**[Java Design Patterns](http://www.journaldev.com/1827/java-design-patterns-example-tutorial)**

**Creational Design Patterns**

1. [Singleton](http://www.journaldev.com/1377/java-singleton-design-pattern-best-practices-examples)
2. [Factory](http://www.journaldev.com/1392/factory-design-pattern-in-java)
3. [Abstract Factory](http://www.journaldev.com/1418/abstract-factory-design-pattern-in-java)
4. [Builder](http://www.journaldev.com/1425/builder-design-pattern-in-java)
5. [Prototype](http://www.journaldev.com/1440/prototype-design-pattern-in-java)

**Structural Design Patterns**

1. [Adapter](http://www.journaldev.com/1487/adapter-design-pattern-java)
2. [Composite](http://www.journaldev.com/1535/composite-design-pattern-in-java)
3. [Proxy](http://www.journaldev.com/1572/proxy-design-pattern)
4. [Flyweight](http://www.journaldev.com/1562/flyweight-design-pattern-java)
5. [Facade](http://www.journaldev.com/1557/facade-design-pattern-in-java)
6. [Bridge](http://www.journaldev.com/1491/bridge-design-pattern-java)
7. [Decorator](http://www.journaldev.com/1540/decorator-design-pattern-in-java-example)

**Behavioral Design Patterns**

1. [Template Method](http://www.journaldev.com/1763/template-method-design-pattern-in-java)
2. [Mediator](http://www.journaldev.com/1730/mediator-design-pattern-in-java-example-tutorial)
3. [Chain of Responsibility](http://www.journaldev.com/1617/chain-of-responsibility-design-pattern-in-java)
4. [Observer](http://www.journaldev.com/1739/observer-design-pattern-in-java)
5. [Strategy](http://www.journaldev.com/1754/strategy-design-pattern-in-java-example-tutorial)
6. [Command](http://www.journaldev.com/1624/command-design-pattern)
7. [State](http://www.journaldev.com/1751/state-design-pattern-java)
8. [Visitor](http://www.journaldev.com/1769/visitor-design-pattern-java)
9. [Interpreter](http://www.journaldev.com/1635/interpreter-design-pattern-in-java-example-tutorial)
10. [Iterator](http://www.journaldev.com/1716/iterator-design-pattern-java)
11. [Memento](http://www.journaldev.com/1734/memento-design-pattern-java)

**Miscellaneous Design Patterns**

1. [Dependency Injection](http://www.journaldev.com/2394/java-dependency-injection-design-pattern-example-tutorial)
2. [Thread Safety in Java Singleton](http://www.journaldev.com/171/thread-safety-in-java-singleton-classes-with-example-code)

I have been involved 2 projects in this assessment

**Project-1:**

1. Individually worked on Agent Simulator to provide the proxy agent to talk with color master. In this assessment, I was use the Spring4 for rest resource implementation, shell script for agent proxy creation and AngularJs for UI.
2. Apart from this I attend the Spring boot and micro services training, POC done on the same with CRUD operation

**Project-2:**

1. Get the input from On-site team , prepare the PDD and SOW with proper time estimation
2. Currently working on Servlets , JSP and Spring Scheduling for the order updating
3. Struts migration working as individual , apart from this involved in Security vulnerabilities in Marcom generic Ordering application

Typically following are the activities that I do every day

1. Attend the scrum call with Onsite team
2. Assign the task to individuals and monitor the work status
3. Sync the code to perforce on day basis and deploy the code on server weekly basis using team city builder tool
4. Update the deliverables to respective team and reporting managers

So all in all, I have been one of the driving force for this project.

Overall Rating: Highly exceeded the Expected results.

I can term my overall rating as "Accelerating Acceleration"