

**PROJECT REPORT  
ON  
Build Automation Tool  
FOR  
Persistent Systems Limited**

**BY  
Aishwarya Shedji**

**UNIVERSITY OF PUNE  
MASTER OF COMPUTER APPLICATIONS  
M.E.S's  
INSTITUTE OF MANGEMENT AND CAREER COURSES  
(IMCC), PUNE-411029  
2012-13**



May 3, 2013

**To Whomsoever it May Concern**

This is to certify that **Aieshwarya Shedji** is undergoing industrial training at Persistent Systems Ltd. Pune under the guidance of Mr. **Rajan Patil**

The details are as follows: -

Duration: 09<sup>th</sup> January 2013 to 05<sup>th</sup> July 2013

Project Title: **"Build-Automation Tool"**

Work Details:

**For Persistent Systems Limited**

A handwritten signature in black ink, appearing to read 'Kaustubh Bhadbhade', is written over a horizontal line.

**Kaustubh Bhadbhade**  
**Senior Manager-Human Resources**

## **ACKNOWLEDGEMENT**

I wish to express deep sense of gratitude towards “**Persistent System Ltd**” for providing me the opportunity to work with the company and to provide a wonderful and challenging environment for the development of “Build Automation Tool”.

I am thankful to **Dr. V.H.INAMDAR** Director, IMCC for his valuable guidance. His keen interest in this project and constant and timely encouragement with affectionate attitude have inspired me a lot to finish my work successfully.

I would like to thank **Dr. Santosh Deshpande HOD, Comp dept, Mrs. Jayeshree Patil** and **Mr. Praful Jadhav** for their valuable guidance and kind co-operation throughout the period of work undertaken.

Last but not least, I am thankful to whose names are not mentioned here but directly or indirectly helped me in completion of this project.

Thank you all.

Miss. Aieshwarya Shedji

MCA-III (Sem-VI)

## INDEX

<b>SR. NO.</b>	<b>CHAPTER NAME</b>	<b>PAGE NO.</b>
	<b>CHAPTER 1: INTRODUCTION</b>	
1.1	Company Profile	1
1.2	Existing System and Need for System	4
1.3	Scope of Work	7
1.4	Operating Environment: Hardware & Software	9
1.5	Detail Description of Technology Used	11
	<b>CHAPTER 2: PROPOSED SYSTEM</b>	
2.1	Proposed System	20
2.2	Objective of System	22
2.3	User Requirements	24
	<b>CHAPTER 3: ANALYSIS &amp; DESIGN</b>	
3.1	Object Diagram	25
3.2	Class Diagram	26
3.3	Use Case Diagram	28
3.4	Activity Diagram	32
3.5	Sequence diagram	36
3.6	Entity Relationship Diagram(ERD)	39
3.7	Module Hierarchy Diagram	41
3.8	Component Diagram	42
3.9	Deployment Diagram	43
3.10	Module Specification	44

3.11	User Interface Design	48
3.12	Table Specification	55
3.13	Test Procedures & Implementation	58
	<b>CHAPTER 4: USER MANUAL</b>	
4.1	User Manual	66
4.2	Operation Manual/Menu Explanation	76
4.3	Program Specification	82
	<b>Drawbacks &amp; Limitations</b>	85
	<b>Proposed Enhancement</b>	86
	<b>Conclusions</b>	87
	<b>Bibliography</b>	
	<b>ANNEXURES</b>	
	ANNEXURES 1: User Interface Screens	
	ANNEXURES 2: Output Reports with Data	
	ANNEXURES 3: Sample Source Code	

**CHAPTER 1**  
**INTRODUCTION**

## **1.1 Company Profile**

### **1.1.1 Company Information**

Established in 1990, Persistent is a global company specializing in software products and technology services. For more than two decades, Persistent has been an innovation partner for world's largest technology brands, leading enterprises and pioneering start-ups. With a global team of 7,000+ employees, Persistent has 300+ customers spread across North America, Europe and Asia.

### **1.1.2 Core Competencies**

Today, Persistent focuses on best-in-class solution in four key next generation technology areas: Cloud Computing, Analytics, Mobility and Collaboration, for telecommunication, life sciences, consumer packaged goods, banking & financial services and health care verticals.

### **1.1.3 Persistent Foundation**

Persistent has been contributing to local and regional Health and Education institutions since 1995. In 2009, the Persistent Foundation was formally established as a public charitable trust to aid in the company's charitable activities. Persistent earmarks 1% of net profit for Persistent Foundation programs.

Persistent Foundation is primarily involved in the following three key areas:

- Healthcare
- Education
- Community Development



### **1.1.4 Persistent Services**

Persistent has created a professional services practice based on its well established product knowledge, deep technical expertise and experience working with product teams.

Persistent delivers integrated end to end solutions and services for our professional services customers. In order to deliver best in class solutions we leverage our skills and expertise across major industry domains as well as IT infrastructure. This helps our clients make their businesses more efficient, agile and responsive to their customers' needs.

#### **Persistent provides professional services in several key areas:**

- Security
- Big Data
- BI & Analytics
- Cloud
- Mobility

## 1.2 Existing System and Need for System

### 1.2.1 Existing System

➤ Today we have to do following tasks manually which is time consuming:

1. Compiling computer source code into binary code.
2. Packaging binary code.
3. Running tests.
4. Deployment to production systems.
5. Creating documentation and/or release notes.

If we have to build or execute applications continuously then all the above steps needs to be repeated manually hence it is time consuming.

➤ The environment where we develop project on different of IDEs (e.g. Eclipse), create .war file as per the IDEs default path which bundles all files that are related in project like .class,/java or other resources used.

## BUILD AUTOMATION TOOL

- While upgrading application user needs to give parameter values as per user requirements and then actual upgradation starts.

### 1.2.2 Need for System

- In recent years, build management solutions have provided even more relief when it comes to automating the build process. Both commercial and open source solutions are available to perform more automated build and workflow processing. These tools are particularly useful for continuous integration builds where frequent calls to the compile process are required and incremental build processing is needed.
- In the field of computer software, the term software build refers either to the process of converting source code files into standalone software artifact(s) that can be run on a computer, or the result of doing so. One of the most important steps of a software build is the compilation process where source code files are converted into executable code.

## BUILD AUTOMATION TOOL

- In software versioning, the build number is often used as a versioning identifier subordinate to, and more finely graded than, the version number. While for simple programs the process consists of a single file being compiled, for complex software the source code may consist of many files and may be combined in different ways to produce many different versions.
- While upgrading the application user needs to get all the default paths to get populated before upgradation of application start. This will help him/her to change the default path at run time as per user's needs.

### 1.3 Scope of System

- Build Automation Process is a project which is about automating the process of build creation. Build creation process includes installing application on JBoss/TomCat, creating/upgrading DB schemas. For this build process creating generic automation framework.
- Build Automation is build integration tool which involves easy installation of database schemas and configuration of production systems along with the build files which needs to be executed on those systems. (i.e. distributed builds).It also supports easy deployment of fresh setup and upgrade setup on existing platform.
- In this tool user needs to send request to admin for its approval to executes different job and see the test result.
- User can either execute the jobs of fresh installation or upgrade the application to next version.

## BUILD AUTOMATION TOOL

- User can also use this tool to execute jobs of different application and get its jar, war or ear and can update the same to next version. Files likes jar, war or ear can be created anywhere as per need of user.
- User can send email notification if build failed to any users by just one single click.
- User can act as a super-admin if he gets a privilege from super-admin as create user.
- Users from different department can use this tool to execute jobs provided they should have necessary privileges from super-admin.
- Super-admin needs to accept user by giving different privileges to different users so that they can login to the tool and execute the jobs automatically.
- Super-admin can view various reports like user's request status, different privileges given to different users, department wise reports etc.

## **1.4 Operating Environment: Hardware and Software**

### **➤ Server side**

#### **1.4.1 Hardware**

1. Operating System : Windows 7
2. RAM : 4 GB
3. Hard Drive : 250 GB

#### **1.4.2 Software**

1. Java 7
2. Ant
3. Jenkins
4. Internet Explorer

### **➤ Client Side**

#### **1.4.1 Hardware**

1. Operating System : Windows 7
2. RAM : 1 GB
3. Hard Drive : 120 GB

## BUILD AUTOMATION TOOL

### 1.4.2 Software

1. Eclipse-Ganymede
2. Java 6
3. Ant
4. Jenkins
5. Internet Explorer



## 1.5 Detailed Description of Technology

### 1.5.1 ANT

- Ant is an abbreviation for Another Neat Tool. Ant as a build tool to compile Java code, pack this code into an executable jar and to create javadoc.
- A Java build process typically includes:
  1. The compilation of the Java source code into Java byte code.
  2. Creation of the .jar file for the distribution of the code.
  3. Creation of the Javadoc documentation.
- Ant uses an xml file for its configuration. This file is usually called build.xml. Ant builds are based on three blocks: tasks, targets and extension points.
- A task is a unit of work which should be performed and are small, atomic steps, for example compile source code or create Javadoc. Tasks can be grouped into targets.

## BUILD AUTOMATION TOOL

- A target can be directly invoked via Ant. Targets can specify their dependencies. Ant will automatically execute dependent targets.
- In your build.xml file you can specify the default target. Ant will execute this target, if no explicit target is specified.

### 1.5.2 Jenkins

- Jenkins is an open source continuous integration tool written in Java. Jenkins provides continuous integration services for software development. It is a server-based system running in a servlet container such as Apache Tomcat. It supports SCM tools including CVS, Subversion and can execute Apache ANT based project as well as arbitrary shell scripts and Windows batch commands.
- Jenkins can monitor CM (Configuration Management) system to detect a check-in. Upon recognition of this change, Jenkins will update a local working directory of code and perform series of build steps (e.g. “ant clean”, “ant

compile” or “make clean” and “make”). Unit tests and tests can be performed after each build.

### 1.5.3 Oracle 11g

- An Oracle **database** is a collection of data treated as a unit. The purpose of a database is to store and retrieve related information. A database server is the key to solving the problems of information management. For e.g. a server reliably manages a large amount of data in a multiuser environment so that many users can concurrently access the same data. All this is accomplished while delivering high performance. A database server also prevents unauthorized access and provides efficient solutions for failure recovery.
- The database has logical structures and physical structures. Because the physical and logical structures are separate, the physical storage of data can be managed without affecting the access to logical storage structures.

## BUILD AUTOMATION TOOL

- Oracle stores data logically in tablespaces and physically in datafiles associated with the corresponding tablespace.

### **Utilities of Oracle**

- Oracle Data Pump technology enables very high-speed movement of data and metadata from one database to another. This technology is the basis for Oracle's data movement utilities, Data Pump Export and Data Pump Import.
- Data Pump enables you to specify whether a job should move a subset of the data and metadata. This is done using data filters and metadata filters, which are implemented through Export and Import parameters.
- Data Pump Export is a utility for unloading data and metadata into a set of operating system files called a dump file set. The dump file set can be moved to another system and loaded by the Data Pump Import utility.
- Data Pump Import is a utility for loading an export dump file set into a target system. The dump file set is made up of one

## BUILD AUTOMATION TOOL

or more disk files that contain table data, database object metadata, and control information. The files are written in a proprietary, binary format.

### 1.5.4 JBOSS Server

- JBOSS stands for JavaBeans Open Source Software Application Server.
- JBOSS is an application server that implements the Java Platform, Enterprise Edition (Java EE).
- JBOSS is written in Java and as such is cross-platform: usable on any operating system that supports Java.

### 1.5.5 Core Java

- Java programming language was originally developed by Sun Microsystems, which was initiated by James Gosling and

## BUILD AUTOMATION TOOL

released in 1995 as core component of Sun Microsystems's Java platform (Java 1.0 [J2SE]).

- As of December 08 the latest release of the Java Standard Edition is 6 (J2SE). With the advancement of Java and its wide spread popularity, multiple configurations were built to suite various types of platforms. Ex: J2EE for Enterprise Applications, J2ME for Mobile Applications.
- Sun Microsystems has renamed the new J2 versions as Java SE, Java EE and Java ME respectively. Java is guaranteed to be **Write Once, Run Anywhere**

### Java is:

1. **Object Oriented:** In java everything is an Object. Java can be easily extended since it is based on the Object model.
2. **Platform independent:** Unlike many other programming languages including C and C++ when Java is compiled, it is not compiled into platform specific machine, rather into

## BUILD AUTOMATION TOOL

platform independent byte code. This byte code is distributed over the web and interpreted by virtual Machine (JVM) on whichever platform it is being run.

3. **Simple:** Java is designed to be easy to learn. If you understand the basic concept of OOP java would be easy to master.
4. **Secure:** With Java's secure feature it enables to develop virus-free, tamper-free systems. Authentication techniques are based on public-key encryption.
5. **Architectural-neutral:** Java compiler generates an architecture-neutral object file format which makes the compiled code to be executable on many processors, with the presence Java runtime system.

## BUILD AUTOMATION TOOL

6. **Portable:** Being architectural neutral and having no implementation dependent aspects of the specification makes Java portable. Compiler and Java is written in ANSI C with a clean portability boundary which is a POSIX subset.
  
7. **Robust:** Java makes an effort to eliminate error prone situations by emphasizing mainly on compile time error checking and runtime checking.
  
8. **Multi-threaded:** With Java's multi-threaded feature it is possible to write programs that can do many tasks simultaneously. This design feature allows developers to construct smoothly running interactive applications.
  
9. **Interpreted:** Java byte code is translated on the fly to native machine instructions and is not stored anywhere. The



## BUILD AUTOMATION TOOL

development process is more rapid and analytical since the linking is an incremental and light weight process.

10. **High Performance:** With the use of Just-In-Time compilers Java enables high performance.
11. **Distributed:** Java is designed for the distributed environment of the internet.
12. **Dynamic:** Java is considered to be more dynamic than C or C++ since it is designed to adapt to an evolving environment. Java programs can carry extensive amount of run-time information that can be used to verify and resolve accesses to objects on run-time.

**CHAPTER 2**  
**PROPOSED SYSTEM**

## 2.1 Proposed System

- The process of building a computer program is usually managed by a build tool, a program that coordinates and controls other programs. Examples of such a program are make, ant, maven, SCons and Phing.
- The build utility needs to compile and link the various files, in the correct order. If the source code in a particular file has not changed then it may not need to be recompiled (may not rather than need not because it may itself depend on other files that have changed).
- Sophisticated build utilities and linkers attempt to refrain from recompiling code that does not need it, to shorten the time required to complete the build.
- Modern build utilities may be partially integrated into revision control programs like Subversion. A more complex

## BUILD AUTOMATION TOOL

process may involve other programs producing code or data for the build process.

- As time passes situation demands different functionalities with old functionalities to be preserved. Hence upgrading the process is essential. This upgradation is done by installing application on JBoss.
- The advantages of build automation to software development projects includes
  1. Improve product quality.
  2. Accelerate the compile and link processing.
  3. Eliminate redundant tasks.
  4. Minimize "bad builds".
  5. Eliminate dependencies on key personnel.
  6. Have history of builds and releases in order to investigate issues.
  7. Saves time and money.

## 2.2 Objectives of System

- Making a new build of a process in development is a very stressful task. It requires great care and concentration, on work which is essentially tedious. Particularly towards the end of a project, the slightest mistake could be disastrous. Because of the demands of marketing, testing, management and the publisher, it is often carried out under high pressure.
- Computers are particularly good at automating repetitive tasks, and carry them out more quickly and more reliably than humans can.

### 1. Compiling Source Code:

The first thing for a process to be developed is to write any Java source code and then compile which creates as .class file.

### 2. Gathering files :

## BUILD AUTOMATION TOOL

Gather all the files like graphics, sounds, music, scripting code, level designs and video. These can be retrieved from fixed place on the network.

### 3. **Write Build Script:**

At this stage build script is developed using ANT script as XML file which will help to deploy the process.

### 4. **Installing application on JBoss:**

The application is installed on server called JBoss.

### 5. **Upgrading Version:**

At this stage to develop new functionalities along with old functionalities version upgrade is done by performing fresh or upgrade install.

## 2.3 User Requirements

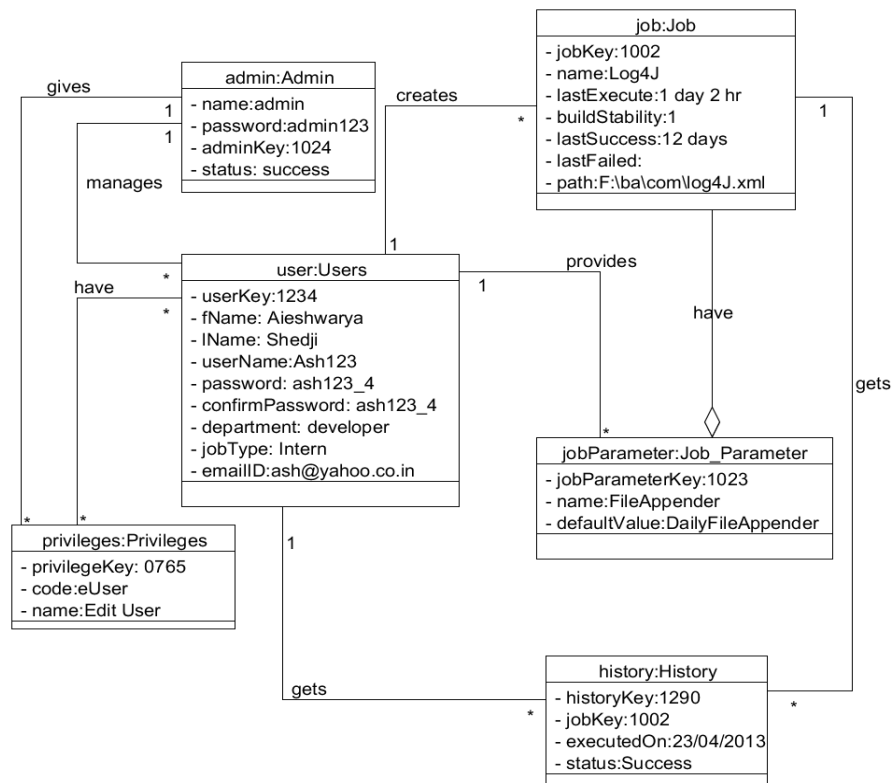
- Following are the task that may be automated with ANT scripts, so they can run without human intervention.
  1. Connect to subversion server.
  2. Download/update with the latest version
  3. Compile the application
  4. Run the test cases
  5. Pack the application ( in jar, war, ear etc)
  6. Commit these build binaries to subversion.
  7. Install the application in a remote server
  8. Restart the server
  9. Send an email with the summary of the job.
- Using build automation tool user can make use of distributed builds by executing only the jobs which is required to build instead of executing whole application.

**CHAPTER 3**  
**ANALYSIS & DESIGN**



### 3.1 Object Diagram

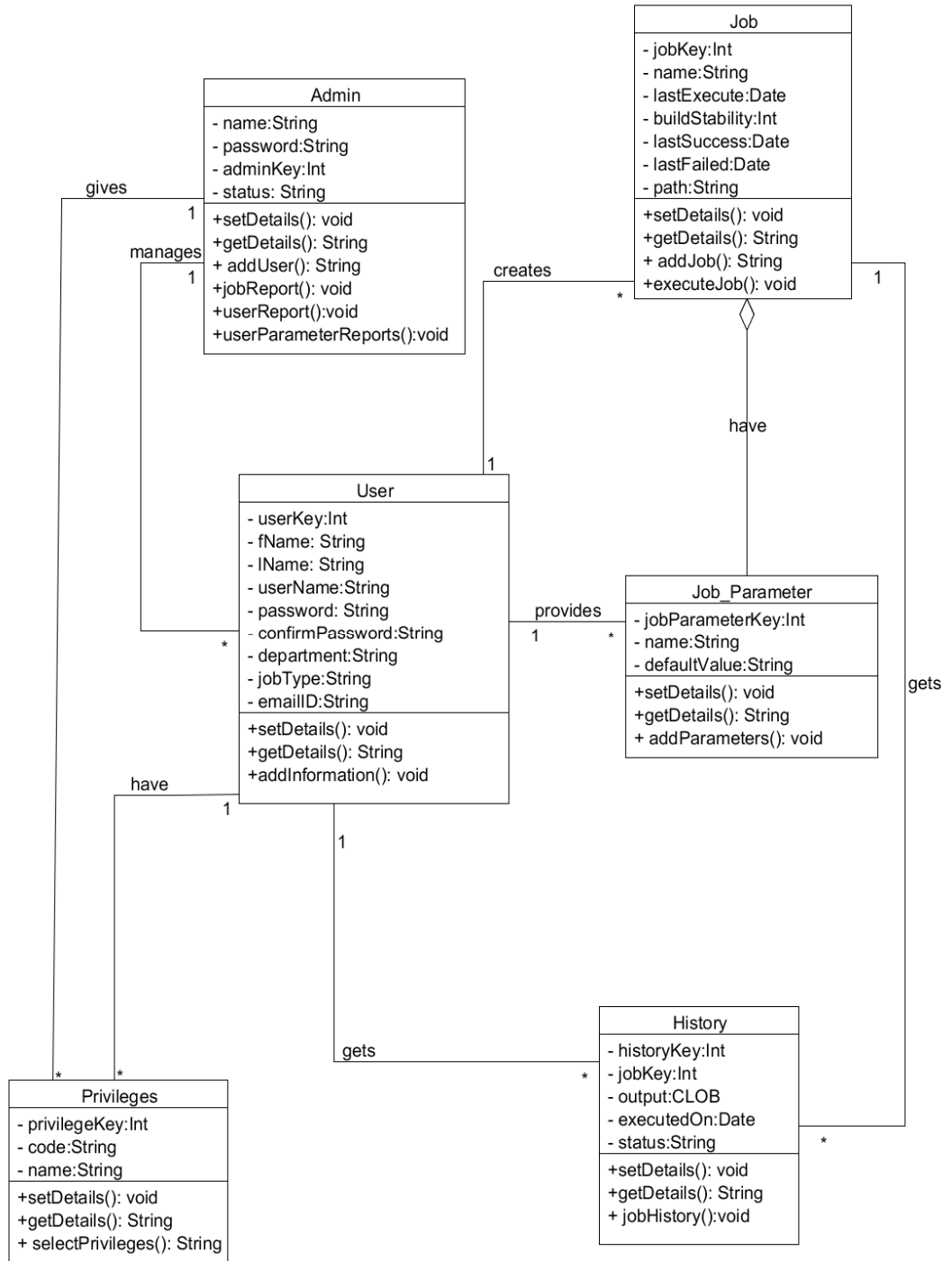
- An object diagram focuses on some particular set of objects and attributes, and the links between these instances. They are often used to provide examples or act as test cases for class diagrams. Only aspects of current interest in a model are typically shown on an object diagram.



## 3.2 Class Diagram

- A class is a system entity that models a real-world object.
- A class is made up of
  1. Attributes:- defines the information that each class knows about itself
  2. Operations: - are the processes that a class can carry out.  
Operations are also known as methods.

# BUILD AUTOMATION TOOL



### 3.3 Use case Diagram

- The use case diagrams describe system functionality as a set of tasks that the system must carry out and actors who interact with the system to complete the tasks. Each use case on the diagram represents a single task that the system needs to carry out. Some use cases may include or extend a task represented by another use case.
- An actor is anything outside the system that interacts with the system to complete a task. It could be a user or another system. The actor "uses" the use case to complete a task.

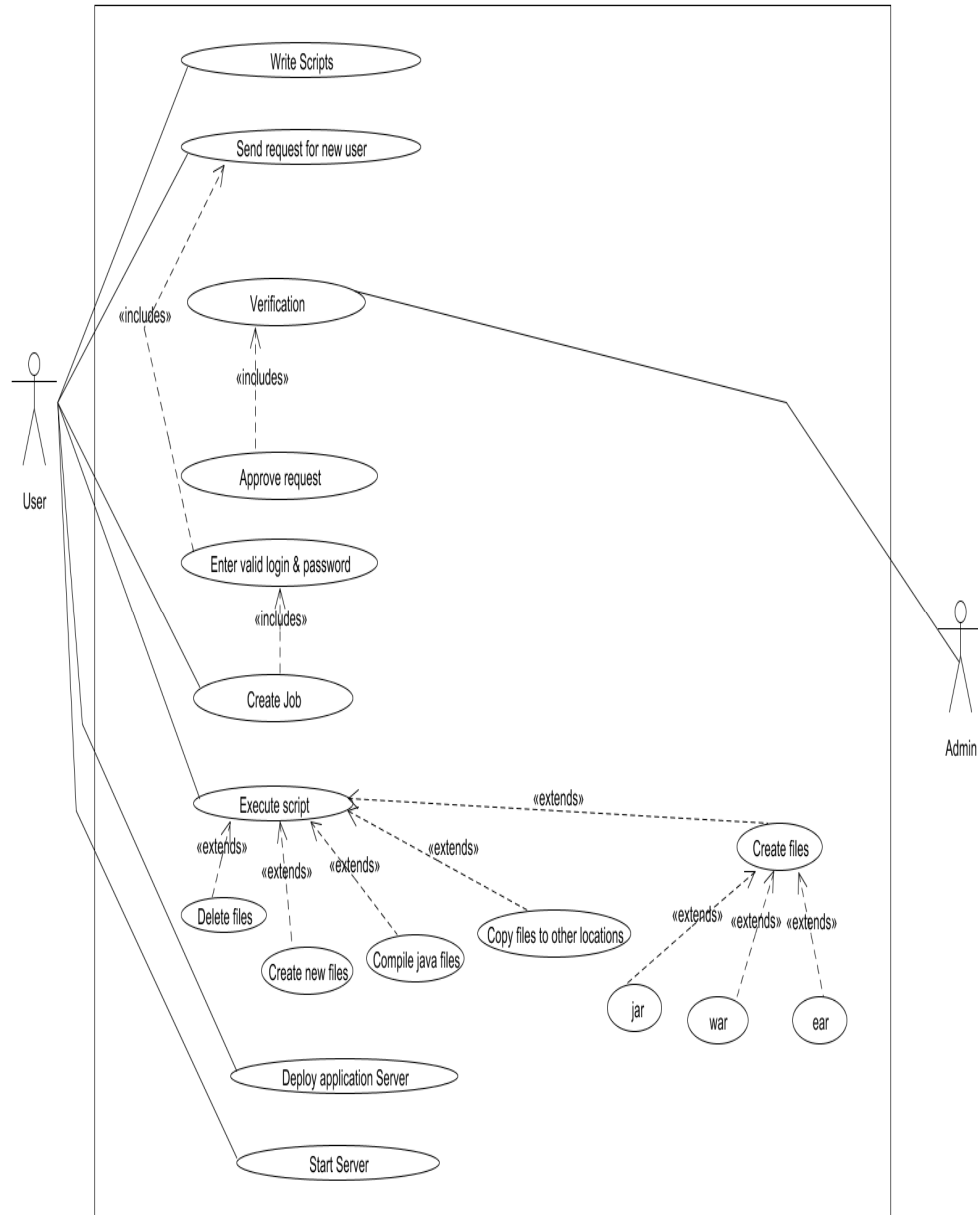
# BUILD AUTOMATION TOOL

## Business Use case:



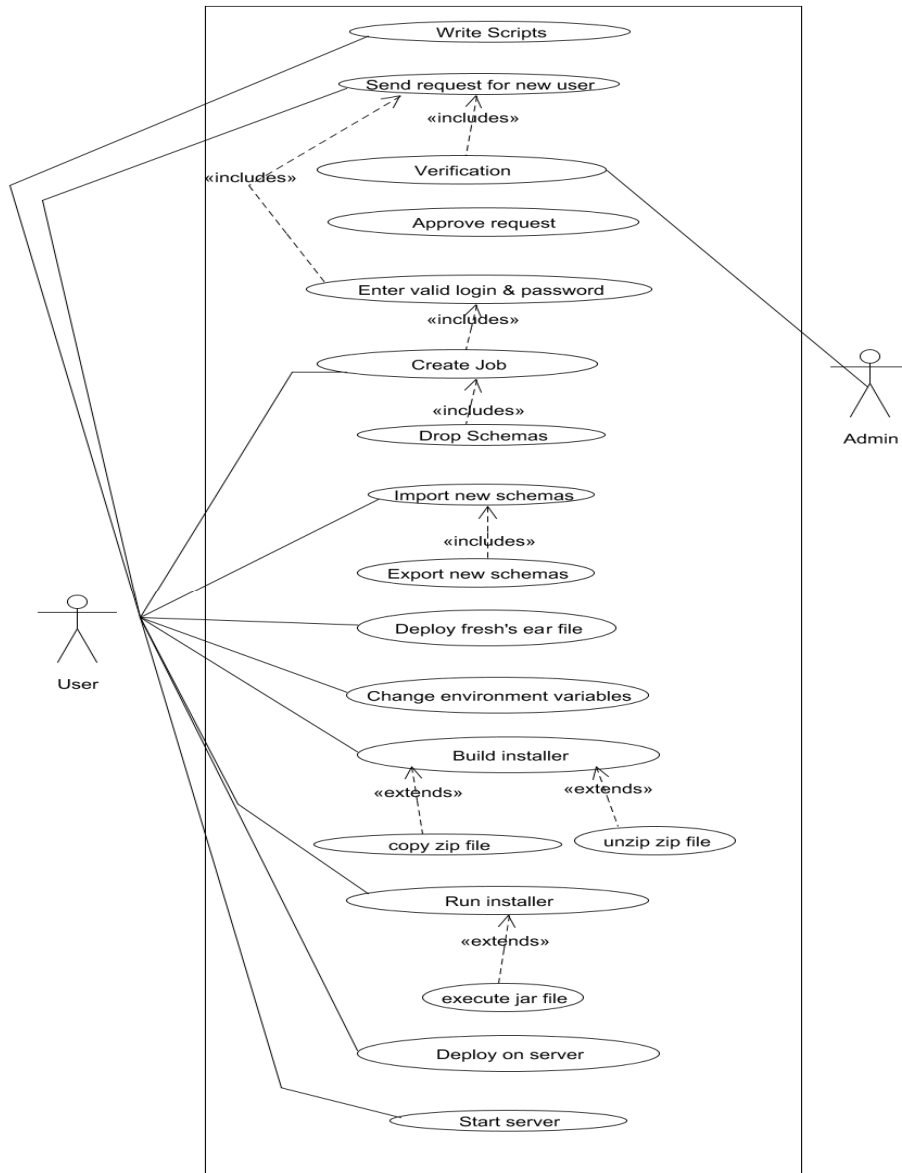
# BUILD AUTOMATION TOOL

## Use case for fresh installation:



# BUILD AUTOMATION TOOL

## Use case for Upgrade installation



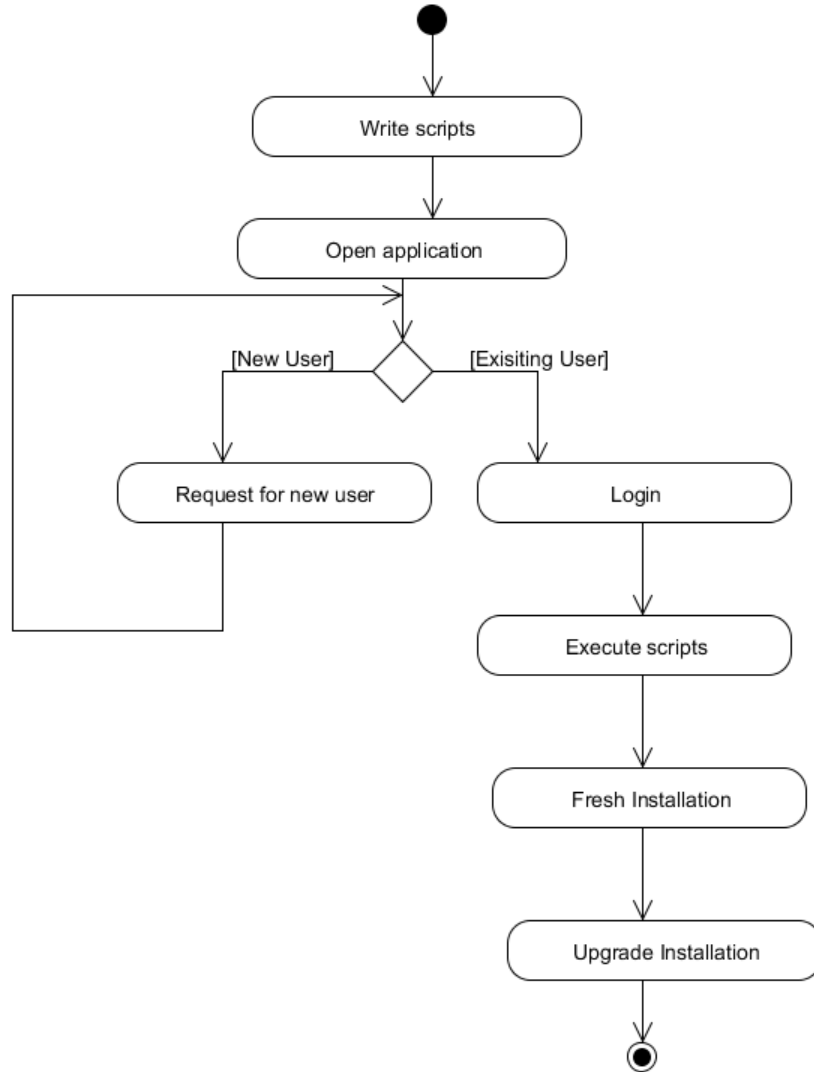
### 3.4 Activity Diagram

- Activity diagram is a flow chart to represent the flow from one activity to another activity.
- The activity can be described as an operation of the system.
- Arrows run from the start towards the end and represent the order in which activities happen.
- Activity diagrams are constructed from a limited number of shapes, connected with arrows. The most important shape types:
  1. Rounded rectangles represent actions;
  2. Diamonds represent decisions;
  3. Bars represent the start (split) or end (join) of concurrent activities;
  4. A black circle represents the start (initial state) of the workflow;
  5. An encircled black circle represents the end (final state).



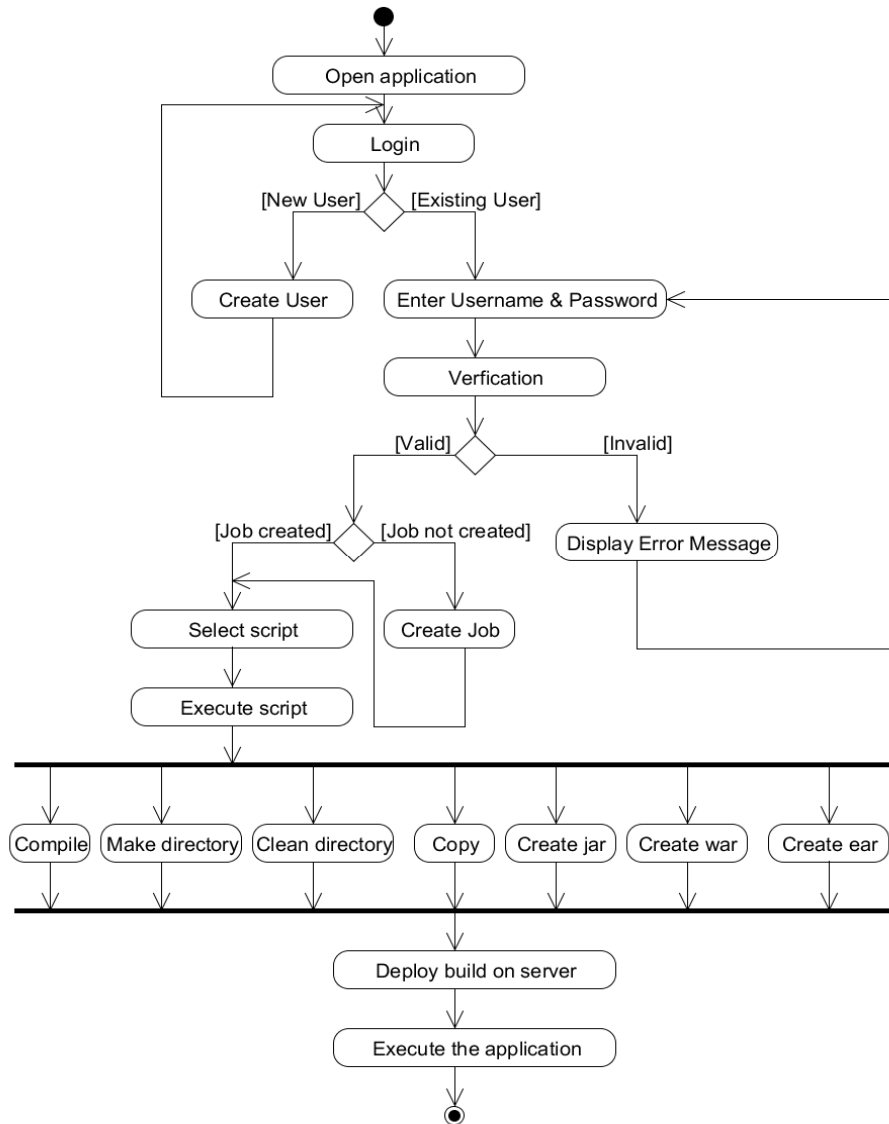
# BUILD AUTOMATION TOOL

## Activity Diagram



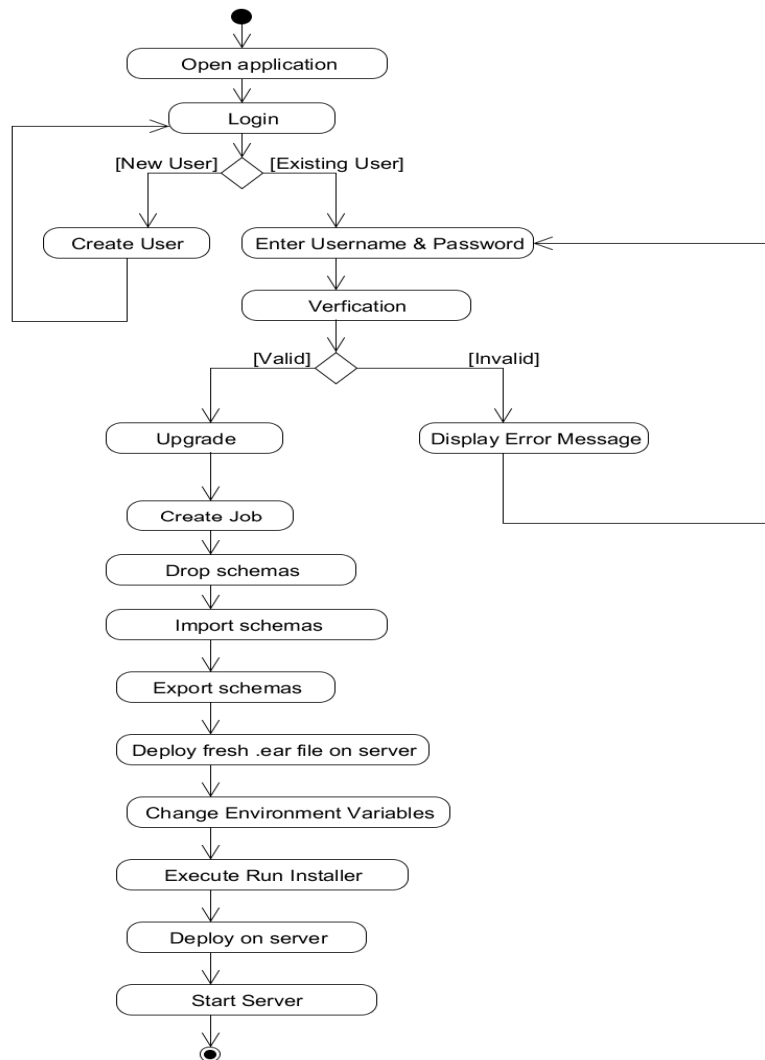
# BUILD AUTOMATION TOOL

## Activity Diagram for fresh installation



# BUILD AUTOMATION TOOL

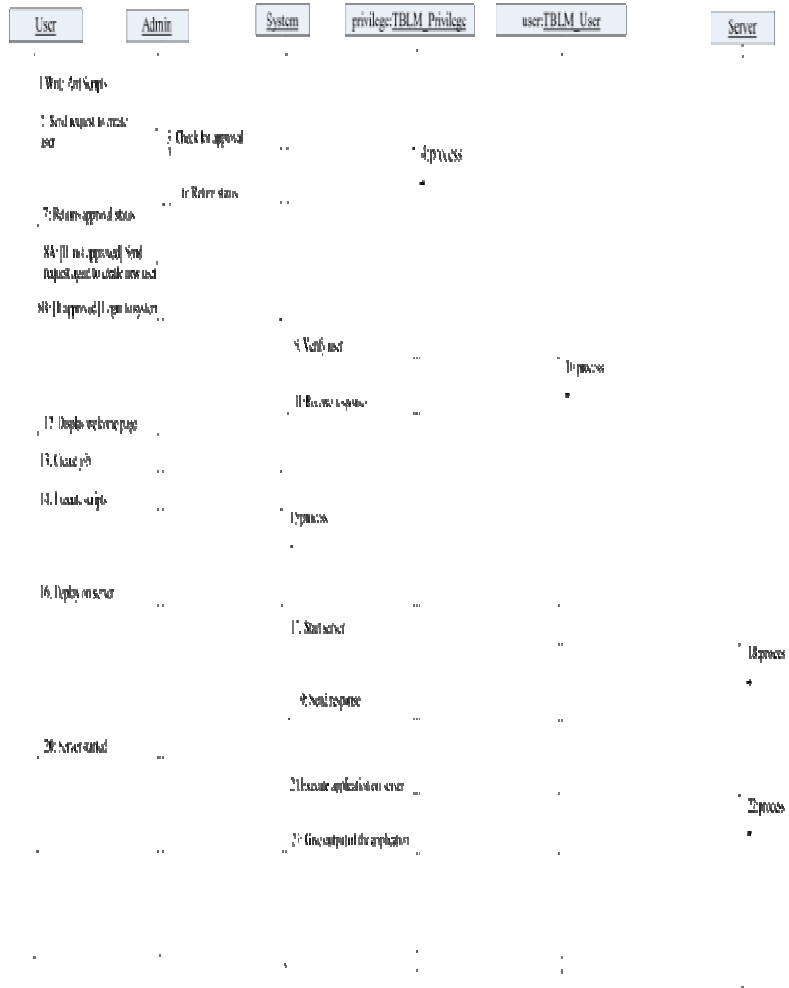
## Activity Diagram for Upgrade Installation



### **3.5 Sequence Diagram**

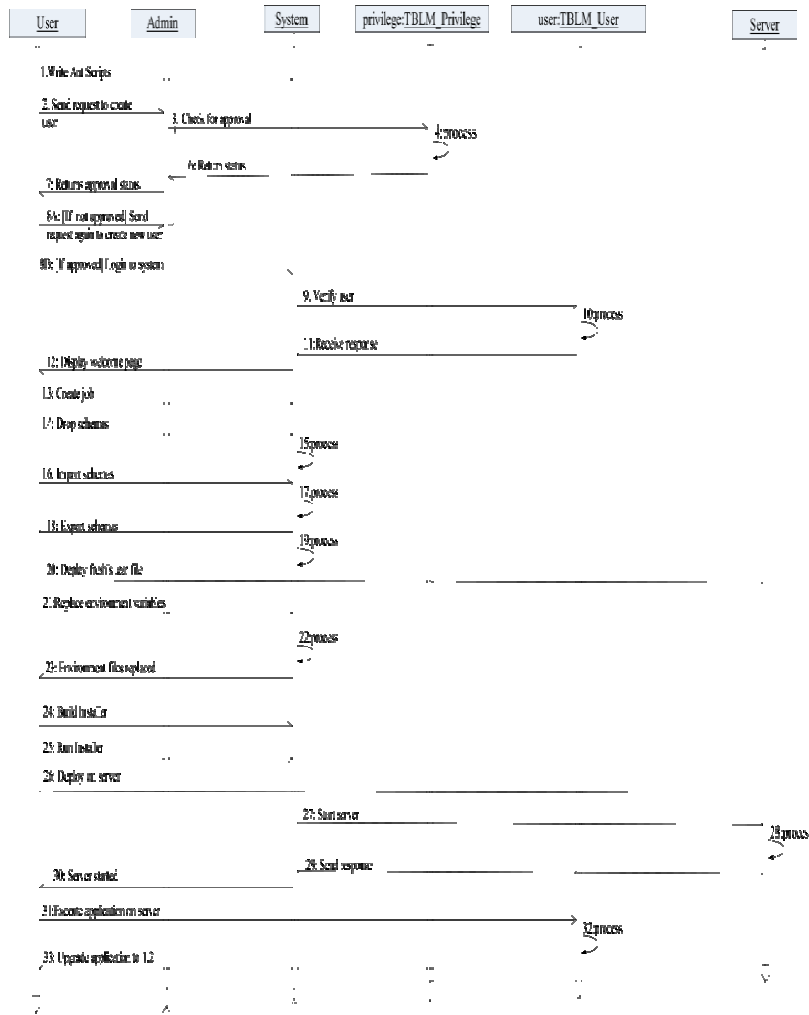
- Sequence diagrams are used to represent or model the flow of messages, events, & actions between the objects or components of a system.
- Time is represented in the vertical direction showing the sequence of interactions of the header elements, which are displayed horizontally at the top of diagram.

## Sequence Diagram for Fresh Installation



# BUILD AUTOMATION TOOL

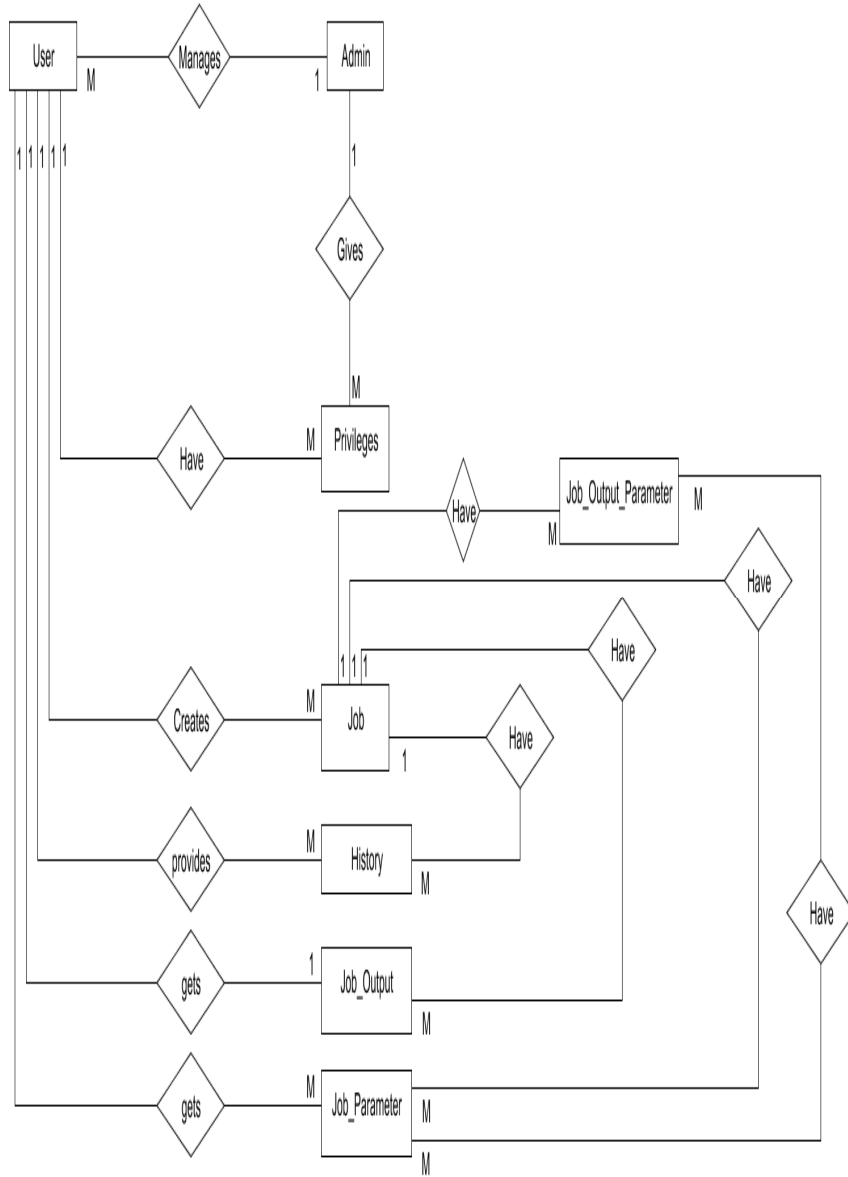
## Sequence Diagram for Upgrade Installation



### 3.6 Entity Relationship Diagram

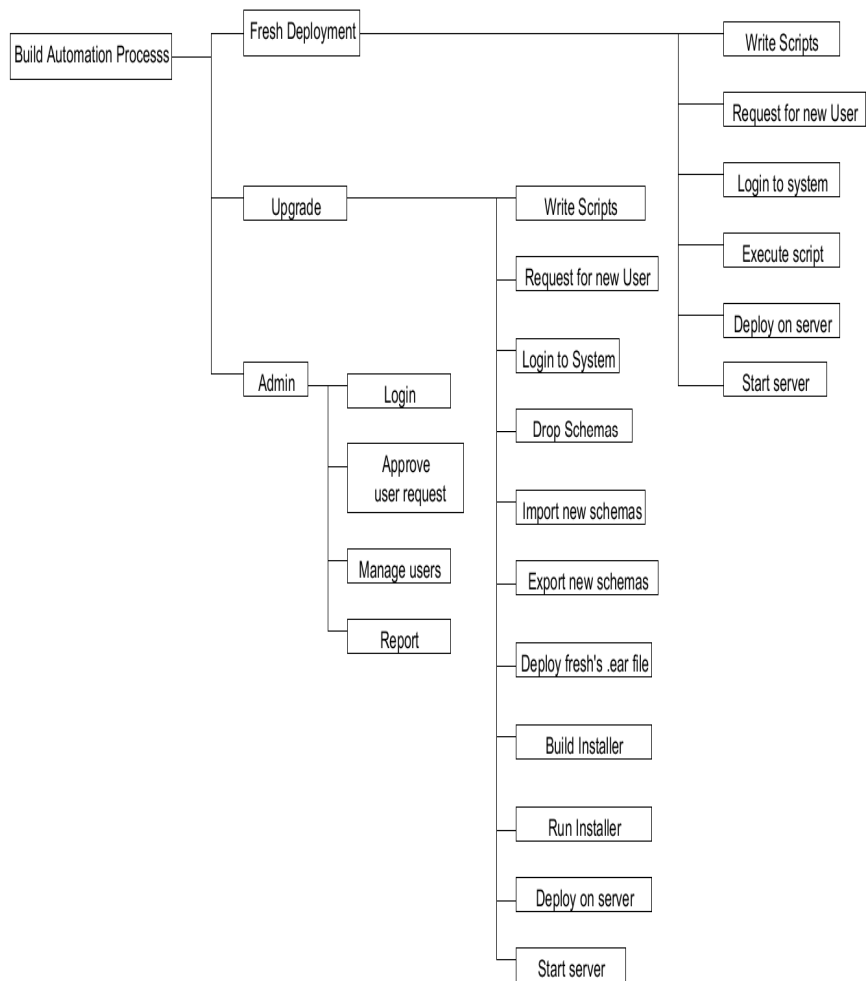
- An Entity Relationship Diagram (ERD) is a visual representation of different data using conventions that describes how these data are related to each other.
- The elements inside the rectangle are called entities while the items inside the diamonds denote relationship between entities.
- The three main components of an ERD are:
  1. The **entity** is a person, object, place or event for which data is collected.
  2. The **relationship** is the interaction between the entities.
  3. The **cardinality** defines the relationship between the entities in terms of numbers.

# BUILD AUTOMATION TOOL



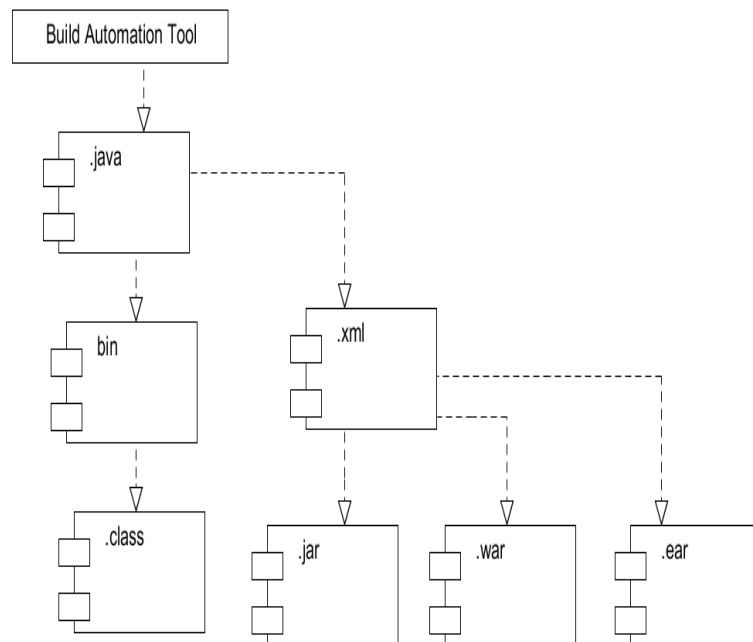


### 3.7 Module Hierarchy Diagram



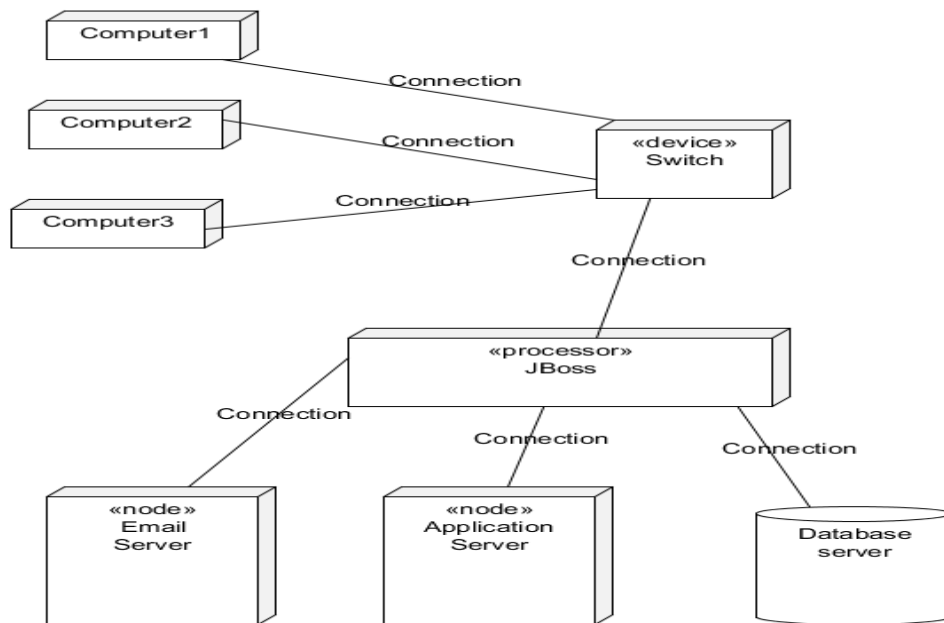
### 3.8 Component Diagram

- Component diagrams show how the physical components of a system are organized. A component is a distributable unit of software. The component diagram allows you to combine deployment nodes with components to show which components run on each node (i.e. hardware).



### 3.9 Deployment Diagram

- The deployment diagram shows how the components of the completed system are physically related.
- Each node represents a computational unit such as a piece of hardware. The connections between the nodes show the system interaction paths.



### **3.10 Module Specification**

#### **I. Admin Module**

➤ This module is developed by considering “administrator ” as user and its functionality in Build Automation Tool. This module contains following functionality.

1. Administrator validates other user registration.
2. Administrator provides privileges to other users.
3. Administrator creates and executes jobs.
4. Administrator can view and maintains job history.
5. Administrator can view user and edit the privileges given to them.
6. Administrator can provide run time parameter values for repective parameters for job, if any.

### **II. Fresh Installation**

- This module is developed by considering various user who develops java application and its functionality in Build Automation Tool. This module contains following functionality.
  1. User can automatically compile java files and stores .class file in respective folder like bin.
  2. User can automatically create jar, war and ear files and stores file in respective folder, so that user just needs to extract jar or war or ear files.
  3. User can move files from any directory to any destination directory by giving correct path.

## BUILD AUTOMATION TOOL

4. User can move limited files instead of folder itself to any destination directory provided that path is correct.
5. User can delete complete folder or any particular file in that folder.
6. User can execute and view job provided he/she has privilege to do so.

### **III. Upgrade Installation**

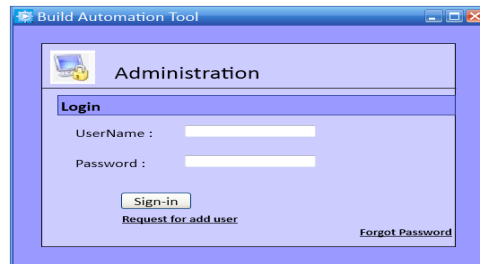
- This module is developed by considering various user who develops java application and its functionality in Build Automation Tool. This module contains following functionality.

## BUILD AUTOMATION TOOL

1. User can upgrade functionality of an application so user can import or export data required for application to other user.
2. User needs to execute batch files in which various sql queries needs to be written to get correct output of import and export of data.
3. User needs to copy.ear file of previous version on server.
4. User needs to execute the job which will prepare or build the installer.
5. User needs to execute the jar file by giving appropriate installation type and required parameter's value.
6. Execute .exe file to get installed upgraded version of an application.

### 3.11 User Interface Design

#### Login Screen



#### Welcome Page





# BUILD AUTOMATION TOOL

## Add User

The screenshot shows a web application window titled "Build Automation Tool" with a sub-header "Add User". The interface is divided into two main sections: "User Details" and "Privilege Groups".

**User Details**

User Name :	<input type="text"/>	New Password :	<input type="text"/>
First Name :	<input type="text"/>	Confirm Password :	<input type="text"/>
Last Name :	<input type="text"/>	Email Address :	<input type="text"/>
Designation :	<input type="text"/>	Department :	<input type="text"/>

**Privilege Groups**

Privileges


- Create User
- Edit User
- List User
- Create Job
- View Job Output
- Edit Job
- List Job
- Execute Job
- List Job Output

At the bottom right, there are two buttons: "Submit" and "Cancel". A "Logout" link is visible in the top right corner of the form area.

# BUILD AUTOMATION TOOL

## Add Job

Build Automation Tool

 **Add Job** Logout

---

**Job Details**

Job Name :

---

**Parameter Details**

Parameter Name :

Default Value :

Description :  ADD

---

**Job Parameters**

Name	Default Value	Description

---

**Script**

Script Path :  Browse

User Notification :  Add    Cancel

# BUILD AUTOMATION TOOL

## Execute Job



## History



# BUILD AUTOMATION TOOL

## Admin grants privileges

**Build Automation Tool** [Logout]

**Add User**

**User Details**

User Name :       New Password :

First Name :       Confirm Password :

Last Name :       Email Address :

Designation :       Department :

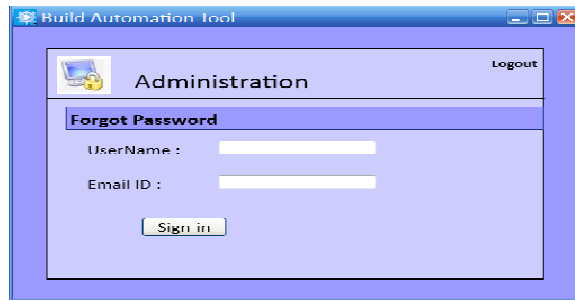
**Privilege Groups**

Privileges

- Create User
- Edit Job
- Edit User
- List Job
- List User
- Execute Job
- Create Job
- List Job Output
- View Job Output

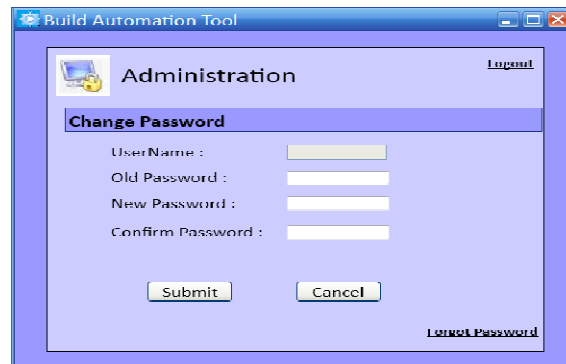
# BUILD AUTOMATION TOOL

## Forgot Password



The screenshot shows a web application window titled "Build Automation Tool". Inside, there is a header area with a logo, the word "Administration", and a "Logout" link. Below this is a section titled "Forgot Password". This section contains two input fields: "UserName :" and "Email ID :". Below the input fields is a "Sign in" button.

## Change Password



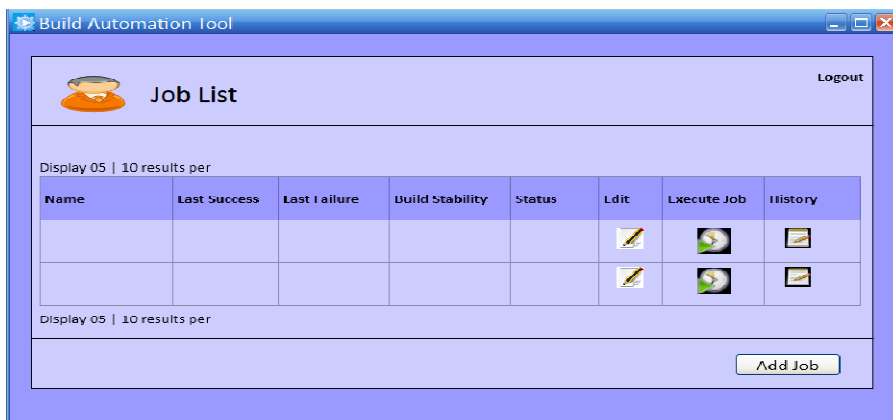
The screenshot shows a web application window titled "Build Automation Tool". Inside, there is a header area with a logo, the word "Administration", and a "Logout" link. Below this is a section titled "Change Password". This section contains four input fields: "UserName :", "Old Password :", "New Password :", and "Confirm Password :". Below the input fields are two buttons: "Submit" and "Cancel". At the bottom right of the form area, there is a "Reset Password" link.

# BUILD AUTOMATION TOOL

## User List



## Job List



### 3.12 Table Specification

**Table Name: TBLM\_Admin**

Sr. No	Field Name	Data Type	Size	Constraints
1	adminKey	Number	4	Primary Key
2	name	Varchar2	10	Not Null
3	Password	Varchar2	10	Not Null

**Table Name: TBLM\_User**

Sr. No	Field Name	Data Type	Size	Constraints
1	userKey	Number	4	Primary Key
2	fName	Varchar2	10	Not Null
3	lName	Varchar2	10	Not Null
4	Username	Varchar2	8	Not Null
5	Password	Varchar2	8	Not Null
6	confirmPassword	Varchar2	8	Not Null
7	emailId	Varchar2	20	Not Null
8	jobtype	Varchar2	10	Not Null
9	department	Varchar2	10	Not Null
10	adminKey	Number	4	Foreign Key

**Table Name: TBLM\_Privilege**

Sr. No	Field Name	Data Type	Size	Constraints
1	privilegeKey	Number	4	Primary Key
2	code	Varchar2	4	Not Null
3	name	Varchar2	10	Not Null
4	adminKey	Number	4	Foreign Key
5	userKey	Number	4	Foreign Key

## BUILD AUTOMATION TOOL

**Table Name: TBLM\_Job**

Sr. No	Field Name	Data Type	Size	Constraints
1	jobKey	Number	4	Primary Key
2	Name	Varchar2	10	Not Null
3	lastExecuted	Date		Not Null
4	buildStability	Number	1	Not Null
5	lastSuccess	Date		Not Null
6	lastFailed	Date		Not Null
7	Path	Varchar2	10	Not Null
8	jobOutputKey	Number	4	Foreign Key
9	jobParameterKey	Number	4	Foreign Key
10	userKey	Number	4	Foreign Key

**Table Name: TBLT\_JobOutput**

Sr. No	Field Name	Data Type	Size	Constraints
1	jobOutputKey	Number	4	Primary Key
2	Output	CLOB		Not Null
3	executedOn	Date		Not Null
4	Status	Varchar2	7	Not Null
5	jobKey	Number	4	Foreign Key

**Table Name: TBLM\_History**

Sr. No	Field Name	Data Type	Size	Constraints
1	historyKey	Number	4	Primary Key
2	jobKey	Number	4	Foreign Key
3	userKey	Number	4	Foreign Key



## BUILD AUTOMATION TOOL

**Table Name: TBLT\_Status**

Sr. No	Field Name	Data Type	Size	Constraints
1	statusKey	Number	4	Primary Key
2	statusName	Varchar2	15	Not Null
3	adminKey	Number	4	Foreign Key
4	userKey	Number	4	Foreign Key

**Table Name: TBLT\_JobParameter**

Sr. No	Field Name	Data Type	Size	Constraints
1	jobParameterKey	Number	4	Primary Key
2	Name	Varchar2	10	Not Null
3	defaultValue	Varchar2	10	Not Null
4	description	Varchar2	20	Not Null
5	jobKey	Number	4	Foreign Key

**Table Name: TBLR\_JobOutput\_Parameter**

Sr. No	Field Name	Data Type	Size	Constraints
1	jobOutputKey	Number	4	Foreign Key
2	jobParameterKey	Number	4	Foreign Key
3	Value	Varchar2	10	Not Null
4	jobKey	Number	4	Foreign Key

### **3.13 Test Procedures & Implementation**

#### **TEST PROCEDURES**

- Software testing is the activity that evaluates the capability of software, along with ascertaining if the software meets the requirements laid out.
- The aim of the entire software testing activity is to verify and validate the software, along with quality assurance and reliability estimation.

#### **Objectives or Goals of Testing**

1. To demonstrate that errors are not present.
2. To show that intended functions are present.
3. To gain confidence in the software ability to do what it is required to do.
4. Discover the unnoticed errors
5. Increase the probability of discovering errors.

### **Testing Principles**

- While doing this we stick to testing principles. Some of them are as follows
1. Testing Plan and Schedules were prepared and incorporated in overall project development.
  2. Test results and reports were prepared and rectification of bugs was also worked out as per test plan.

### **Testing Procedure**

1. Prepare a test plan.
2. Review the test plan.
3. Evolve the test plan.
4. Generate test cases.
5. Prepare test scripts.
6. Create required test environment.
7. Executes test by initiating application under test. Applying as input already generated test cases.

8. Compare actual results with expected results.

1) See the outcome: Test failed or Test pass.

### **Security Testing**

- System is in threat in various ways. Security testing attempts to verify that protection mechanism built into a system will in fact protect it from improper penetration. During security testing the developer itself test various type of illegal access to the system and confirm it.
- This includes session handling-destroying session, setting session timeout, validating username and password.

### **User Interface Testing**

- The Graphical User Interface (GUI) plays a key role in the success of any system. Providing a better GUI is now become major trend today and hence it's testing is very important.
- This includes validating different windows operation, pull-down menus and mouse operations. Data entry – front end

## BUILD AUTOMATION TOOL

validations, preventing all invalid data entry. Checking performance and speed of search.

### **Business Logic/DB Testing**

- This includes checking records updated, deleted, inserted properly, hidden variables carry proper information, no internal error, no 500 error, no DB error, different constraint are meet.

### **Black Box Testing**

- In this structure of the program is not considered. The software is considered as Black box to which defined inputs are given. From this defined outputs are obtain. In this tester only knows what software is supposed to do but tester cannot look in the box to see how it operates. We do not have to worry about internal structure of program as we are carrying the functional testing.

## BUILD AUTOMATION TOOL

- The Black Box testing can find errors such as interface errors, incorrect or missing functions, initialization and termination errors, performance errors, error in data structure and external DB access etc. it is also called as Data Driven testing.
- Doing this all parameters are meet, no internal errors, no 500 errors, no page not found errors.

### Test Case

- A test case is a set of conditions or variables and inputs that are developed for a particular goal or objective to be achieved on a certain application to judge its capabilities or features.

### Test Case for Login

Test Case id	Test name	Test description	User	Test desired result	Actual result	Pass /Fail	Remark
TC-1	User Authentication	Enter the URL & click enter	User_id	Display Login page	Login page is displayed	Pass	Page display with required field
TC-2	User Authent	Keep usernam	User_id	Displ ay	Error messa	Pass	Errors for both

## BUILD AUTOMATION TOOL

	ication	e , passwor d field empty & press Login		error messa ge	ge displa yed		field is display
TC-3	User Authent ication	Keep usernam e field empty & fill passwor d field & press Login	Useri d	Displ ay error messa ge	Error messa ge displa yed	Pass	Error for usernam e field is display
TC-4	User Authent ication	Keep passwor d field empty & fill usernam e field & press Login	User_ id	Displ ay error messa ge	Error messa ge displa yed	Pass	Error for passwor d field is display
TC-5	User Authent ication	Insert usernam e & passwor d field invalid & press Login	User_ id	Displ ay error messa ge	Error messa ge displa yed	Pass	Error is display
TC-6	User Authent	Insert usernam	User_ id	Login	Login	Pass	User directed

## BUILD AUTOMATION TOOL

	ication	e & password field valid & press Login					to home page
--	---------	--	--	--	--	--	--------------

### Test Case for Privileges

Test Case id	Test name	Test description	User	Test desired result	Actual result	Pass /Fail	Remark
TC-1	User Authentication	User Enters his login credentials	User id	Display Appropriate Screens	Only permissible menu options shown	Pass	Page display with required field
TC-2	User Authentication	User Tries to access other's Credentials	User id	Display error message	Error message displayed	Pass	Errors for both field is display
TC-3	User Authentication	Clicks on Any Permissible Option	User id	Display The Appropriate Page	Appropriate Page is shown	Pass	Access the page



## BUILD AUTOMATION TOOL

### Test Case for Master Skills

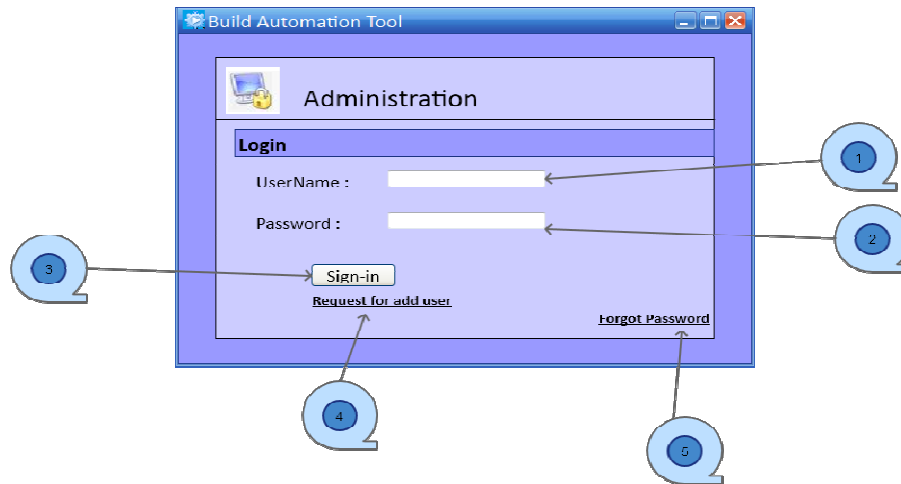
<b>Test Case id</b>	<b>Test name</b>	<b>Test description</b>	<b>User</b>	<b>Test desired result</b>	<b>Actual result</b>	<b>Pass /Fail</b>	<b>Remark</b>
TC-1	User Authentication	User Enters his login credentials	User_id	Display Appropriate Screens	Only permissible privileges are shown	Pass	Page display with required field
TC-2	Check Box list	Select the check box	User_id	Only privileges which are checked should be enabled	Only privileges which are checked are enabled properly	Pass	Access the Page to perform desired operations
TC-3	Empty Field test	Keeps any field blank	User_id	Display Error Message	Error Message Displayed	Pass	Error Field Displayed
TC-4	Submit Button Clicked	Every Field is Entered Properly	User_id	Data is stored into the Database	Data Stored Successfully	Pass	Success message Displayed

**CHAPTER 4**  
**USER MANUAL**

## 4.1 User Manual

This manual gives briefing to the user as how to use Build Automation Tool. When the user double clicks on the application, user gets first page of application.

This looks like this:



### **Login Authentication**

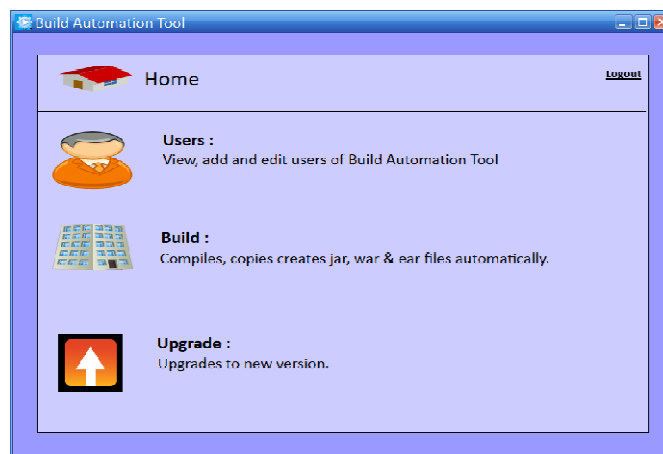
1. Enter valid user name.
2. Enter valid password.
3. Press Sign In button to get logged in.
4. If user is not registered to the tool, then he/she may go to add user button.
5. If user has forgotten the password, then he/she may click on Forgot Password link and get new password.

# BUILD AUTOMATION TOOL

## Home Page

After valid username and password user is moved to next page as Home page.

1. User can view users list by clicking on Users link,
2. User can create or view job and can create files like jar,war,ear.
3. Users can upgrade the applicaion by clicking on upgrade.



## BUILD AUTOMATION TOOL

### Add User

1. Enter valid values for all fields shown in below screen.
2. Send it to admin to get privileges from admin.
  - If you get Edit Job as a privilege then List job will be automatically gets selected but it's not true for vice versa.
  - If you get Edit User as a privilege then List User will be automatically gets selected but it's not true for vice versa.

Build Automation Tool

**Add User** [Logout](#)

**User Details**

User Name :  New Password :   
First Name :  Confirm Password :   
Last Name :  Email Address :   
Designation :  Department :

**Privilege Groups**

Privileges

<input type="checkbox"/> Create User	<input type="checkbox"/> Edit Job
<input type="checkbox"/> Edit User	<input type="checkbox"/> List Job
<input type="checkbox"/> List User	<input type="checkbox"/> Execute Job
<input type="checkbox"/> Create Job	<input type="checkbox"/> List Job Output
<input type="checkbox"/> View Job Output	

## BUILD AUTOMATION TOOL

### **Add Job**

1. Enter valid values for Job in below screen.
2. If user wants to add parameter for job, he/she should provide its name, default value and description.
3. Click add button to add parameters.
4. Users will see list of parameters added if any.
5. Provide path of ANT or batch script which users want to execute.
6. Select name of the user to whom you want to send mail related to job.
7. Click add button to add the job.

# BUILD AUTOMATION TOOL

**Build Automation Tool** [Logout]

**Add Job**

**Job Details**

Job Name :  (1)

**Parameter Details**

Parameter Name :  (2)

Default Value :  (3)

Description :

**Job Parameters**

Name	Default Value	Description

(4)

**Script**

Script Path :   (5)

User Notification :  (6)

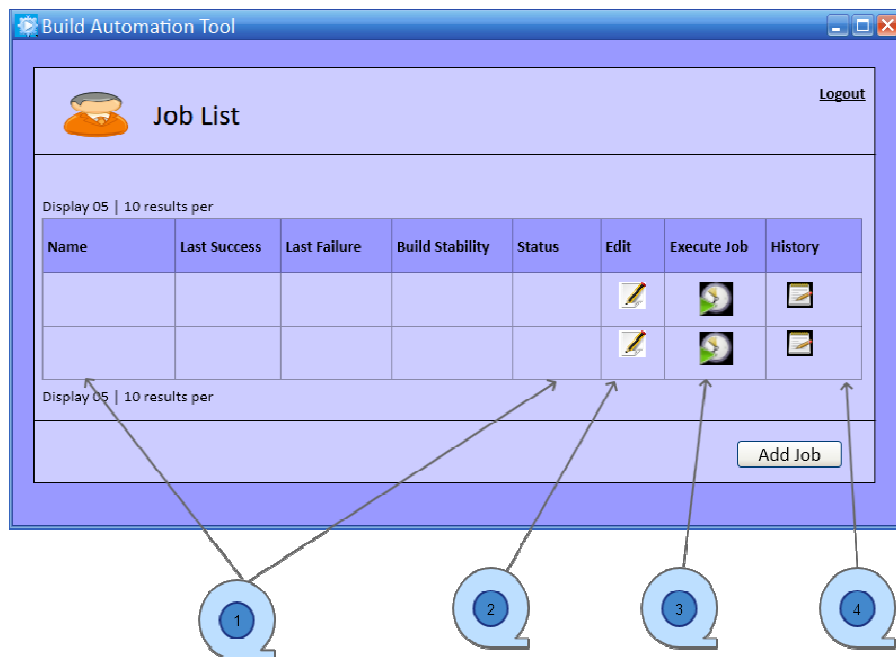
(7)



# BUILD AUTOMATION TOOL

## Job List

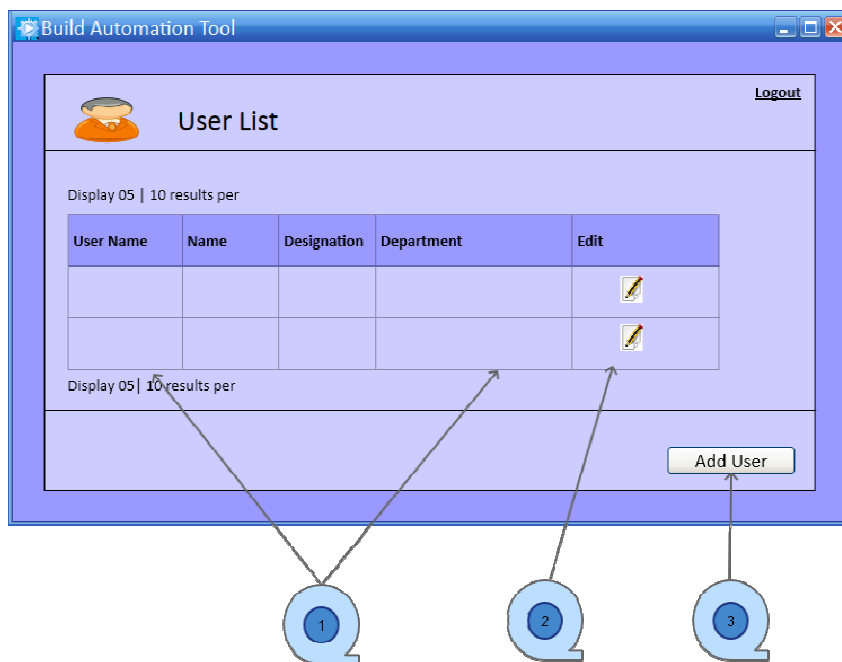
1. User can view the list of jobs.
2. If user wants to edit the job, user needs to click on edit icon.
3. If user wants to execute the job, user needs to click on execute icon.
4. If user wants to view the history of the job, user needs to click on history icon.



# BUILD AUTOMATION TOOL

## User List

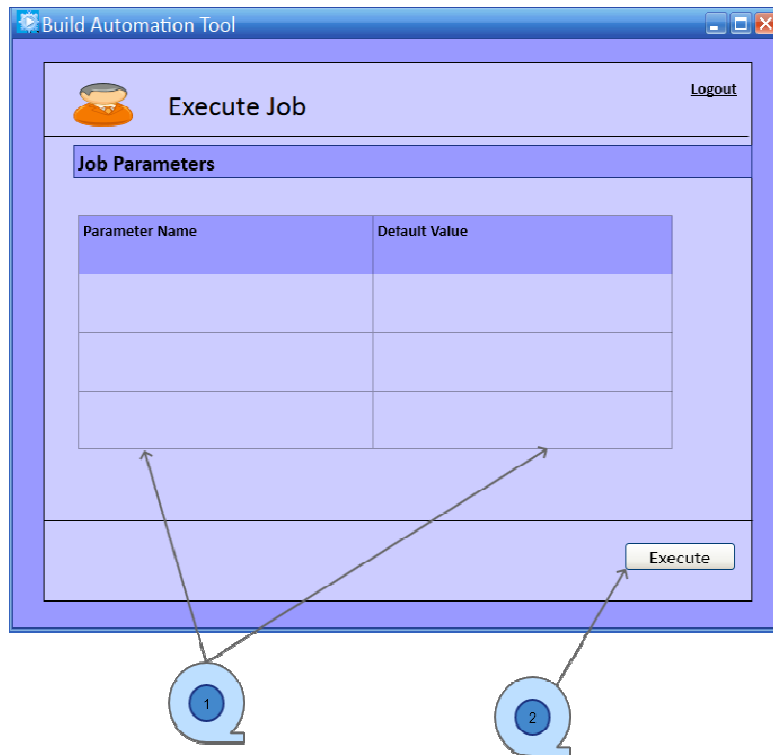
1. User can view the list of various users.
2. If user wants to edit the user information, user needs to click on edit icon.
3. If user has privilege to add another user then user may add another user, if not then add user button will get disable as you do not have privilege from admin.



# BUILD AUTOMATION TOOL

## Execute job

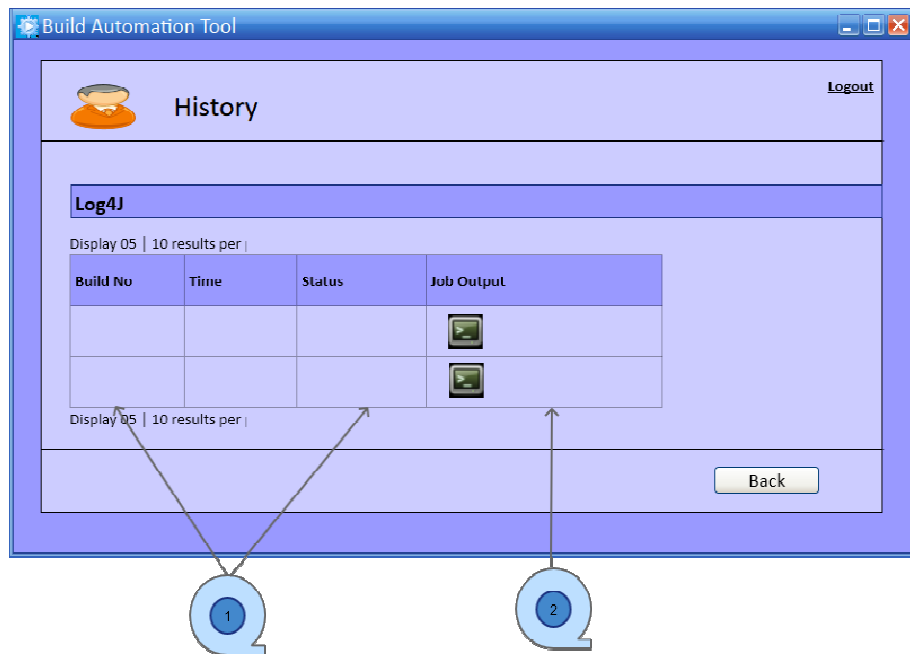
1. Here user can view parameter name and its value and then execute.



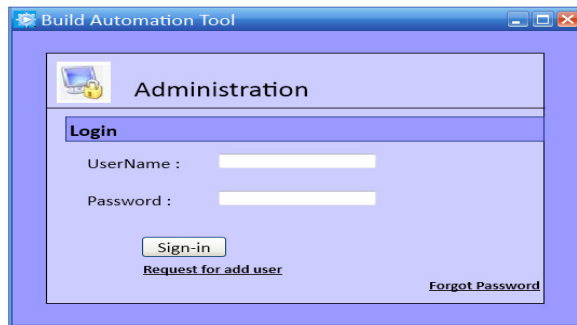
# BUILD AUTOMATION TOOL

## History

1. User can view history of all jobs that are executed with name, time, status and job output.
2. User can view the job output as well.



## 4.2 Operation Manual



Sr. No	Field Name	Description	Constraint	Default Value
1	UserName	Enter username	This is text field and the length of username should be between 5-10 characters	Blank
2	Password	Enter valid password. It is case sensitive. String is displayed in encrypted format	This is text field and the length of password should be between 8-10 characters	Blank

## BUILD AUTOMATION TOOL

**Add User** [Logout](#)

**User Details**

User Name :       New Password :   
 First Name :       Confirm Password :   
 Last Name :       Email Address :   
 Designation :       Department :

**Privilege Groups**

Privileges

- Create User
- Edit User
- List User
- Create Job
- View Job Output
- Edit Job
- List Job
- Execute Job
- List Job Output

Sr. No	Field Name	Description	Constraint	Default Value
1	User Name	Enter username	This is text field and the length of username should be between 5-10 characters	Blank
2	First Name	Enter name	This is text field, only characters	Blank

## BUILD AUTOMATION TOOL

			are accepted	
3	Last Name	Enter last name	This is text field, only characters are accepted	Blank
4	New Password	Enter valid password. It is case sensitive. String is displayed in encrypted format	This is text field and the length of password should be between 8-10 characters. Alphabets and one numeric value as well as special characters such as ‘_’,and ‘/’ must be there	Blank
5	Confirm Password	Enter same password as new password. It is case sensitive. String is displayed in encrypted format	This is text field and the length of password should be between 8-10 characters. Alphabets and one numeric value as well as special characters such as	Blank


## BUILD AUTOMATION TOOL

			'_',and '/' must be there	
6	Email Address	Enter valid email address	This is text field.	Blank
7	Privileges	Choose the privileges	This are check boxes will be filled by admin after he gets requests sent by you.	Blank



# BUILD AUTOMATION TOOL

Build Automation Tool Logout

 **Add Job**

---

**Job Details**

Job Name :

---

**Parameter Details**

Parameter Name :

Default Value :

Description :  ADD

---

**Job Parameters**

Name	Default Value	Description

---

**Script**

Script Path :  Browse

User Notification :

Add Cancel

## BUILD AUTOMATION TOOL

Sr. No	Field Name	Description	Constraint	Default Value
1	Job Name	Enter job name	This is text field only character are accepted	Blank
2	Parameter Name	Enter parameter name	This is text field, only characters are accepted	Blank
3	Default Value	Enter default value	This is text field, the length of password should be 30 characters.	Blank
4	Script Path	Select path by clicking on browse button	This is text field valid xml or batch file path will be accepted	Blank
5	User Notification	Select name of user whom you want to send Email	This is list box, and value needs to be selected from the list	Name

### 4.3 Program Specification

#### 1. User Registration

<b>Module</b>	User Registration	
<b>Program Name</b>	Add User	
<b>Purpose</b>	To add user details	
<b>Input</b>	<b>Constraints</b>	<b>Description</b>
User Details	All the text fields must not be null, but user may select any number of check boxes as per user . To choose privileges is an optional.	User Details get stored in table TBLM_User. Privilege Details gets stored in table TBLM_Privilege.
<b>Output</b>	User gets the acknowledgment of the record added.	

# BUILD AUTOMATION TOOL

## 2. Job Creation

<b>Module</b>	Job Creation	
<b>Program Name</b>	Add Job	
<b>Purpose</b>	To add Job	
<b>Input</b>	<b>Constraints</b>	<b>Description</b>
Job Details	All the text fields must not be null, but user may enter any number of parameter name and their respective default value and description To enter parameter details is an optional.	Job Details get stored in table TBLR_Job. Job Parameter details get stored in TBLT_JobParameter
<b>Output</b>	The user gets the acknowledgment of the job added.	

## BUILD AUTOMATION TOOL

### 3. Execute Job

<b>Module</b>	Execute Job	
<b>Program Name</b>	Execute Job	
<b>Purpose</b>	To execute Job	
<b>Input</b>	<b>Constraints</b>	<b>Description</b>
Parameter Details	Parameters value needs to be given for respective parameter names.	Job Parameter details get stored in TBLT_JobParameter
<b>Output</b>	The user gets the output as per the execution of job result on console screen.	

## **Drawbacks & Limitations**

- The system requires prerequisites such as
  1. ANT
  2. Jdk
  3. JBOSS server
- The system cannot generate .exe file of an application.
- The Email module does not provide the provision to send the attachment.

### **Proposed Enhancement**

- Provision to send builds information as attachment to another user.
- Provision to generate .exe file for upgrade application to generate next level version.

### **Conclusions**

- The Build Automation Tool helps to automate the integration of distributed builds among different nodes. It also provides the reporting capability for generating the report based on this information.
- While installing application user needs to do following tasks manually which is time consuming:
  1. Compiling computer source code into binary code.
  2. Packaging binary code.
  3. Running tests.
  4. Deployment to production systems.
  5. Creating documentation and/or release notes.
- If we have to build or execute applications continuously then all the above steps needs to be repeated manually hence it is time consuming.



## BUILD AUTOMATION TOOL

- While upgrading application user needs to give parameter values as per user requirements and then actual upgradation starts.
- Hence there is a requirement to build the system; that solves this problem and speeds up the turnaround the time required to perform some of the build processes like installation of fresh, upgrade installers and report generation.

## **Bibliography**

### **Books**

1. Complete Reference Java, By Herbert Schildt.
2. Thinking in Java, By Bruce Eckel.

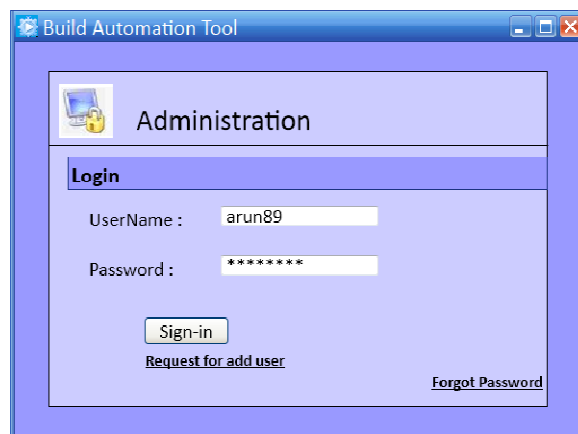
### **Links**

1. [www.Google.com](http://www.google.com)
2. [www.roseindia.com](http://www.roseindia.com)
3. [www.codeproject.com](http://www.codeproject.com)
4. <http://ant.apache.org/manual/tutorial>

**ANNEXURES 1:**  
**USER INTERFACE SCREENS**

## ANNEXURES 1: User Interface Screens

### Login Screen



The screenshot shows a window titled "Build Automation Tool" with a blue header bar. Inside the window, there is a section titled "Administration" with a lock icon. Below this, there is a "Login" section with two input fields: "UserName" containing "arun89" and "Password" containing "\*\*\*\*\*". A "Sign-in" button is located below the password field. At the bottom of the login section, there are two links: "Request for add user" and "Forgot Password".

Build Automation Tool

Administration

**Login**

UserName : arun89

Password : \*\*\*\*\*

Sign-in

[Request for add user](#)

[Forgot Password](#)

## Welcome Page



## Add User

Build Automation Tool

### Add User

**User Details**

User Name :	arun89	New Password :	*****
First Name :	Arun	Confirm Password :	*****
Last Name :	Deshpande	Email Address :	arun89@persistent.co.in
Designation :	Team Lead	Department :	BI/Analytics

**Privilege Groups**

Privileges

- Create User
- Edit User
- List User
- Create Job
- View Job Output
- Edit Job
- List Job
- Execute Job
- List Job Output

Submit Cancel

## Add Job

Build Automation Tool

Welcome ARUN  
[Logout](#)

### Add Job

**Job Details**

Job Name :

**Parameter Details**

Parameter Name :

Default Value :

Description :

**Job Parameters**

Name	Default value	Description
Log path	C:\BAT\LOG4J.log	Default log path
Log Level	DEFAULT	Default log level

**Script**

Script Path :

User Notification :

## Execute Job



The screenshot shows a window titled "Build Automation Tool" with a sub-header "Execute Job". The window contains a table of job parameters and an "Execute" button.

Build Automation Tool

Welcome ARUN  
[Logout](#)

**Job Parameters**

Parameter Name	Default Value
Log Path	C:\BAT\Log4J.log
Log Level	DEFAULT
Log Appender	Daily Rolling File Appender

Execute



## Admin User

Build Automation Tool

Welcome BATADMIN [Logout](#)

### Add User

**User Details**

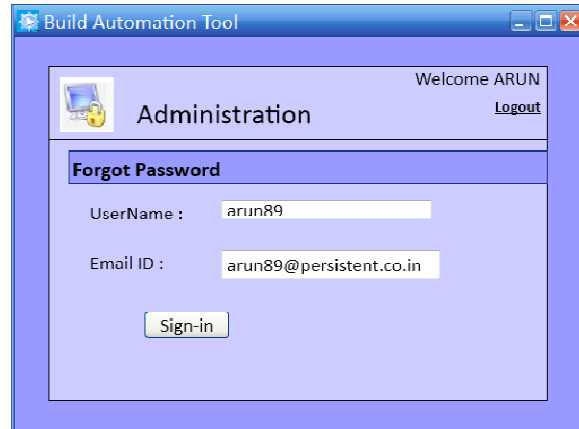
User Name :	<input type="text" value="Ashedji123"/>	New Password :	<input type="password" value="*****"/>
First Name :	<input type="text" value="Aieshwarya"/>	Confirm Password :	<input type="password" value="*****"/>
Last Name :	<input type="text" value="Shedji"/>	Email Address :	<input type="text" value="ashedji@gmail.com"/>
Designation :	<input type="text" value="Intern"/>	Department :	<input type="text" value="Development"/>

**Privilege Groups**

Privileges

<input type="checkbox"/> Create User	<input type="checkbox"/> Edit Job
<input type="checkbox"/> Edit User	<input type="checkbox"/> List Job
<input checked="" type="checkbox"/> List User	<input checked="" type="checkbox"/> Execute Job
<input checked="" type="checkbox"/> Create Job	<input type="checkbox"/> List Job Output
<input checked="" type="checkbox"/> View Job Output	

## Forgot Password



The screenshot shows a web application window titled "Build Automation Tool". The main content area is titled "Administration" and includes a "Welcome ARUN" message and a "Logout" link. Below this, there is a "Forgot Password" section with two input fields: "UserName" containing "arun89" and "Email ID" containing "arun89@persistent.co.in". A "Sign-in" button is located at the bottom of the form.

Build Automation Tool

Welcome ARUN  
[Logout](#)

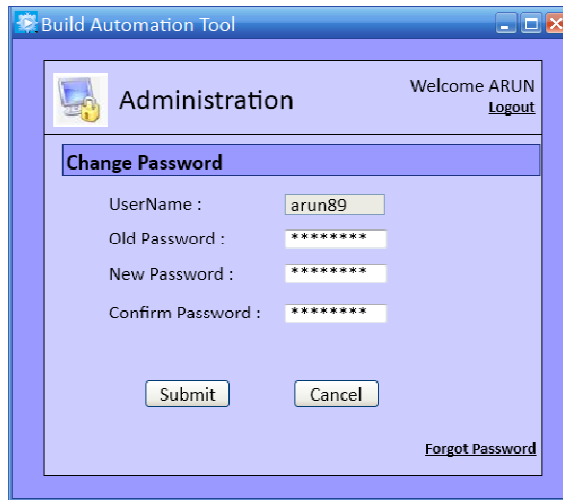
**Administration**

**Forgot Password**

UserName :

Email ID :

## Change Password



The screenshot shows a web application window titled "Build Automation Tool". The main content area is labeled "Administration" and includes a "Welcome ARUN" message with a "Logout" link. The "Change Password" section contains the following fields and controls:

UserName :	<input type="text" value="arun89"/>
Old Password :	<input type="password" value="*****"/>
New Password :	<input type="password" value="*****"/>
Confirm Password :	<input type="password" value="*****"/>

Below the form are two buttons: "Submit" and "Cancel". A "Forgot Password" link is located at the bottom right of the form area.

# History

Build Automation Tool

Welcome ARUN  
[Logout](#)

### History

Log4J

Display 05 | 10 results per |

Build No	Time	Status	Job Output
5	2 mins	Success	
4	10 mins	Failed	
3	1 day	Success	
2	2 days 3hr	Success	
1	2 days 4hr	Failed	

Display 05 | 10 results per |

[Back](#)

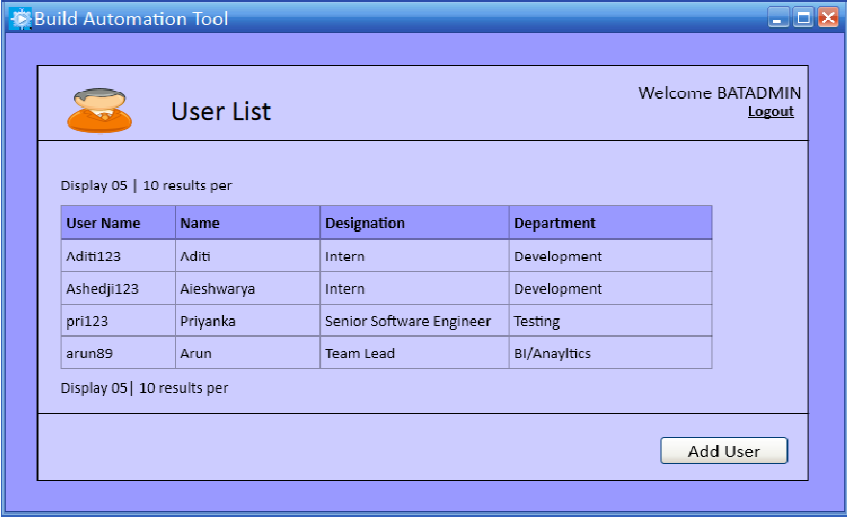
## Job List



The screenshot shows a web application window titled "Build Automation Tool". The main content area is titled "Job List" and includes a "Welcome ARUN" message with a "Logout" link. Below the header, there is a table displaying job information. The table has five columns: Name, Last Success, Last Failure, Build Stability, and Status. The jobs listed are LoginApplication, Log4J, JAXB, Integrate, and Importdata. The table is flanked by "Display 05 | 10 results per" text. At the bottom right of the interface is an "Add Job" button.

Name	Last Success	Last Failure	Build Stability	Status
LoginApplication	14 days	N/A	stable	Success
Log4J	N/A	27 days	broken since this build	Failure
JAXB	12 days	N/A	stable	Success
Integrate	2 days	N/A	stable	Success
Importdata	N/A	1hr	broken since this build	Failure

## User List



Build Automation Tool

Welcome BATADMIN  
[Logout](#)

Display 05 | 10 results per

User Name	Name	Designation	Department
Aditi123	Aditi	Intern	Development
Ashedji123	Aieshwarya	Intern	Development
pri123	Priyanka	Senior Software Engineer	Testing
arun89	Arun	Team Lead	BI/Analytics

Display 05 | 10 results per

[Add User](#)

**ANNEXURES 2:**  
**OUTPUT REPORTS WITH DATA**

## ANNEXURES 2: Output Reports with Data

### Job List



The screenshot shows a web application window titled "Build Automation Tool". The main content area is titled "Job List" and includes a "Welcome ARUN" message with a "Logout" link. Below the header, there is a table displaying job information. The table has five columns: Name, Last Success, Last Failure, Build Stability, and Status. The data rows are as follows:

Name	Last Success	Last Failure	Build Stability	Status
LoginApplication	14 days	N/A	stable	Success
Log4J	N/A	27 days	broken since this build	Failure
JAXB	12 days	N/A	stable	Success
Integrate	2 days	N/A	stable	Success
Importdata	N/A	1hr	broken since this build	Failure

Below the table, there is a "Date: 04/05/2013 Time: 11:30 AM" and an "Add Job" button.



## User List

Build Automation Tool

Welcome BATADMIN  
[Logout](#)

Display 05 | 10 results per

User Name	Name	Designation	Department
Aditi123	Aditi	Intern	Development
Ashedji123	Aishwarya	Intern	Development
pri123	Priyanka	Senior Software Engineer	Testing
arun89	Arun	Team Lead	BI/Analytics

Display 05 | 10 results per

Date: 04/05/2013 Time: 11:30 AM

# History

Build Automation Tool

Welcome ARUN  
[Logout](#)

### Log4J

Display 05 | 10 results per |

Build No	Time	Status	Job Output
5	2 mins	Success	
4	10 mins	Failed	
3	1 day	Success	
2	2 days 3hr	Success	
1	2 days 4hr	Failed	


Display 05 | 10 results per |

Date: 04/05/2013    Time: 11:30 AM    [Back](#)

# Privilege Report

Build Automation Tool

Welcome BATADMIN [Logout](#)

 **Privilege Report**

Display 05 | 10 results per

Privilege Key	Code	Privilege Name	UserName	Name	EmailID
0765	CU	Create User	Ashedji123	Aieshwarya	ashedji@gmail.com
0768	EJ	Eecute Job	Ashedji123	Aieshwarya	ashedji@gmail.com
0301	CJ	Create Job	Aditi123	Aditi	aditi_p@hotmail.com
0303	VJ	View Job	pri123	Priyanka	p_patil@gmail.com
0325	VJO	View Job Output	arun89	Arun	arun89@gmail.com

Display 05 | 10 results per

Date: 04/05/2013      Time: 11:30 AM

# Admin Status Report

The screenshot shows a web application window titled "Build Automation Tool". The main content area is titled "Status Report" and includes a welcome message "Welcome BATADMIN" and a "Logout" link. Below the header, there is a table displaying user status information. The table has five columns: Status Key, User Name, Name, Email ID, and Status Name. There are four rows of data. Below the table, there are two buttons: "Privilege Report" and "Add User". The date and time are displayed as "Date: 04/05/2013" and "Time: 11:30 AM".

Build Automation Tool

Welcome BATADMIN  
Logout

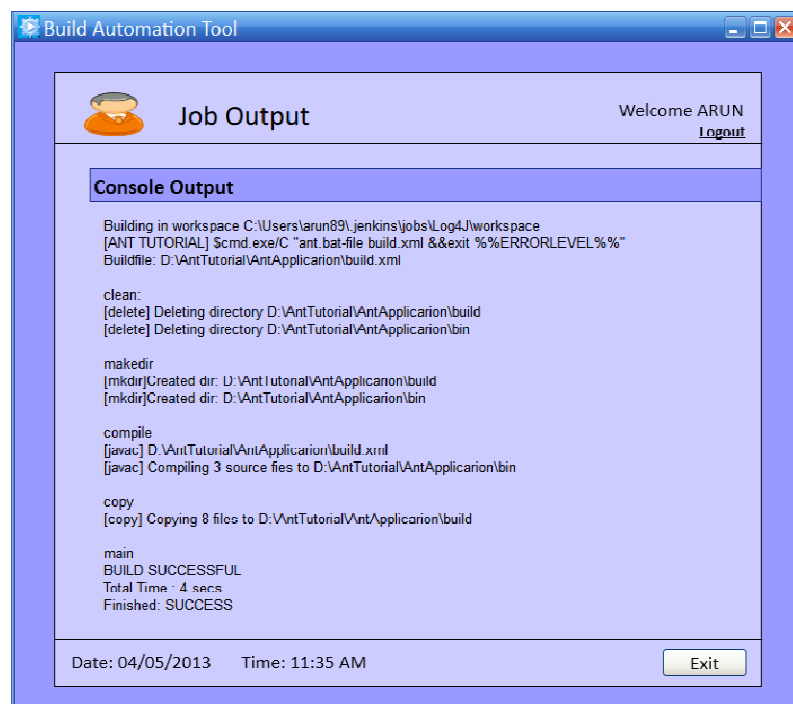
Display 05 | 10 results per

Status Key	User Name	Name	Email ID	Status Name
1029	Ashedji123	Aleshwarya	ashedji@gmail.com	Approved
1030	Aditi123	Aditi	aditi_p@hotmail.com	Approved
1031	pri123	Priyanka	p_patil@gmail.com	Pending
1032	arun89	Arun	arun89@gmail.com	Approved

Display 05 | 10 results per

Date: 04/05/2013      Time: 11:30 AM      [Privilege Report](#)      [Add User](#)

## Console Output



The screenshot shows a window titled "Build Automation Tool" with a blue header bar. Below the header, there is a "Job Output" section with a small icon of a bowl and a "Welcome ARUN" message with a "Logout" link. The main area is titled "Console Output" and contains the following text:

```
Building in workspace C:\Users\arun89\jenkins\jobs\Log4J\workspace
[ANT TUTORIAL] %cmd.exe/C %ant.bat-file build.xml &&exit %%ERRORLEVEL%%"
Buildfile: D:\AntTutorial\AntApplication\build.xml

clean:
[delete] Deleting directory D:\AntTutorial\AntApplication\build
[delete] Deleting directory D:\AntTutorial\AntApplication\bin

mkdir:
[mkdir] Created dir: D:\AntTutorial\AntApplication\build
[mkdir] Created dir: D:\AntTutorial\AntApplication\bin

compile:
[javac] D:\AntTutorial\AntApplication\build.xml
[javac] Compiling 3 source files to D:\AntTutorial\AntApplication\bin

copy:
[copy] Copying 8 files to D:\AntTutorial\AntApplication\build

main
BUILD SUCCESSFUL
Total Time : 4 secs
Finished: SUCCESS
```

At the bottom of the window, there is a status bar showing "Date: 04/05/2013 Time: 11:35 AM" and an "Exit" button.

**ANNEXURES 3:**  
**SAMPLE SOURCE CODE**

### ANNEXURES 3: Sample Source Code

```
package com.Build_Auto;

import java.io.*;

import java.util.*;

public class CDOS_Properties {

    public static void setting_Properties() throws IOException,
        NullPointerException {

        Map<String, String> env_Map = new HashMap<String,
String>();

        /* Taking already set environment variables from system */

        env_Map = System.getenv();

        inti = 0, j = 0, k = 0, b = 0;

        String environmentVariable =
System.getenv("SLAVE_HOME");

        System.out.print("Environment Variable are
=>" + environmentVariable);

        File inputFile = new
File(environmentVariable + "\\CDOS_UPGRADE\\UpgradeInstaller\\
config\\dvCDOSInstall.properties");

        String[] map_Key = new String[env_Map.size()];
```

```
String[] map_Value = new String[env_Map.size()];
String[] split_str = new String[2];
String[] split_str1 = new String[2];
String[] split = new String[2];
intlen = (int) inputFile.length();
String[] str1 = new String[len];
String random_string = null;
Set set = env_Map.entrySet();
Iterator it = set.iterator();
while (it.hasNext()) {
    Map.Entry me = (Map.Entry) it.next();
    map_Key[j] = me.getKey().toString();
    map_Value[j] = me.getValue().toString();
    j++;
}
intcountOfFileLine = 0;
FileInputStreamrandom_File = new
FileInputStream(inputFile);
BufferedReaderbr = new BufferedReader(new
InputStreamReader(random_File));
```



```

        String[] file_Content_String = new String[(int)
inputFile.length()];

        while ((random_string = br.readLine()) != null) {

            file_Content_String[countOfFileLine]
=random_string+ "\n";

            countOfFileLine++;

        }

        br.close();

        random_File.close();

        random_File.close();

        FileOutputStreamfos = new FileOutputStream(inputFile);

        BufferedWriterbw = new BufferedWriter(new
OutputStreamWriter(fos));

        for (k = 0, b = 0; k <countOfFileLine; k++) {

            String stringToBeWrittenInFile =
file_Content_String[k];

            for (i = 0; i<j; i++) {

                if (file_Content_String[k].contains("$")) {

                    split_str = file_Content_String[k].split("\\$");

                    str1[b] = (split_str[1].substring(1,
split_str[1].length() - 2));

```

```

        if (str1[b].endsWith("}")) {split_str1 =
str1[b].split("}");

        str1[++b] = split_str1[0];

        }

        if (str1[b].startsWith("WORKSPACE")) {

        split = str1[b].split("}");

        if (split[0].compareTo(map_Key[i]) == 0) {

                int index = file_Content_String[k].indexOf("=");

                stringToBeWrittenInFile =
file_Content_String[k].substring(0, index + 1)+ map_Value[i] +"\n";

                break;

        }

    }

    if (str1[b].startsWith("COMPONENT_TYPE")) {

        split = str1[b].split("}");

        if (split[0].compareTo(map_Key[i]) == 0) {

                int index = file_Content_String[k].indexOf("=");

                stringToBeWrittenInFile = file_Content_String[k]

                .substring(0, index + 1)+ "{"+map_Value[i]+"}" + "\n"

                break;

        }

    }

```

```
}  
if (str1[b].compareTo(map_Key[i]) == 0) {  
    int index = file_Content_String[k].indexOf("=");  
    stringToBeWrittenInFile = file_Content_String[k]  
        .substring(0, index + 1)+ map_Value[i] + "\n";  
    break;  
    }  
}  
}  
bw.append(stringToBeWrittenInFile);  
}  
bw.close();  
fos.close();  
}  
public static void main(String a[]) throws IOException {  
    setting_Properties();  
}  
}
```

```
<?xml version="1.0" encoding="UTF-8"?>
<project name="Copy to User Content" default="main" basedir=".">
    <!-- Sets variables which can later be used. -->
    <!-- The value of a property is accessed via ${} -->
<property environment="env"/>
<property name="Run-Installer-location"
location="${env.SLAVE_HOME}"/>
<property name="build-unzip" location="${Run-Installer-
location}\CDOS_UPGRADE"/>
<property name="build-copy-unzip" location="${Run-Installer-
location}\workspace\${env.JOB_NAME}"/>
<property name="unzip-Run-Installer" location="${Run-Installer-
location}\workspace\${env.JOB_NAME}\Unzip_Run_Installer"/>
<target name="mkdir">
    <echo message="Making Directories ${env.JOB_NAME}"/>
    <mkdir dir="${build-unzip}"/>
</target>
<target name="remove">
    <echo message="Deleting Directories"/>
    <delete dir="${build-unzip}\CDOS-ear" />
    <delete dir="${build-unzip}\Build Automation" />
    <delete dir="${build-unzip}\UpgradeInstaller" />
</target>
<target name="Run-zip" depends="remove,mkdir">
```

```
<echo message="Making directory for unzipping
Run_Installer.zip"/>
<mkdir dir="${unzip-Run-Installer}"/>
<echo message="Unzipping Run_Installer.zip"/>
<unzip dest="${unzip-Run-Installer}" src="${build-copy-
unzip}\Run_Installer.zip"/>
<echo message="Unzipping CDOS ear"/>
<unzip dest="${build-unzip}\CDOS-ear" src="${unzip-Run-
Installer}\CDOS-ear.zip"/>
<echo message="Unzipping Build Automation (Java Program)"/>
<unzip dest="${build-unzip}\Build Automation" src="${unzip-
Run-Installer}\Build Automation.zip"/>
<echo message="Unzipping Upgrade Installer"/>
<unzip dest="${build-unzip}\UpgradeInstaller" src="${unzip-
Run-Installer}\UpgradeInstaller.zip"/>
<echo message="Deleting property file "/>
<delete file="${build-
unzip}\UpgradeInstaller\config\dvCDOSInstall.properties"/>
<echo message="Copying property file "/>
<copy todir="${build-unzip}\UpgradeInstaller\config">
<fileset file="${build-unzip}\Build
Automation\dvCDOSInstall.properties"/>
</copy>
</target>
<target name="copy-ear">
<echo message="copying files in cdos-ear.xml"/>
```

```
        <copy todir="D:\jboss-5.1.0.GA\server\default\deploy\dv-
cdos.ear">
            <filesetdir="${build-unzip}\CDOS-ear"/>
        </copy>
    </target>
    <target name="run-java">
        <echo message="Running Java replacing property code"/>
        <ant antfile= "${build-unzip}\Build
Automation\src\com\Build_Auto\build.xml" />
    </target>
    <target name="run-installer" >
        <echo message="Running Installer"/>
        <java jar="${build-unzip}\UpgradeInstaller\upgradeInstaller.jar"
fork="true" failonerror="true" />
    </target>
    <target name="main" depends="Run-zip,copy-ear,run-java,run-installer">
    </target>
</project>
```