Accenture DevOps Platform (ADOP)

**ADOP** is a cloud based pre-configured and Integrated Development Workstation for configuration management, Continuous integration, Continuous delivery, Test tooling, Deployment, Dev and Test Environments. **ADOP** is a blueprint for DevOps/Continuous Delivery concepts, principles and best technical practices.

The core tools used in **ADOP** are as follows  
 **1. CHEF  
 2. CONFLUENCE  
 3. JIRA AGILE  
 4. GIT  
 5. JENKINS  
 6. DOCKER  
 7. CUCUMBER  
 8. ANSIBLE  
 9. SONARQUBE  
10. SENSU  
11. SOAPUI   
12. GERRIT  
13. KIBANA  
14. SELENIUM GRID  
15. ELASTICSEARCH  
16. OWASP ZAP  
17. LOGSTASH**

**CHEF**

Chef is a configuration management tool written in Ruby and Erlang that allows us to write Infrastructure as Code. It uses a pure-Ruby, [domain-specific language](https://en.wikipedia.org/wiki/Domain-specific_language) (DSL) for writing system configuration "recipes". Chef is used to streamline the task of configuring and maintaining a company's servers, and can integrate with cloud-based platforms such as [Internap](https://en.wikipedia.org/wiki/Internap), [Amazon EC2](https://en.wikipedia.org/wiki/Amazon_EC2), [Google Cloud Platform](https://en.wikipedia.org/wiki/Google_Cloud_Platform), [OpenStack](https://en.wikipedia.org/wiki/OpenStack), [SoftLayer](https://en.wikipedia.org/wiki/SoftLayer), [Microsoft Azure](https://en.wikipedia.org/wiki/Microsoft_Azure) and [Rackspace](https://en.wikipedia.org/wiki/Rackspace) to automatically provision and configure new machines.

**Advantages:**

* We can automate an entire infrastructure using Chef. All tasks that were manually being done, can now be done via Chef tool.
* We can configure thousands of nodes within minutes using Chef.
* Chef automation works with most of the public cloud offerings like [***AWS***](https://www.edureka.co/blog/amazon-aws-tutorial/).
* Chef will not only automate things, but will also keep the systems under consistent check, and confirm that the system is in fact configured the way it is required (Chef Agent/Client does this job). If somebody makes a mistake by modifying a file, Chef will correct it.
* An entire infrastructure can be recorded in the form of a Chef repository, that can be used as a blueprint to recreate the infrastructure from scratch.

**Best Practices for CHEF:**

* **Project creation and “readme driven development”**It is very important to create a project which should be source controlled. This helps in storing the codes and for verification or to see step to step process of development as well as mistakes, which can be viewed from the commit history. Using Readme Driven Development is top best practice that can be started by**README.md** file. This practice is important as it helps we to focus wer mind and cause we think like a user.
* **Continuous integration with Travis**Configuring Travis helps in eliminating development gem dependencies so as to pace things up and tests on the CI server. Furthermore, it also tests the master as well as the development branches.
* **Workflow with Spork**It is another practice in Chef. This tool helps in collaborative cookbook development. It also helps single developer to control and place things in order when it comes to bumping versions.
* **Verification with Minitest**Helps verify the cookbooks on the real servers.

**CONFLUENCE**

**Confluence** is a team [collaboration software](https://en.wikipedia.org/wiki/Collaboration_software). Written in [Java](https://en.wikipedia.org/wiki/Java_(programming_language)) and mainly used in corporate environments. An enterprise wiki that can be used for capturing information such as requirements, how-to guides and meeting note. It is a powerful way of sharing knowledge across a team. Team members can create, share and collaborate information.

**ADVANTAGES:  
  
1. Intranet -** internal company portal documentation site to create share and collaborate information or to centralize technical documentation, policy procedure and knowledge base articles to better support and service internal 'customers'.   
**2. Extranet** – we can also use Confluence as a public documentation site.  
**3. Project Documentation** - we use Confluence internally to document and collaborate with our clients for projects we work on, keeping meeting minutes, design docs, etc.

**Best practices for Confluence:  
  
1. Resting access to pages**It ensure that pages with sensitive content are limited to viewing and/or changing by specified individuals.

**2. Content That needs to Be Protected should not be Stored in Confluence**Although precautions are taken to ensure Confluence is secure, it is still possible for the information in Confluence to be compromised, due to either an application bug or misinterpreted permissions. Ideally, for security and privacy, sensitive information should not be stored in Confluence, either within a page or as an attachment. If the data is needed to collaborate, access to that content should be **heavily** restricted to only the users that have a business need to know it, and then **removed** when it is no longer needed.

**JIRA Agile**

An application Lifecyle management tool that allows we to prioritize, assign, track, report and audit wer ‘issues’, from software bugs and helpdesk tickets to project tasks and change requests.

**ADVANTAGES**

1. It helps us by allowing us to filter options. For instance, we can filter out all the issues in our backlog which are not high priority. We may also want to filter on things such as cards with a high business value.

2. JIRA Agile allows us to set up rankings for our issues to help us organize tasks in our product/sprint backlog more effectively. Rankings allow us to prioritize issues at a more granular level.

3. It provides charts which helps us to learn from our previous encounters. We can use the estimates from our completed tasks to guess the likelihood of our remaining estimates being correct.

4. From the charts we can ascertain a teams' "velocity" — that is, the speed at which the team is able to complete the tasks estimated in units of difficulty (hours/complexity/score out of 10). Once we know a team's velocity we can then calculate, based on the current team performance, how long the remaining tasks should take to complete.

**GIT**

**Git** is a [version control system](https://en.wikipedia.org/wiki/Version_control_system) 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 [source code management](https://en.wikipedia.org/wiki/Source_code_management) in [software development](https://en.wikipedia.org/wiki/Software_development), but it can be used to keep track of changes in any set of files. It is a [distributed revision control](https://en.wikipedia.org/wiki/Distributed_revision_control) system that (allows for multiple branches which are easily created, merged and deleted) aims at speed, data integrity,[and support for distributed, non-linear workflows.

**ADVANTAGES:**

**1.Data redundancy** and replication  
**2.Distributed model**   
This means wer work is wer own. We can let others see only what is necessary. Not everything has to be public. There are other advantages to the distributed model, such as the speed (since most everything is local) and possibility of working offline

**3.Branching and merging are easy**  
 Branching is a walk in the park. It feels like a natural part of the workflow. They are cheap (fast and consume very little space) so that we can branch whenever we want. This means we can sandbox wer features and ideas till they are ready for the mainstream.  
**4. High availability**

**5.Superior disk utilization** and network performance  
**6.Collaboration friendly**

**7.Any sort of projects can use GIT**

**Best Practices with GIT:**

**1**.**Useful Commit messages**  
Creating insightful and descriptive commit messages is one of the best things we can do for others who use the repository. It lets people quickly understand changes without having to read code.

**2.Integration with external tools**For Example: **Web Views**-Setting up a tool like gitweb (or cgit or whatever) to allow URL reference to commits (among other visualization interfaces it provides) gives people a great way to refer to commits in email and conversations.  
**Bug Tracking**- integrating our bug tracking system with git makes the systems one thousand times more effective.

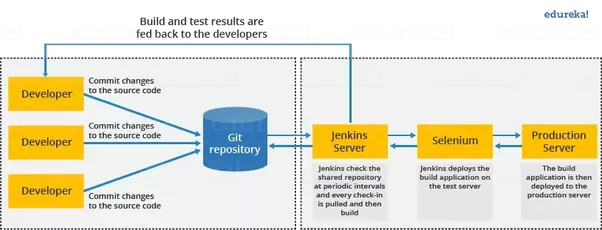
**3.Dividing Work into repositories**  
For example creating one conceptual group per repository or creating Separate repositories for files which might be needed by multiple projects or for large binary files is considered one of the best practices.

**4.Review Code before Checkin with git diff**.

**5.Keeping our repositories up-to-date** and **periodic maintenance** of it like Cleaning up our git repro every so often and checking our stash for forgotten work (git stash list).

**Jenkins**

Jenkins is an open source automation tool written in Java with plugins built for Continuous Integration purpose. With Jenkins, organizations can accelerate the software development process through automation. Jenkins integrates development life-cycle processes of all kinds, including build, document, test, package, stage, deploy, static analysis and much more. Jenkins triggers a build for every change made in the source code repository for example Git repository. Once the code is built it deploys it on the test server for testing. Concerned teams are constantly notified about build and test results. Finally, Jenkins deploys the build application on the production server.



**ADVANTAGES**

**1**.It is **open source** and it is **user-friendly**, easy to install and does not require additional installations or components.  
**2**. It is **free of cost**.  
**3**. **Easily Configurable**. Jenkins can be easily modified and extended. It deploys code instantly, generates test reports. Jenkins can be configured according to the requirements for continuous integrations and continuous delivery.  
**4.** **Platform Independent**. Jenkins is available for all platforms and different operating systems, whether OS X, Windows or Linux.  
**5. Rich Plugin ecosystem**. The extensive pool of plugins makes Jenkins flexible and allows building, deploying and automating across various platforms.  
**6. Easy support**. Because it is open source and widely used, there is no shortage of support from large online communities of [agile teams](https://apiumhub.com/tech-blog-barcelona/building-agile-team/).  
**7.**Developers **write the tests to detect the errors** of their code as soon as possible. So, the developers don’t waste time on large-scale error-ridden integrations.  
**8. Issues are detected and resolved almost right away** which keeps the software in a state where it can be released at any time safely.  
**9.** Most of the integration work is **automated.** Hence fewer integration issues. This saves both time and money over the lifespan of a project

**Best practices for Jenkins:**

1.**Set up version control** of job configurations Keep all scripts in version control - avoid running scripts that live on the Jenkins server filesystem

2. **Don't install unnecessary plugins** - plugins are often written by third parties and can interact with each other in strange ways

3. **Backup Jenkins Home** regularly.  
Between archived builds, build logs that let we determine exactly what happened, and the SCM history information that tells we exactly what was built, Jenkins contains a lot of information we don't want to lose.

4. **Integrate tightly** with wer issue tracking system, like JIRA or Bugzilla, to reduce the need for maintaining a Change Log

5. **Prevent resource collisions in jobs that are running in parallel**.

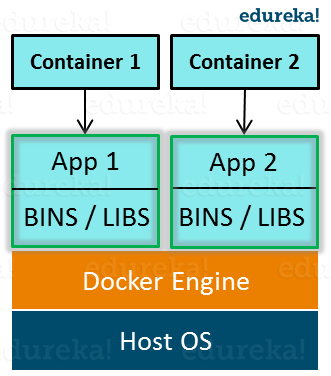
Multiple jobs running at the same time often cause collisions if they set up some kind of service, or need exclusive access. If our builds involve use of databases or other networked services, we need to ensure that they don't interfere with each other. Allocate a different port for parallel project builds to avoid build collisions. If that's not possible (e.g. in the case of a persistent resource that needs to be locked) we can prevent builds that use it from running at the same time using e.g. [Throttle Concurrent Builds Plugin](file:///C:\display\JENKINS\Throttle+Concurrent+Builds+Plugin).

6**. Set up email notifications** mapping to ALL developers in the project, so that everyone on the team has his pulse on the project's current status.

7. **Use "file fingerprinting" to manage dependencies**.  
When we have interdependent projects on Jenkins, it often becomes hard to keep track of which version of this is used by which version of that. Jenkins supports "file fingerprinting" to simplify this, so make best use of it.

**DOCKER**

Docker is a containerization platform that packages wer application and all its dependencies together in the form of Containers to ensure that wer application works seamlessly in any environment. It is designed to benefit both developers and system administrators, making it a part of many DevOps (developers + operations) toolchains. For developers, it means that they can focus on writing code without worrying about the system that it will ultimately be running on. It also allows them to get a head start by using one of thousands of programs already designed to run in a Docker container as a part of their application. For operations staff, Docker gives flexibility and potentially reduces the number of systems needed because of its small footprint and lower overhead



**ADVANTAGES:**

**1.Multi-Cloud Platforms**  
One of Docker’s greatest benefits is portability. Over last few years, all major cloud computing providers, including Amazon Web Services (AWS) and Google Compute Platform (GCP), have embraced Docker’s availability and added individual support. Docker containers can be run inside an Amazon EC2 instance, Google Compute Engine instance, Rackspace server or VirtualBox, provided that the host OS supports Docker

**2. Environment Standardization and Version Control**   
Docker containers ensure consistency across multiple development and release cycles, standardizing our environment. On top of that, Docker containers work just like GIT repositories, allowing us to commit changes to our Docker images and version control them. Suppose, we perform a component upgrade that breaks our whole environment. It is very easy to roll back to a previous version of our Docker image

**3.Isolation**   
Docker ensures wer applications and resources are isolated and segregated. Docker ensures that applications that are running on containers are completely segregated and isolated from each other, granting us complete control over traffic flow and management. No Docker container can look into processes running inside another container.

**4. Security**   
Docker uses host OS sensitive mount points as read-only mount points and uses a copy-on-write filesystem to make sure containers can’t read each other’s data. It also limits system calls to wer host OS and works well with SELinux and AppArmor. Additionally, [Docker images](http://iamondemand.com/blog/how-ironsource-scales-node-js-with-docker-to-support-millions-of-daily-users/) that are available on Docker Hub are digitally signed to ensure authenticity. Since Docker containers are isolated and resources are limited, even if one of wer applications is hacked, it won’t affect applications that are running on other Docker containers.

**5. Continuous Deployment and Testing**   
There are always minor differences between environments in development and release lifecycles, unless we have wer own private repository environment with tight checks in place. These differences may be because of different package versions or dependencies. Nevertheless, Docker can address that gap by ensuring consistent environments from development to production. Docker containers are configured to maintain all configurations and dependencies internally. As a result, we can use the same container from development to production making sure there are no discrepancies or manual intervention.

**Best practices for docker**

**1. Beware of inheritance and dependencies**

Our containers inherit a parent image that generally includes its base operating system and dependencies—things like dependent packages, default users, etc. Those inherited attributes and dependencies might expose our containers to unnecessary risk. Make sure we’re aware of the inherited attributes and take any additional steps necessary to further isolate and protect wer containers.

**2. Limit container interaction**

Container security has emerged as a serious concern for many organizations—specifically how containers interact with one another and with the outside world. Wer containers should not accept connections on exposed ports through any network interface. We should take steps both to control how—and how much—containers interact with each other internally, and limit the number of containers that have contact with the outside world so we can minimize exposure to external risks.

**3. Monitor containers for vulnerabilities**

One of the challenges of using a code repository like Docker Hub is that once a container image is uploaded to the repository nobody takes responsibility for keeping it patched and secure. It might be fine when originally created, but over time new vulnerabilities and exploits are discovered and we need to scan for those before using containers in production. A [tool like Twistlock](http://containerjournal.com/2015/07/01/embrace-docker-containers-without-compromising-on-security/) can help we monitor for and identify vulnerabilities in wer container images.

**4. Run containers as read only where possible**

One of the best and simplest ways to limit exposure to risk for container is to run them in read-only mode. That obviously won’t work for all containers—there will be containers that must accept input of some sort in order for apps to work, but containers that can be run in read-only mode should be. We should also never run containers in privileged mode.

**5**.**Keep it simple**

We should try to keep our Docker container ecosystem as simplified as possible. We should run processes in separate, individual containers. If there are services that are dependent on one another we should use the container linking feature to connect two containers rather than combining them in the same Docker container. We should also focus on keeping the footprint of containers small—don’t load unnecessary packages or services that just make the file larger and waste resources—and make sure that our containers are designed to be easy to replace. Container ecosystems tend to be very volatile and the containers should be easy to delete and recreate as necessary.

**CUCUMBER**

Cucumber is a testing tool that supports Behavior Driven Development (BDD) framework. It defines application behavior using simple English text, defined by a language called Gherkin. Cucumber allows automation functional validation that is easily read and understood. Cucumber was initially implemented in Ruby and then extended to Java framework

**ADVANTAGES:**

1.Cucumber supports different languages like Java.net and Ruby.  
2. It acts as a bridge between the business and technical language. We can accomplish this by creating a test case in plain English text.  
3. It allows the test script to be written without knowledge of any code, it allows the involvement of non-programmers as well.  
4. It serves the purpose of end-to-end test framework unlike other tools.  
5. Due to simple test script architecture, Cucumber provides code reusability.

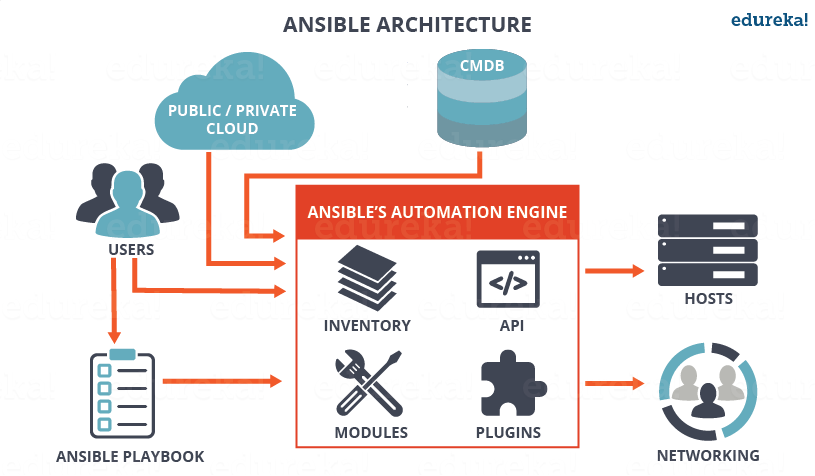
**Best practices with cucumber**

**1.Write declarative features**  
Scenarios should be written like a user would describe them. Beware of scenarios that only describe clicking links and filling in form fields, or of steps that contain code or CSS selectors. This is just another variant of programming, but certainly not a feature description. Declarative features are vivid, concise and contain highly maintainable steps.  
**2.Insert a narrative**   
Narratives describe in about one sentence what a feature does. Typical narratives contain a benefit for the user, a role that needs the feature and the feature itself. Narratives are important to envision why we are implementing a feature in the first place. They also give a brief overview of the feature so others get a rough understanding what it is about without reading the scenarios.  
**3. Avoid conjunctive step**  
When we encounter a Cucumber step that contains two actions conjuncted with an “and”, we should probably break it into two steps. Sticking to one action per step makes our steps more modular and increases reusability. This is not a general rule though. There may be reasons for conjunctive steps. However, most of the time it’s best to avoid them.  
**4. Reuse step definitions**  
In Cucumber we can reuse steps in other steps. This comes in handy when a step extends another step’s behavior or defines a superior behavior that consists of multiple steps. We should try to reuse steps as often as possible. This will improve the maintainability of our app: If we need to change a certain behavior, we just need to change a single step definition  
**5.Use backgrounds wisely**  
If we use the same steps at the beginning of all scenarios of a feature, put them into the feature’s Background. Background steps are run before each scenario. But take care that we don’t put too many steps in there as our scenarios may become hard to understand.

**ANSIBLE**

Ansible is an open source IT Configuration Management, Deployment & Orchestration tool. It aims to provide large productivity gains to a wide variety of automation challenges. This tool is very simple to use yet powerful enough to automate complex multi-tier IT application environments. Ansible is usually grouped along with other Configuration Management tools like Puppet, Chef, SaltStack etc. Ansible is not just limited to Configuration Management. It can be used in many ways too such as in Provisioning, Application Deployment, Security and Compliance, Orchestration.

**Ansible Architecture**



**ADVANTAGES**

**1.Simple**Ansible uses a simple syntax written in YAML called **playbooks**. YAML is a human-readable data serialization language  
**2.Agentless**  
There are no agents/software or additional firewall ports that we need to install on the client systems or hosts which we want to automate. We do not have to separately set up a management infrastructure which includes managing our entire systems, network and storage. Ansible further reduces the effort required for our team to start automating right away.  
**3**. **Powerful & Flexible** Ansible has powerful features that can enable we to model even the most complex IT workflows. In this aspect, Ansible’s batteries included approach can manage the infrastructure, networks, operating systems and services that we are already using, as Ansible provides we with hundreds of modules to manage them. Together Ansible’s capabilities allow we to orchestrate the entire application environment regardless of where it is deployed.  
**4. Efficient**   
No extra software on our servers means more resources for our applications. Also, since Ansible modules work via JSON, Ansible is extensible with modules written in a programming language we already know. Ansible introduces modules as basic building blocks for our software. So, we can even customize it as per our needs.  
 **Best practices with Ansible**  
1. Content should be organized  
2. Plays and Tasks should be Named  
 Always name the plays and tasks. Adding name with a human meaningful description better communicates the intent to users when running a play.   
3. Native YAML Syntax should be used  
4. Debugging Messages should be cleaned up.

**SonarQube**  
SonarQube (formerly Sonar)is an [open source](https://en.wikipedia.org/wiki/Open_source) platform for continuous inspection of [code quality](https://en.wikipedia.org/wiki/Software_quality) to perform automatic reviews with static analysis of code to detect bugs, code smells and security vulnerabilities on 20+ [programming languages](http://docs.sonarqube.org/display/PLUG/Plugin+Library) including [Java (including Android)](https://en.wikipedia.org/wiki/Java_(programming_language)), [C#](https://en.wikipedia.org/wiki/C_Sharp_(programming_language)), [PHP](https://en.wikipedia.org/wiki/PHP), [JavaScript](https://en.wikipedia.org/wiki/JavaScript), [C](https://en.wikipedia.org/wiki/C_(programming_language))/[C++](https://en.wikipedia.org/wiki/C%2B%2B), [COBOL](https://en.wikipedia.org/wiki/COBOL), [PL/SQL](https://en.wikipedia.org/wiki/PL/SQL), [PL/I](https://en.wikipedia.org/wiki/PL/I), [ABAP](https://en.wikipedia.org/wiki/ABAP), [VB.NET](https://en.wikipedia.org/wiki/Visual_Basic_.NET), [VB6](https://en.wikipedia.org/wiki/Visual_Basic), [Python](https://en.wikipedia.org/wiki/Python_(programming_language)), [RPG](https://en.wikipedia.org/wiki/IBM_RPG), [Flex](https://en.wikipedia.org/wiki/Apache_Flex), [Objective-C](https://en.wikipedia.org/wiki/Objective-C), [Swift](https://en.wikipedia.org/wiki/Swift_(programming_language)), [Web](https://en.wikipedia.org/wiki/HTML) and [XML](https://en.wikipedia.org/wiki/XML). SonarQube offers reports on [duplicated code](https://en.wikipedia.org/wiki/Duplicate_code), [coding standards](https://en.wikipedia.org/wiki/Programming_style), [unit tests](https://en.wikipedia.org/wiki/Unit_testing), [code coverage](https://en.wikipedia.org/wiki/Code_coverage), [code complexity](https://en.wikipedia.org/wiki/Cyclomatic_complexity), [comments,](https://en.wikipedia.org/wiki/Comment_(computer_programming)) [bugs](https://en.wikipedia.org/wiki/Defensive_programming), and security vulnerabilities. SonarQube can record metrics history and provides evolution graphs. SonarQube's greatest asset is that it provides fully automated analysis and integration with [Maven](https://en.wikipedia.org/wiki/Apache_Maven), [Ant](https://en.wikipedia.org/wiki/Apache_Ant), [Gradle](https://en.wikipedia.org/wiki/Gradle), [MSBuild](https://en.wikipedia.org/wiki/MSBuild) and [continuous integration](https://en.wikipedia.org/wiki/Continuous_integration) tools ([Atlassian Bamboo](https://en.wikipedia.org/wiki/Bamboo_(software)), [Jenkins](https://en.wikipedia.org/wiki/Jenkins_(software)), [Hudson](https://en.wikipedia.org/wiki/Hudson_(software)), etc.).SonarQube also integrates with [Eclipse](https://en.wikipedia.org/wiki/Eclipse_(software)), [Visual Studio](https://en.wikipedia.org/wiki/Microsoft_Visual_Studio) and [IntelliJ IDEA](https://en.wikipedia.org/wiki/IntelliJ_IDEA) development environments through the SonarLint plugins and integrates with external tools like [LDAP](https://en.wikipedia.org/wiki/LDAP), [Active Directory](https://en.wikipedia.org/wiki/Active_Directory), [GitHub](https://en.wikipedia.org/wiki/GitHub), etc. SonarQube is expandable with the use of [plugins](https://en.wikipedia.org/wiki/Plugins).

## **ADVANTAGES**: **1. Detects and Alerts:**

SonarQube reduces the risk of software development within a very short amount of time. It detects bugs in the code automatically and alerts developers to fix them before rolling it out for production.

## **2. Sustainability**

SonarQube platform significantly increases the lifetime of applications by reducing complexities, duplications and potential bugs in the code, by keeping neat and clean code architecture and increased unit tests.

## **3. Productivity**

SonarQube increases productivity by enabling development teams to detect and muzzle duplication and redundancy of code.

## **4. Raise Quality**

## SonarQube can perform as a multi-dimensional analyst and can inform on seven sections of code quality. For the better quality, it avoids duplicate code, keeps code complexity low and increases coverage by units. It can determine violation of code standards and helps software development team to abolish bugs. It maintains high-quality architecture, enforces coding standards and document APIs.

## **5. Enable Continuous Code Quality Management**

With SonarQube, analysis of code becomes easier and developers receive valuable insights to ensure that this is broadly adopted. Code quality becomes a part of development process and development teams. By enabling continuous code quality management, the software quality is raised and decreases the cost and risk of software management.

**SENSU**

Sensu is an open-core monitoring platform that empowers organizations to compose comprehensive monitoring & telemetry solutions that meet their unique business requirements. By providing a platform to build upon, Sensu enables we to focus on what to monitor and measure, rather than how. It is considered to be a single monitoring platform for the entire business which comprises of Servers, Applications, Containers, Network devices, Remote Resources, and Services.

**ADVANTAGES:**

**1. Execute Service Checks**   
Sensu can monitor application and system services, detecting those in an unhealthy state.   
**2. Send Notifications**   
It also notifies our team about events before our customers do.   
**3. Collect Metrics**   
Sensu simplifies the process of gaining insightful metrics from complex distributed systems   
**4. Documented API**   
Sensu’s API provides access to event and client data, the ability to request check executions, and resolve events. The API also provides a key/value store which can be leveraged to extend Sensu's functionality in a variety of ways.  
**5. Self-Service Monitoring**   
Sensu provides support for [centralized](https://sensuapp.org/docs/latest/reference/checks.html#subscription-checks) and [decentralized](https://sensuapp.org/docs/latest/reference/checks.html#standalone-checks) (or distributed) monitoring, enabling operations teams to maintain a standard service level for the entire organization without placing unnecessary restrictions on developers.

**6. Secure Connectivity S**  
Sensu leverages transports that offer SSL encryption, authentication, and granular ACLs. Sensu's connections traverse complex network topologies, including those that use NAT and VPNs.

**7. Reuse existing monitoring logic**   
Sensu allows us to reuse monitoring checks and plugins from legacy monitoring tools like **Nagios**, **Icinga**, **Zabbix**, and many more. Sensu was designed from day one as a replacement for an aging Nagios installation, and to this day monitoring plugin compatibility remains as one of Sensu's most compelling features

**SoapUI**

**SoapUI** is an [open-source](https://en.wikipedia.org/wiki/Open-source_software) web service testing application for [service-oriented architectures](https://en.wikipedia.org/wiki/Service-oriented_architecture) (SOA) and [representational state transfers](https://en.wikipedia.org/wiki/Representational_state_transfer) (REST). SOAP UI can be used to test complete RESTful API and SOAP Web Service testing. It

supports Functional Testing, Performance Testing, Interoperability Testing, Regression

Testing, Load Testing, and much more.

It is user friendly as well as it is easy to convert functional test into non-functional tests

such as Load, Stress testing.

SOAP UI is rich in the following five aspects:

1. Functional Testing

2. Security Testing

3. Load Testing

4. Protocols and Technologies

5. Integration with other tools

**ADVANTAGES**  
1. It can **perform** the role of both client and service.

2. It enables the users to **create functional and non-functional tests** quickly and in an

efficient manner using a single environment.

3. It is licensed under the terms of the GNU Leaser General Public License (LGPL).

4. It is purely implemented using JAVA platform.

5. It **supports** Windows, Mac, multiple Linux dialects.

6. It allows testers to **execute automated** functional, regression, compliance, and load

tests on different Web API.  
7. It **supports all the standard protocols** and technologies to test all kinds of APIs.

**GERRIT**

It is a code review system developed for the Git version control system. It is a web based code review system, facilitating online code reviews for projects. Gerrit is a Git server which adds a fine-grained access control system and a code review system and workflow. The user interface of Gerrit is based on *Google Web Toolkit*.   
Its uses JGit library to provide the Git functionality. A developer can use Gerrit to suggest a change. Other developers can review the change and suggest improvements. If a Gerrit change needs improvement, it is possible to update it with a new commit. Once the suggested changes are accepted, they can be merged to the target branch of the Git repository via the Gerrit user interface.  
Gerrit makes code reviews easier by showing changes in a side-by-side display. It also supports to display the change as a unified diff which is often easier to read on smaller screens.

**ADVANTAGES**1. It is a lightweight framework for reviewing every commit  
2. We can easily find the error in the source code using Gerrit.

3. We can work with Gerrit, if we have regular Git client; there is no need to install any Gerrit client.

4. Gerrit can be used as an intermediate between developers and git repositories  
5. Gerrit provides access control for Git repositories and web frontend for code review.

6. We can push the code without using additional command line tools.

7. Gerrit can allow or decline the permission on the repository level and down to the

branch level.  
8. Gerrit is supported by Eclipse

**Best practices with Gerrit**   
1. Avoid pushing untested work to a Gerrit server  
2.Keep track of the changes  
3.local branches should be used for each feature or issue we are working on

**KIBANA**

Kibana is an open-source data visualization and exploration tool used for log and time series analytics, application monitoring, and operational intelligence use cases  
Kibana is also popular due to its powerful and easy-to-use features such as histograms, line graphs, pie charts, heat maps and built-in geospatial support Kibana is designed to work in conjunction with Elasticsearch to make huge and complex data streams more quickly and easily understandable through graphic representation

**ADVANTAGES**

**1.Interactive Chats**Kibana offers intuitive charts and reports that we can use to interactively navigate vast amounts of log data. We can dynamically drag time windows, zoom into specific data subsets, zoom out to see the bigger picture, and drill down on reports to extract actionable insights from wer data  
**2.Mapping Support**  
Kibana comes with powerful geospatial capabilities so that we can seamlessly layer in geographical information on top of our data and visualize results on maps  
**3.Pre-built Aggregations and Filters**  
Using Kibana’s pre-built aggregations and filters, we can easily run a variety of analytics like histograms, top-N queries, and trends with just a few clicks  
**4. Easy Distributions of Dashboards**  
We can easily share dashboards and reports with anyone in our team or outside. All they need is a browser to view and explore the data

**SELENIUM GRID**  
  
Selenium Grid is a tool that distributes the tests across multiple physical or virtual machines so that we can execute scripts in parallel (simultaneously). It dramatically accelerates the testing process across browsers and across platforms by giving us quick and accurate feedback.  
Selenium Grid allows us to execute multiple instances of WebDriver or Selenium Remote Control tests in parallel which uses the same code base, hence the code need NOT be present on the system they execute. The selenium-server-standalone package includes Hub, WebDriver, and Selenium RC to execute the scripts in grid.

**ADVANTAGES**  
1. It provides the **flexibility** to run our test automation scripts on different browsers and different OS at the same time  
2. Selenium-Grid is used to **speed up the execution** of a test pass by using multiple machines to run tests in parallel   
3. Perform **cross browser testing**  
4. **Manage the browser infrastructure** in a much flexible and efficient way when updates to browsers [or new browsers] arrives in the market  
5. When a node is free, it automatically picks up the test case waiting in the execution queue

**ELASTICSEARCH**

Elasticsearch is an open-source, broadly-distributable, readily-scalable, enterprise-grade search engine. Accessible through an extensive and elaborate API, Elasticsearch can power extremely fast searches that support wer data discovery applications. It is based on [Lucene](https://en.wikipedia.org/wiki/Lucene). It provides a distributed, [multitenant](https://en.wikipedia.org/wiki/Multitenancy)-capable full-text search engine with an [HTTP](https://en.wikipedia.org/wiki/HTTP) web interface and schema-free [JSON](https://en.wikipedia.org/wiki/JSON) documents. Elasticsearch is developed in [Java](https://en.wikipedia.org/wiki/Java_(programming_language)) and is released as [open source](https://en.wikipedia.org/wiki/Open_source_software) under the terms of the [Apache License](https://en.wikipedia.org/wiki/Apache_License).

**ADVANTAGES  
1. Built on top of Lucene**  
Elastic Search is built on top of Lucene, which is a full-featured information retrieval library, so it provides the most powerful full-text search capabilities of any open source product  
**2. Document Oriented**  
Elastic Search is document-oriented. It stores real world complex entities as structured JSON documents and indexes all fields by default, with a higher performance result  
**3.Full-Text Search**  
Elastic Search implements a lot of features, such as customized Splitting text into words, customized stemming, facetted search, and more.  
**4. Schema Free**  
Elastic search is schema free-instead, it accepts JSON documents, as well as tries to detect the data structure, index t5he data, and make it searchable.  
**5. Restful API**  
Elastic search in API driven, actions can be performed using a simple Restful API  
**6. Per-Operation Persistence**  
Elastic Search records any changes made in transaction logs on multiple nodes in cluster to minimize the chance of data loss

**LOGSTASH**  
Logstash is a light-weight, open-source, server-side data processing pipeline that allows we to collect data from a wide variety of sources, transform it on the fly, and send it to the desired destination. It is most often used as a data pipeline for [Elastic search](https://aws.amazon.com/elasticsearch-service/what-is-elasticsearch/), a popular analytics and search engine. Logstash is a popular choice for loading data into Elasticsearch because of its tight integration, powerful log processing capabilities, and over 200 pre-built open-source plugins that can help us get our data indexed the way we want it.

**ADVANTAGES**  
**1. Easily Load Unstructured Data**  
Logstash allows we to easily ingest unstructured data from a variety of data sources including system logs, website logs, and application server logs.

**2. Pre-built Filters**  
Logstash offers pre-built filters, so we can readily transform common data types, index them in Elasticsearch, and start querying without having to build custom data transformation pipelines.

**3. Flexible Plugin Architecture**  
With over 200 plugins already available on GitHub, it is likely that someone has already built the plugin we need, to customize our data pipeline. But if none is available that suits our requirements, we can easily write our own plugin

**Owasp zap**

**OWASP ZAP** (short for **Zed Attack Proxy**) is an [open-source](https://en.wikipedia.org/wiki/Open-source) [web application security scanner](https://en.wikipedia.org/wiki/Web_application_security_scanner). It is intended to be used by both those new to application security as well as professional penetration testers. It is one of the world’s most popular free security tools and is actively maintained by hundreds of international volunteers[\*](https://www.owasp.org/index.php/OWASP_Zed_Attack_Proxy_Project#Justification). It can help us automatically find security vulnerabilities in our web applications while we are developing and testing our applications. It is also a great tool for experienced pentesters to use for manual security testing.

**ADVANTAGES**  
  
1.It helps users develop and apply application security skills

2. It builds a competitive, open source, and community oriented platform

3. It provides an extensible platform for testing

4. It is designed to be easy to use

5. It raises the bar for other security tools