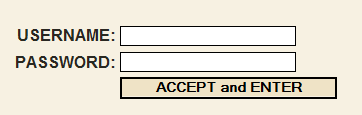
Create Session

**Figure 3.2** Flow Diagram for OTP Authentication

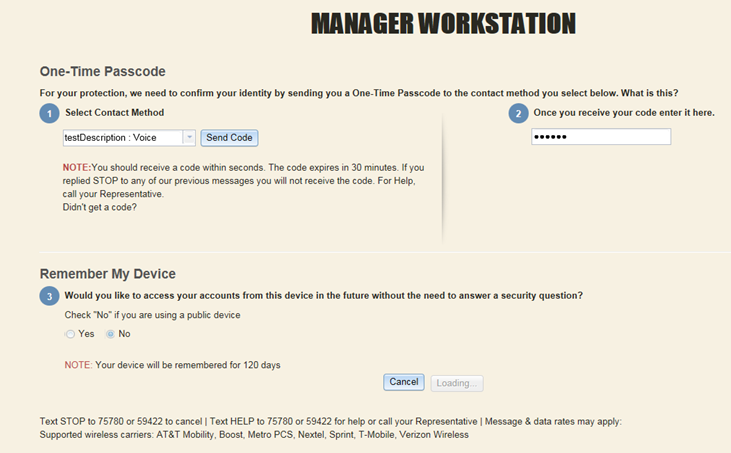
Criteria involved:

* User can enter incorrect OTP only thrice, after which the account will be locked for 24 hours.
* User can reset password for the account once in 24 hours only.
* All incorrect attempts are being logged.

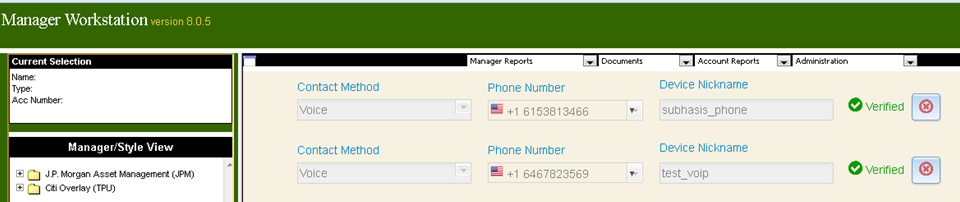
The following screens shows how the user enters valid passcode to log n into the application.



**Figure 3.3** Screenshot of Login Screen



**Figure 3.4** Screenshot of User’s OTP method selection



**Figure 3.5** Screenshot of User’s OTP verification

**3.3 User Experience:**

The first requirement for an exemplary user experience is to meet the exact needs of the customer, without fuss or bother. Next comes simplicity and elegance that produce products that are a joy to own, a joy to use. True user experience goes far beyond giving customers what they say they want, or providing checklist features. In order to achieve high-quality user experience in a company's offerings there must be a seamless merging of the services of multiple disciplines, including engineering, marketing, graphical and industrial design, and interface design.

It's important to distinguish the total user experience from the user interface (UI), even though the UI is obviously an extremely important part of the design. As an example, consider a website with movie reviews. Even if the UI for finding a film is perfect, the UX will be poor for a user who wants information about a small independent release if the underlying database only contains movies from the major studios.



**Figure 3.6** Types of User



**Figure 3.7** Use case Diagram depicting User’s basic functions

**3.4 Tree View**

**CLIENT**

**………….**

**ACCOUNT N**

**ACCOUNT 3**

**ACCOUNT 2**

**ACCOUNT 1**

**GROUP 3 / HOUSEHOLD 3**

**GROUP 2 / HOUSEHOLD 2**

**GROUP 1 / HOUSEHOLD 1**

**ADVISOR 2**

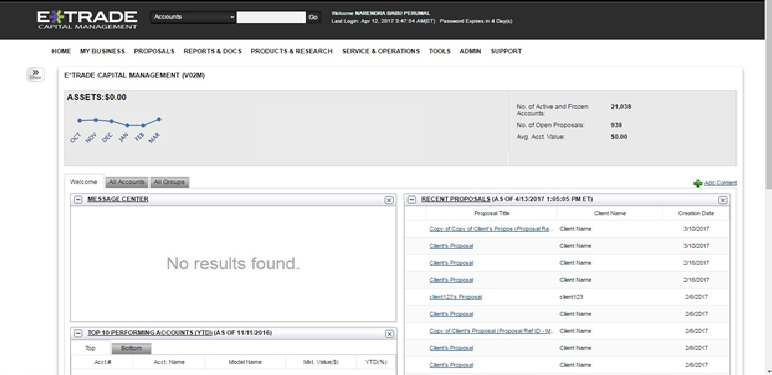
**ADVISOR 1**

**REGION 2**

**REGION 1**

**FIRM**

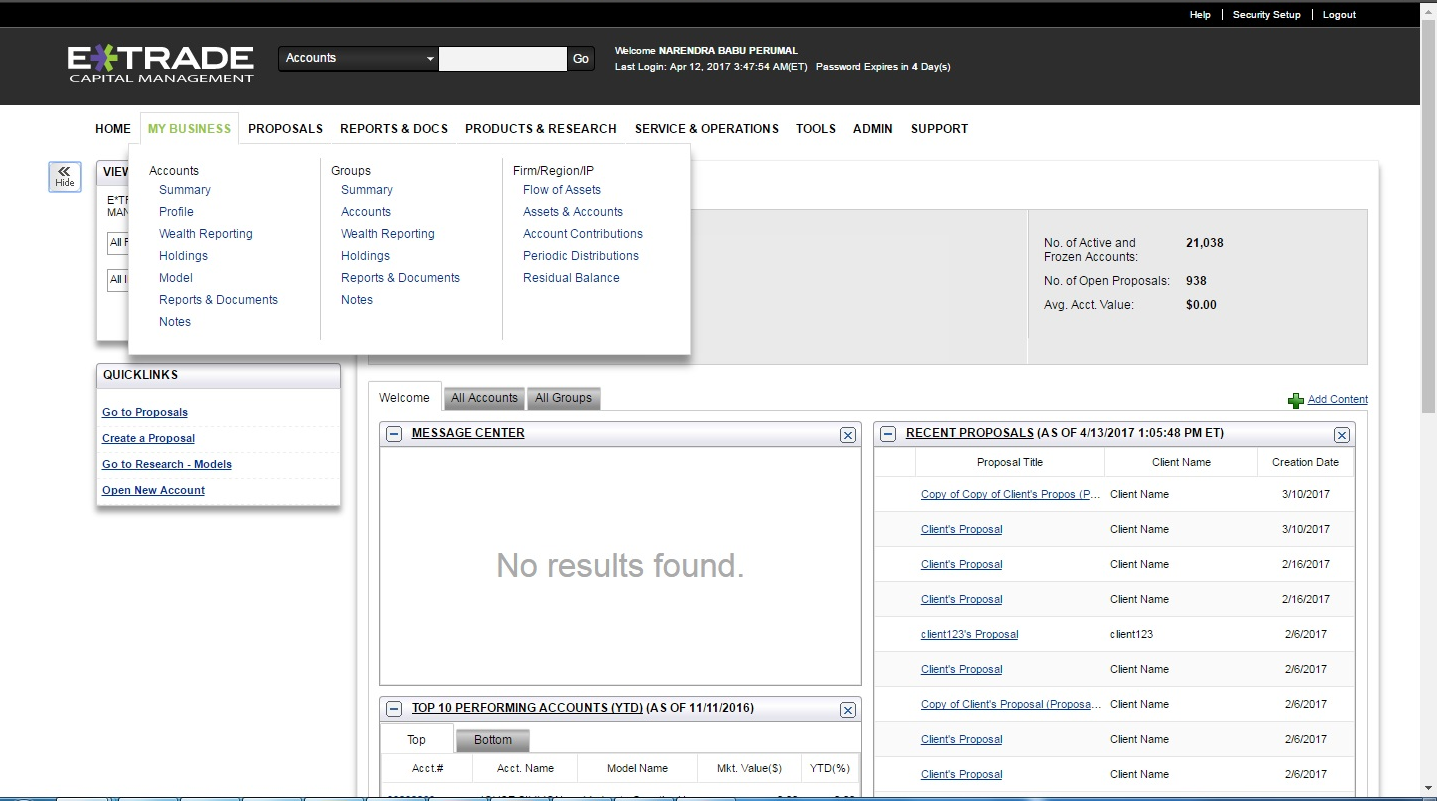
**Figure 3.8** Tree View of client’s account &Grouping



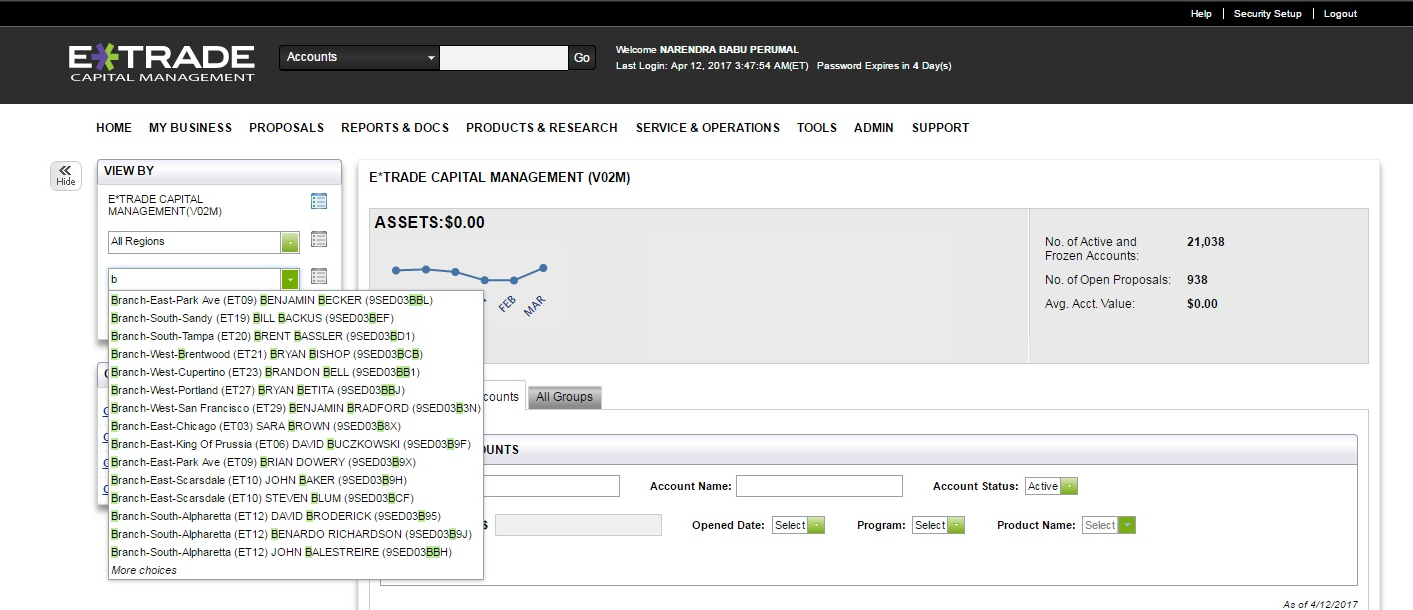
**Figure 3.9** Screenshot of User’s Dashboard



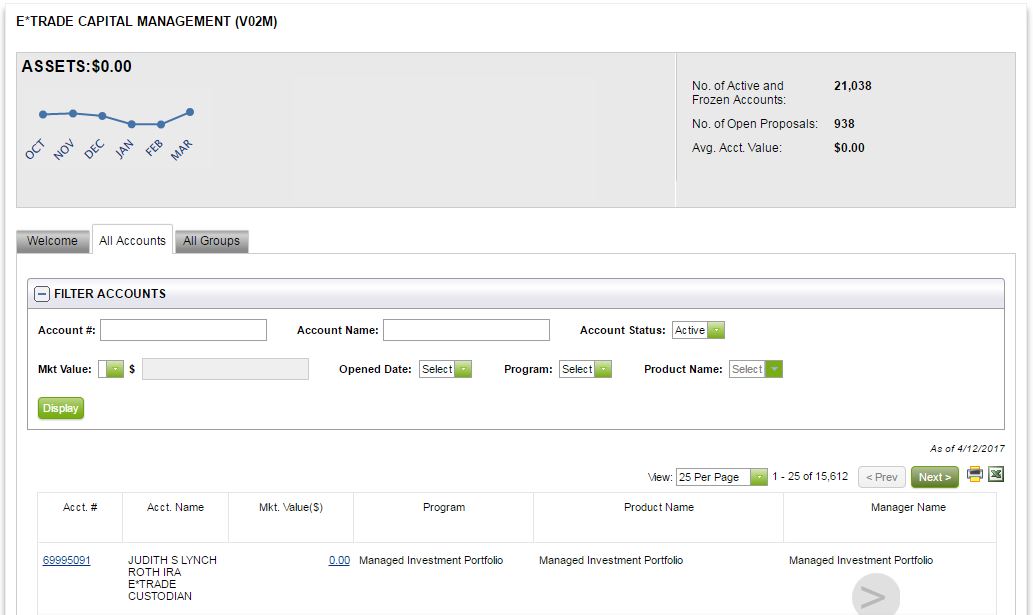
**Figure 3.10** Screenshot of User’s Key searching dialog box



**Figure 3.11** Screenshot of Various menu available for User

****

**Figure 3.12** Screenshot of Lucene search integrated with search box

****

**Figure 3.13** Screenshot of Reports generated

**3.5 Session Management:**

A web session is a sequence of network HTTP request and response transactions associated to the same user. Modern and complex web applications require the retaining of information or status about each user for the duration of multiple requests. Therefore, sessions provide the ability to establish variables – such as access rights and localization settings – which will apply to each and every interaction a user has with the web application for the duration of the session.

Web applications can create sessions to keep track of anonymous users after the very first user request. An example would be maintaining the user language preference. Additionally, web applications will make use of sessions once the user has authenticated. This ensures the ability to identify the user on any subsequent requests as well as being able to apply security access controls, authorized access to the user private data, and to increase the usability of the application. Therefore, current web applications can provide session capabilities both pre and post authentication.

Once an authenticated session has been established, the session ID (or token) is temporarily equivalent to the strongest authentication method used by the application, such as username and password, passphrases, one-time passwords (OTP), client-based digital certificates, smartcards, or biometrics (such as fingerprint or eye retina).

**Session Id**

**(Initiated)**

**Authentication**

**Request**

**User Interaction**

**Logout**

**Session closed**

**Figure 3.13** Flow Diagram of Session management

Session management mechanisms based on cookies can make use of two types of cookies, non-persistent (or session) cookies, and persistent cookies. If a cookie presents the “Max-Age” (that has preference over “Expires”) or “Expires” attributes, it will be considered a persistent cookie and will be stored on disk by the web browser based until the expiration time. Typically, session management capabilities to track users after authentication make use of non-persistent cookies. This forces the session to disappear from the client if the current web browser instance is closed. Therefore, it is highly recommended to use non-persistent cookies for session management purposes, so that the session ID does not remain on the web client cache for long periods of time, from where an attacker can obtain it.

In this project, Session for a user is available maximum for 15 hours. If a user is idle for about 20 minutes popup will be shown and session gets invalidated.

**4. TOOLS USED**

**4.1 Lucene Search:**

Lucene is an open source java based search library. Lucene is very popular and fast search library used in java based application to add document search capability to any kind of application in a very simple and efficient way.

Lucene is simple yet powerful java based search library. It can be used in any application to add search capability to it. Lucene is open-source project. It is scalable and high-performance library used to index and search virtually any kind of text. Lucene library provides the core operations which are required by any search application. Indexing and Searching.

1. **Acquire Raw Content**
2. **Build the document**
3. **Analyze the document**
4. **Indexing the document**
5. **User Interface for Search**
6. **Build Query**
7. **Search Query**
8. **Render Results**

Lucene plays role in steps 2 to step 7 mentioned above and provides classes to do the required operations. In nutshell, lucene works as a heart of any search application and provides the vital operations pertaining to indexing and searching. Acquiring contents and displaying the results is left for the application part to handle.

**4.2 Sonarqube**

Software quality measurement is a quantitative process summing up weighted attribute values, which in part describe specific software characteristics. For each characteristic, a set of such measurable attributes is defined.

Now the question is, what are software characteristics?  Well it could be:

* Whether  the coding has been done following a specific convention
* Whether well-known/established good practices have been followed and well-known/established bad practices have been avoided
* Are there any potential bugs and performance issues, security vulnerabilities
* Is there any duplicate code
* Is the code logic very complex
* Whether the public API has good documentation and comments
* Whether the code has unit tests
* Whether the code follows good design and architecture principles

  SonarQube collects and analyzes source code, measuring quality and providing reports for your projects. It combines static and dynamic analysis tools and enables quality to be measured continuously over time.  Everything that affects our code base, from minor styling details to critical design errors, is inspected and evaluated by SonarQube, thereby enabling developers to access and track code analysis data ranging from styling errors, potential bugs, and code defects to design inefficiencies, code duplication, lack of test coverage, and excess complexity. The Sonar platform analyzes source code from different aspects and hence it drills down to your code layer by layer, moving from the module level down to the class level. At each level, SonarQube produces metric values and statistics, revealing problematic areas in the source that require inspection or improvement.

### Features:

* SonarQube doesn't just show you what's wrong. It also offers quality-management tools to actively help you put it right
* SonarQube's commercial competitors seem to focus their definition of quality mainly on bugs and complexity, whereas SonarQube's offerings span what its creators call the Seven Axes of Quality
* SonarQube addresses not just bugs but also coding rules, test coverage, duplications, API documentation, complexity, and architecture, providing all these details in a dashboard
* It gives you a moment-in-time snapshot of your code quality today, as well as trends of lagging (what's already gone wrong) and leading (what's likely to go wrong in the future) quality indicators
* It provides you with metrics to help you take right decision. In nearly every industry, serious leaders track metrics. Whether it's manufacturing defects and waste, sales and revenue, or baseball hits and RBIs, there are metrics that tell you how you're doing: if you're doing well overall, or whether you're getting better or worse.

**4.3 Jenkins**

* Jenkins is an [award-winning](https://wiki.jenkins-ci.org/display/JENKINS/Awards), cross-platform, continuous integration and continuous delivery application that increases your productivity. Use Jenkins to build and test your software projects continuously making it easier for developers to integrate changes to the project, and making it easier for users to obtain a fresh build. It also allows you to continuously deliver your software by providing powerful ways to define your build pipelines and integrating with a large number of testing and deployment technologies.
* Jenkins is a software that allows continuous integration. Jenkins will be installed on a server where the central build will take place. The following flowchart demonstrates a very simple workflow of how Jenkins works.

**JENKINS**

Source code

Triggers Build & Runs test

On Success, Developer gets an automated message

* Along with Jenkins, sometimes, one might also see the association of Hudson. Hudson is a very popular open-source Java-based continuous integration tool developed by Sun Microsystems which was later acquired by Oracle.

What is Continuous Integration?

* Continuous Integration is a development practice that requires developers to integrate code into a shared repository at regular intervals. This concept was meant to remove the problem of finding later occurrence of issues in the build lifecycle. Continuous integration requires the developers to have frequent builds. The common practice is that whenever a code commit occurs, a build should be triggered.

Features:

Jenkins offers the following major features out of the box, and many more can be added through [plugins](https://wiki.jenkins-ci.org/display/JENKINS/Plugins):

* Easy installation: Just run java -jar jenkins.war, deploy it in a servlet container. No additional install, no database. Prefer an installer or native package? We have those as well.
* Easy configuration: Jenkins can be configured entirely from its friendly web GUI with extensive on-the-fly error checks and inline help.
* Rich plugin ecosystem: Jenkins integrates with virtually every SCM or build tool that exists. [View plugins](https://wiki.jenkins-ci.org/display/JENKINS/Plugins).
* Extensibility: Most parts of Jenkins can be extended and modified, and it's easy to create new Jenkins plugins. This allows you to customize Jenkins to your needs.
* Distributed builds: Jenkins can distribute build/test loads to multiple computers with different operating systems. Building software for OS X, Linux, and Windows? No problem.

**4.4 GIT**

Git is a [version control system](https://en.wikipedia.org/wiki/Version_control_system) (VCS) for tracking changes in [computer files](https://en.wikipedia.org/wiki/Computer_file) and coordinating work on those files among multiple people. It is primarily used for [software development](https://en.wikipedia.org/wiki/Software_development),but it can be used to keep track of changes in any files. As a [distributed revision control](https://en.wikipedia.org/wiki/Distributed_revision_control) system it is aimed at speed,data integrity,and support for distributed, non-linear workflows.Git is currently the most popular implementation of a distributed version control system.

The core of Git was originally written in the programming language \_C, but Git has also been re-implemented in other languages, e.g., Java, Ruby and Python.

A Git repository contains the history of a collection of files starting from a certain directory. The process of copying an existing Git repository via the Git tooling is called cloning. After cloning a repository the user has the complete repository with its history on his local machine. Of course, Git also supports the creation of new repositories.

**5. SYSTEM CONFIGURATION**

**5.1 Sofware Requirements**

App Server : Tomcat

Database Server : MS SQL Server 2008

Language : Java

UI & Middleware : Spring, REST Web services, JSF and HTML

Operating System : Platform Independent

**5.2 Hardware Configuration**

Processor : Intel 5

RAM : 4 GB (Minimum)

* 1. **CONCLUSION**

**6.1 Conclusion**

**//Conclusion & future work has to be drafted**