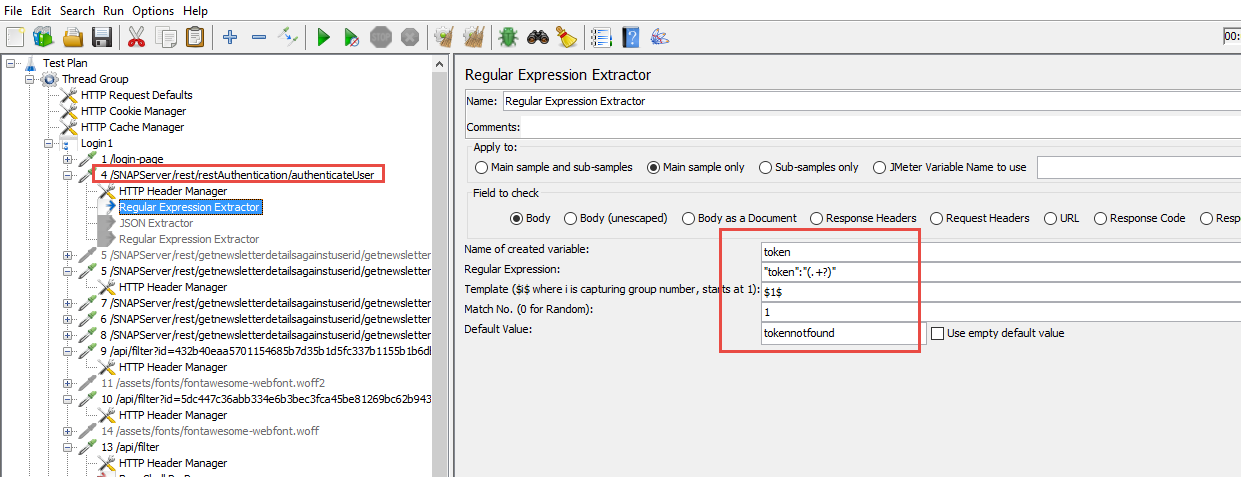
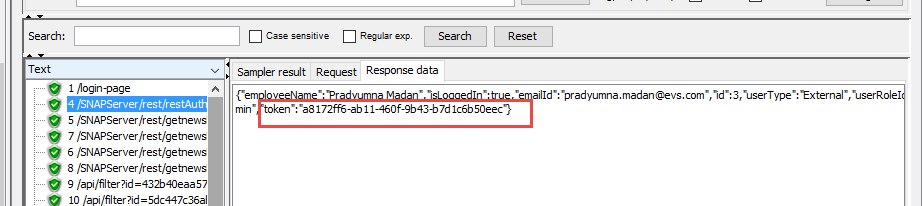
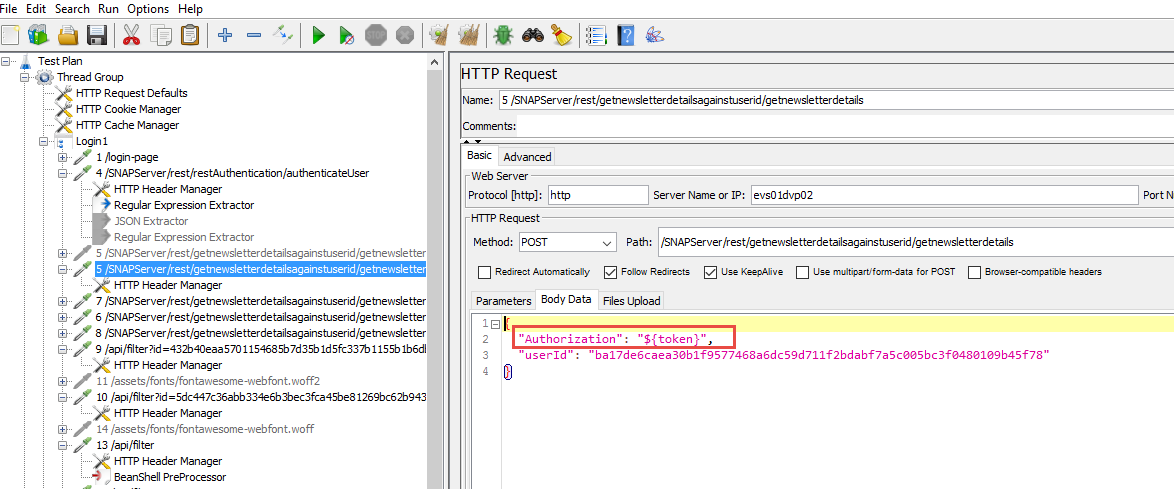
**Load testing**

When request is made to API and authorization is AES method

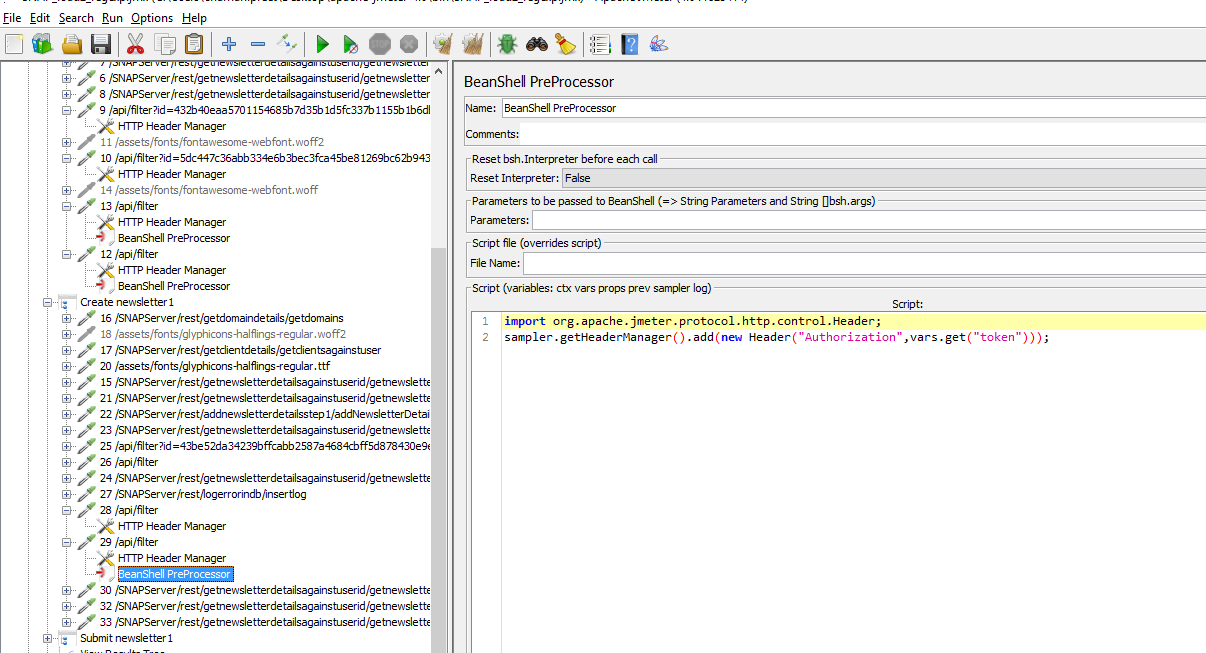
An token value is received in response of authorization which needs to be used in all subsequent requests in Authorization parameter.







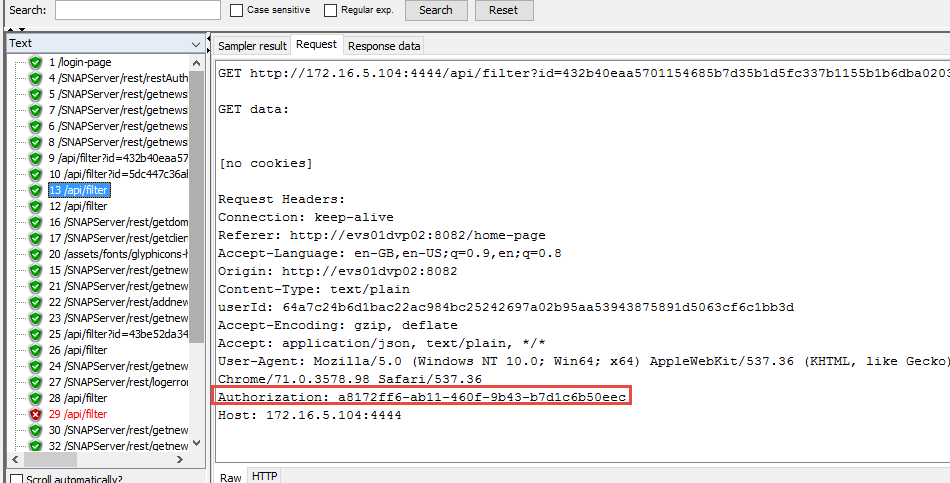
For API requests where Authorization is not given in request to pass, use **Beanshell preprocessor** to pass Authorization value in header.



Script- import org.apache.jmeter.protocol.http.control.Header;

sampler.getHeaderManager().add(new Header("Authorization","token"+vars.get("token")));

It will add a parameter and pass its value in headers



To encrypt value using base64 method-

${\_\_base64Encode(test string)}

To encrypt value using MD5 method-

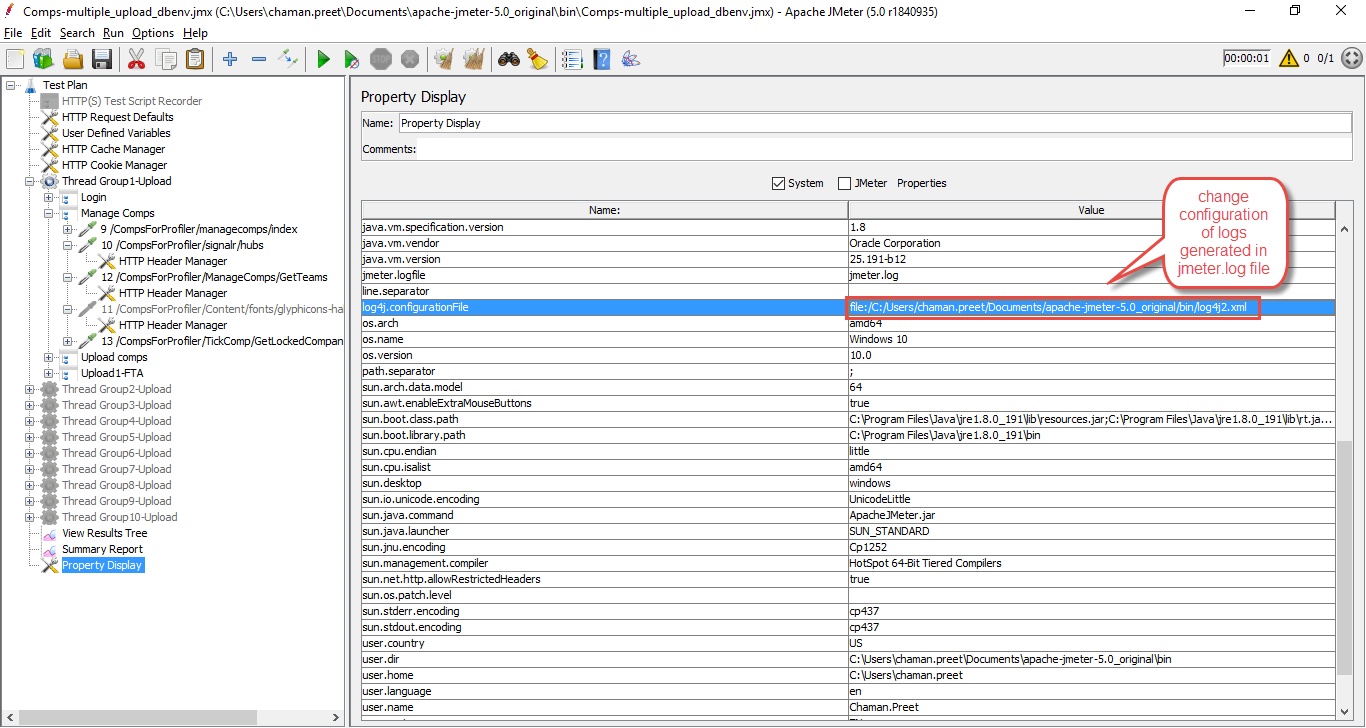
${\_\_MD5(test)}

**Important points to consider while load testing-**

https://www.loadtestingtool.com/sharepoint-testing.shtml

**Configuring logs in Jmeter**

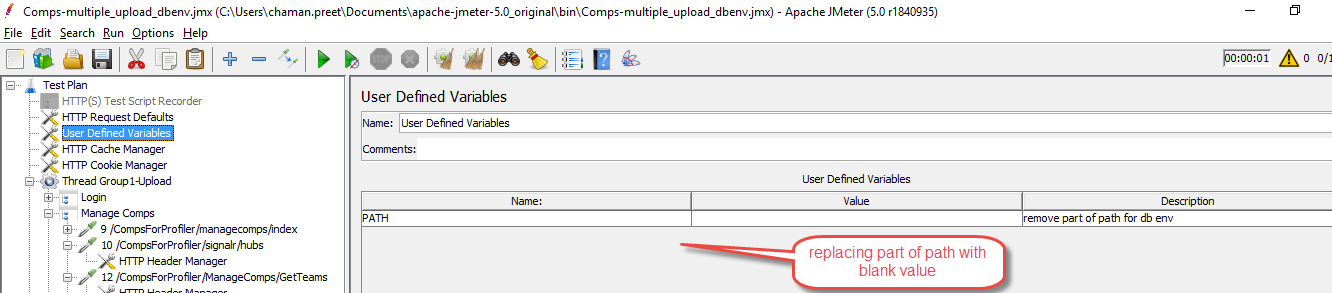
Logging is configured through an Apache **Log4j 2 configuration file**. The configuration file (log4j2.xml) is located in the bin directory of your JMeter.

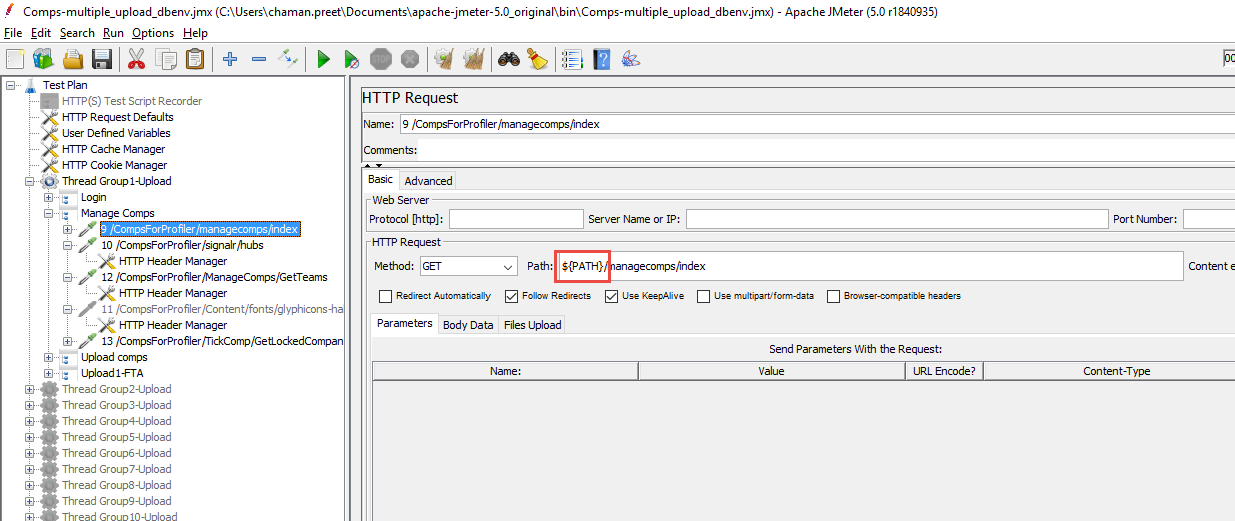


**User defined variables-**

To make changes to any parameter in whole script.

Add variable name , give its value and use it by ${VAR}





Jmeter useful features we can use-

[**https://steelkiwi.com/blog/api-testing-with-jmeter/**](https://steelkiwi.com/blog/api-testing-with-jmeter/)

**4 ways to run Jmeter behind proxy-**

<https://jmetervn.com/2017/05/08/how-to-run-jmeter-behind-the-proxy/>

**To test angular apps-**

[TestingWhiz](http://www.testing-whiz.com/?utm_campaign=Pratik&utm_medium=QnA&utm_source=Quora) is considered as one of the most user friendly test automation tools because of its code-less scripting feature. TestingWhiz offers end-to-end testing solution to test AngularJS applications.

It is a paid tool.

**Table code-**

To find particular row and column value from table-

Using Xpath "//\*[@id=\"leftcontainer\"]/table/tbody/tr[3]/td[2]

To read a table-



**Dynamic row count-**

for (int i =1;i<rows.size();i++)

{

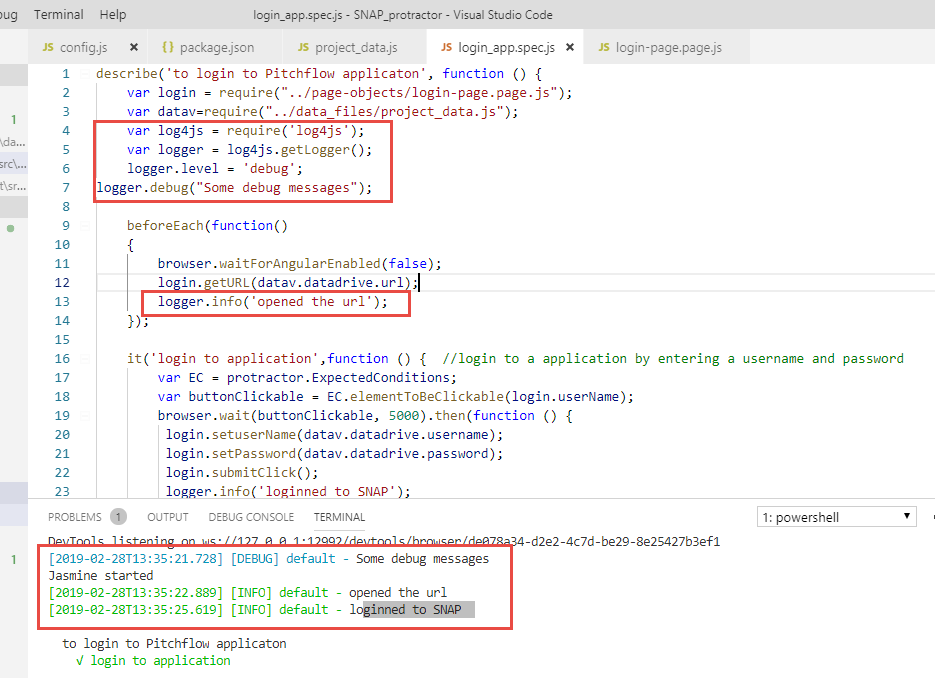
wd.findElement(By.xpath("html/body/div[1]/div[5]/table/tbody/tr[" + (i+1)+ "]/td[4]")).getText();

}

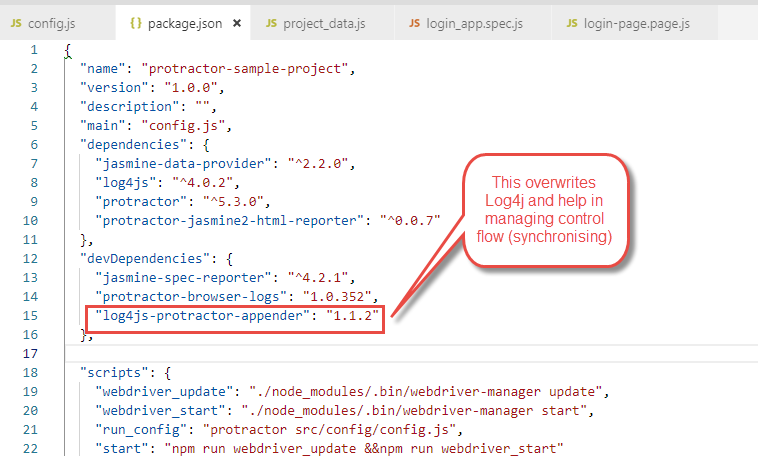
**Protractor project diagram-**

<https://coggle.it/diagram/WfBJZdhdtQABCVnH/t/protractor-project-creation>

Logs in protractor using Log4j

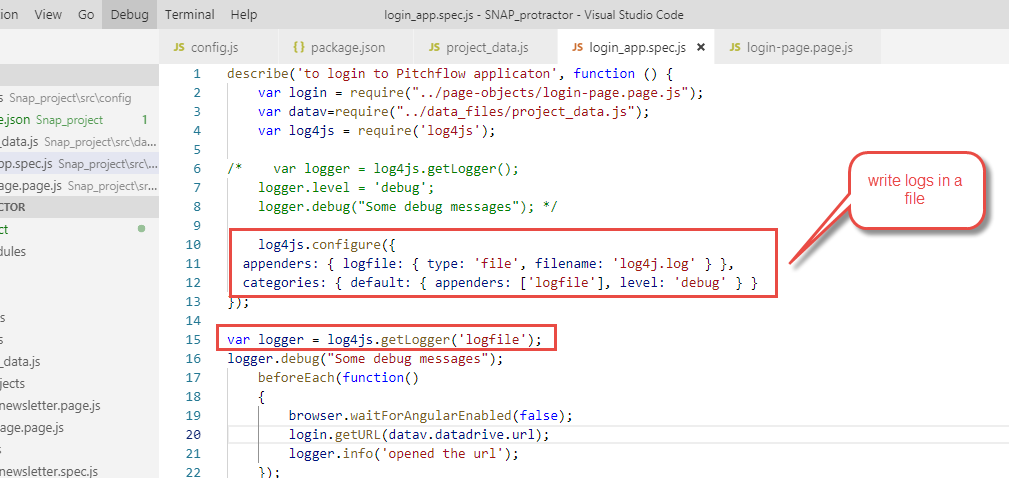


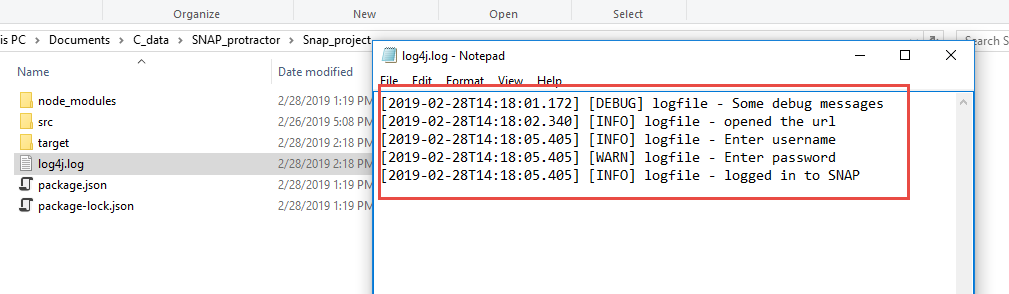
By default, log4js will not output any logs (so that it can safely be used in libraries). The level for the default category is set to OFF. To enable logs, set the level (as in the example). This will then output to stdout with the coloured layout (thanks to [masylum](http://github.com/masylum)),



Article link- <https://github.com/log4js-node/log4js-node>

**To write logs in a file-**





**Step by Step Database testing with Selenium-**

<http://blogs.quovantis.com/database-testing-using-selenium/>

**Css for handling dynamic locators-**

* Starting-text- [**[attribute^=value]**](https://www.w3schools.com/cssref/sel_attr_begin.asp)
* Ending-text-[**[attribute$=value]**](https://www.w3schools.com/cssref/sel_attr_end.asp) like [id$="downshift-main-"]
* text-in-between- [**[attribute\*=value]**](https://www.w3schools.com/cssref/sel_attr_contain.asp)
* 2 or more CSS attributes- input[type^="radio-91"][name="point-24"]
* Multiple matches- **CSS (**[**nth-of-type**](https://www.w3schools.com/cssref/sel_nth-of-type.asp)**,**[**nth-child**](https://www.w3schools.com/cssref/sel_nth-child.asp)**)**

1. [id="downshift-main"]:first-of-type // when multiple matching type & target first match
2. [id="downshift-main"]:last-of-type // when multiple matching type & target first match
3. [id="downshift-main"]:nth-of-**type**(2) // when multiple matching type & target 2nd match

**Xpath for handling dynamic locators-**

* Using single attribute- //input[@id=’userid’]
* Using multiple attribute- //input[@id=idname][@name=’username’]
* Using contain method- //input[contains(@id,’value‘)]
* Using starts-with method- //id[starts-with(@id,’ value‘)]
* Using following node- //input[@id=’ ‘]/following::input[1]
* Using preceding node- //input[@id=’ ‘]/preceding::input[1]
* Using Text method- //a[text()=’Admin’]
* Using text method with contain- //\*[contains(text(),’submit’)]
* Using OR- //\*[@type='submit' OR @name='btnReset']
* Using AND- input[@type='submit' and @name='btnLogin']

**Selenium mail function-**

**private** **static** **void** sendPDFReportByGMail(String from, String pass, String to, String subject, String body) {

Properties props = System.*getProperties*();

System.*setProperty*("https.proxyHost", "cpinternet");

System.*setProperty*("https.proxyPort", "8080");

String host = "smtp.gmail.com";

/\*props.setProperty("proxySet","true");

props.setProperty("httpProxyHost","cpinternet");

props.setProperty("httpProxyPort","8080");\*/

props.put("mail.smtp.starttls.enable", "true");

props.put("mail.smtp.host", host);

props.put("mail.smtp.user", from);

props.put("mail.smtp.password", pass);

props.put("mail.smtp.port", "587");

props.put("mail.smtp.auth", "true");

Session session = Session.*getDefaultInstance*(props);

MimeMessage message = **new** MimeMessage(session);

**try** {

//Set from address

message.setFrom(**new** InternetAddress(from));

message.addRecipient(Message.RecipientType.***TO***, **new** InternetAddress(to));

//Set subject

message.setSubject(subject);

message.setText(body);

BodyPart objMessageBodyPart = **new** MimeBodyPart();

objMessageBodyPart.setText("Please Find The Attached Report File!");

Multipart multipart = **new** MimeMultipart();

multipart.addBodyPart(objMessageBodyPart);

objMessageBodyPart = **new** MimeBodyPart();

//Set path to the pdf report file

String filename = System.*getProperty*("user.dir")+"\\default test.pdf";

//Create data source to attach the file in mail

DataSource source = **new** FileDataSource(filename);

objMessageBodyPart.setDataHandler(**new** DataHandler(source));

objMessageBodyPart.setFileName(filename);

multipart.addBodyPart(objMessageBodyPart);

message.setContent(multipart);

Transport transport = session.getTransport("smtp");

transport.connect(host, from, pass);

transport.sendMessage(message, message.getAllRecipients());

transport.close();

}

**catch** (AddressException ae) {

ae.printStackTrace();

}

**catch** (MessagingException me) {

me.printStackTrace();

}

}

***Testing terminology-***

**Defect Clustering:** When a small number of modules contains most of the bugs detected or show the most operational failures. **Pesticide Paradox**: If the same **tests**are repeated over and over again, eventually the same **test**cases will no longer find new bugs.

**Commercial off-the-shelf (COTS)** - describes software or hardware products that **are** ready-made and available for sale to the general public. For example, Microsoft Office **is a COTS** product that **is a** packaged software solution for businesses. **Commercial off-the-shelf** or **commercially available off-the-shelf**[[1]](https://en.wikipedia.org/wiki/Commercial_off-the-shelf#cite_note-1) (**COTS**) products are packaged solutions which are then adapted to satisfy the needs of the purchasing organization, rather than the commissioning of custom-made

**Defect Leakage** is the Metric which is used to identify the efficiency of the QA testing i.e., how many **defects** are missed/slipped during the QA testing.

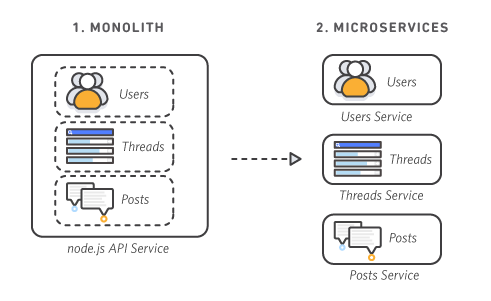
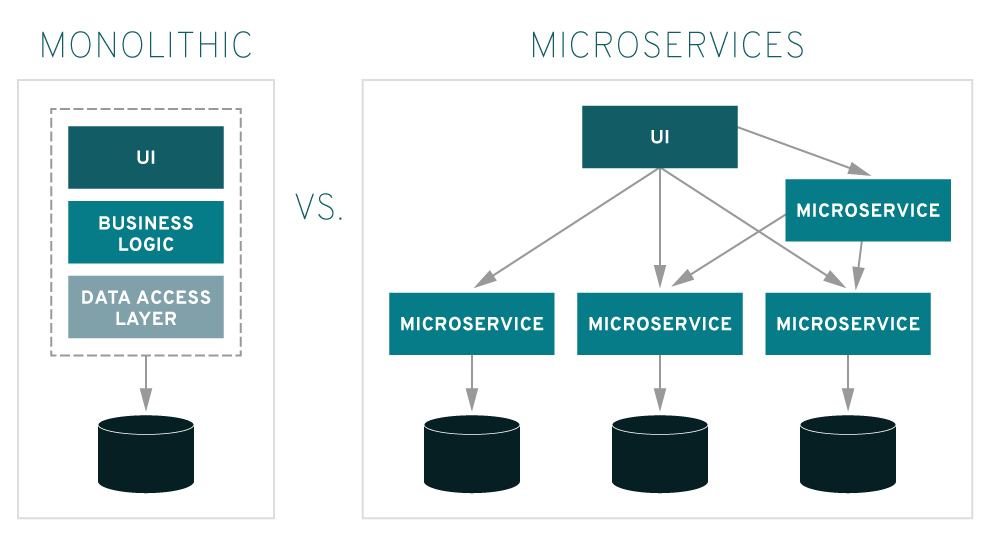
**Defect Leakage** = (No. of **Defects** found in UAT / No. of **Defects** found in QA testing (before UAT).)\*100

**Microservices-**  Microservices are small independently deployable web-services that encapsulate functionality in a scalable architecture. The idea is to split your application into a set of smaller, interconnected services instead of building a single monolithic application. Each microservice is a small application that has its own hexagonal architecture consisting of business logic along with various adapters

**As per timeline---**

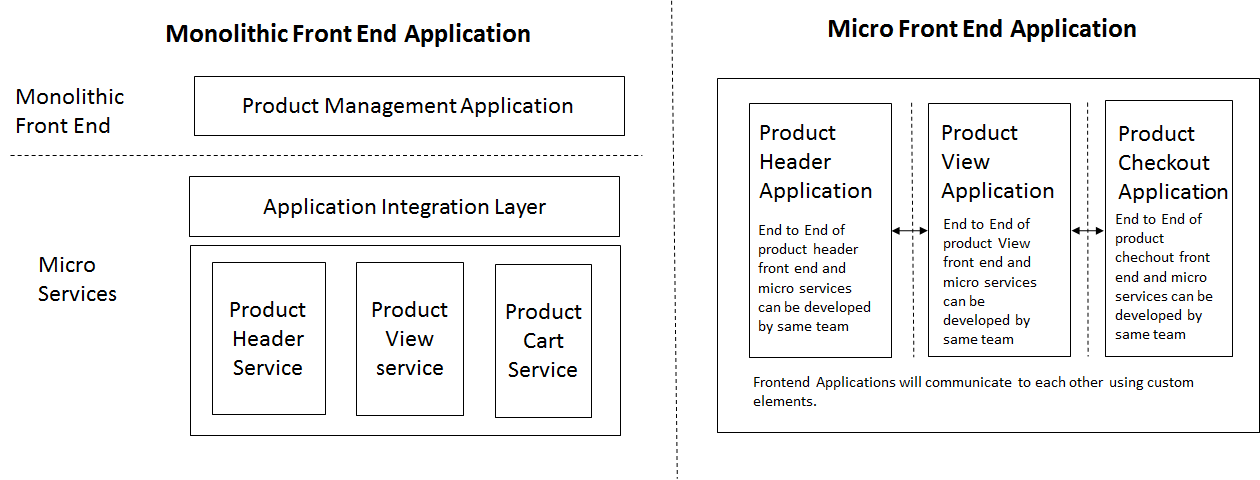
**1. Monolithic architecture 2. Microservices (for backend) 3. Microfrontend**

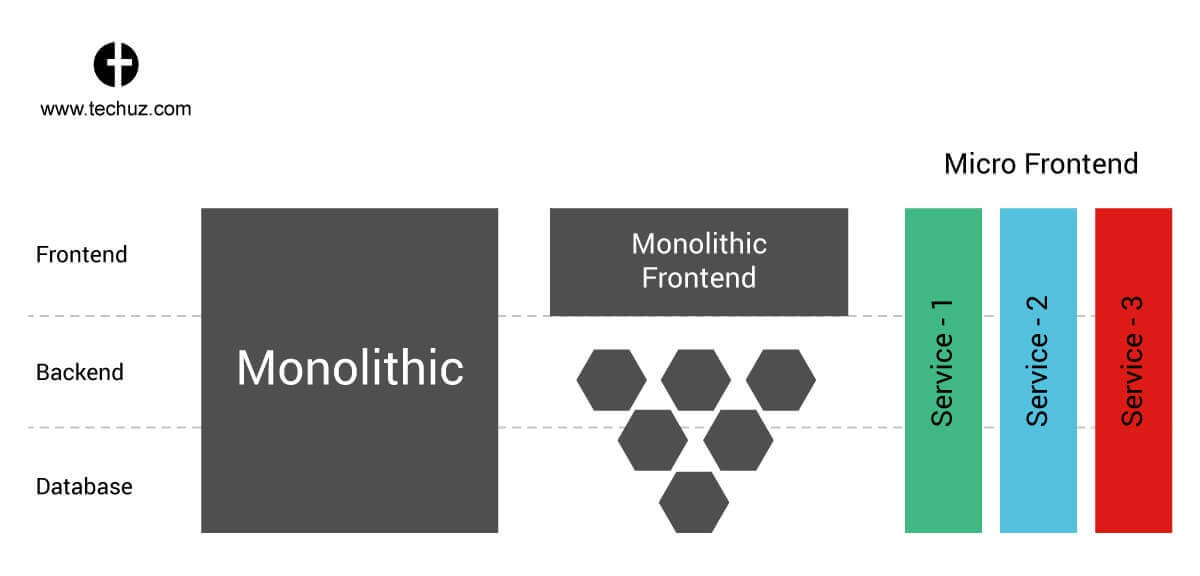
**(microservices for frontend)**



**Microfrontend-** Micro Frontend is a Microservice approach to front-end web development (i.e UI part). Microservice was a development technique but only for backend.

support heterogenous technologies, replacement of microservices is easy, mix and match as per requirement is possible.





**Java-**

* Can we override variable? --No

We can overwrite it (Giving new value to variable) but cannot override it (Giving new definition to the function).

Overriding is for instance methods.

# Can we override static method? why?

We cannot override static methods. Static methods belong to class, not belongs to object.

When we inherit static method in subclass, superclass static method is just hidden, not overridden.

Static methods in Java are inherited, but can not be overridden.

Example:

public class Writer {

public static void write() {

System.out.println("Writing");

}

}

public class Author extends Writer {

public static void write() {

System.out.println("Writing book");

}

}

public class Programmer extends Writer {

public static void write() {

System.out.println("Writing code");

}

public static void main(String[] args) {

Writer w = new Programmer();

w.write();

Writer secondWriter = new Author();

secondWriter.write();

Writer thirdWriter = null;

thirdWriter.write();

Author firstAuthor = new Author();

firstAuthor.write();

}

}

You'll get the following:

Writing

Writing

Writing

Writing book

**PUT and POST** methods in REST services.

POST is often used to create a new entity, and PUT is often used to update an existing entity.

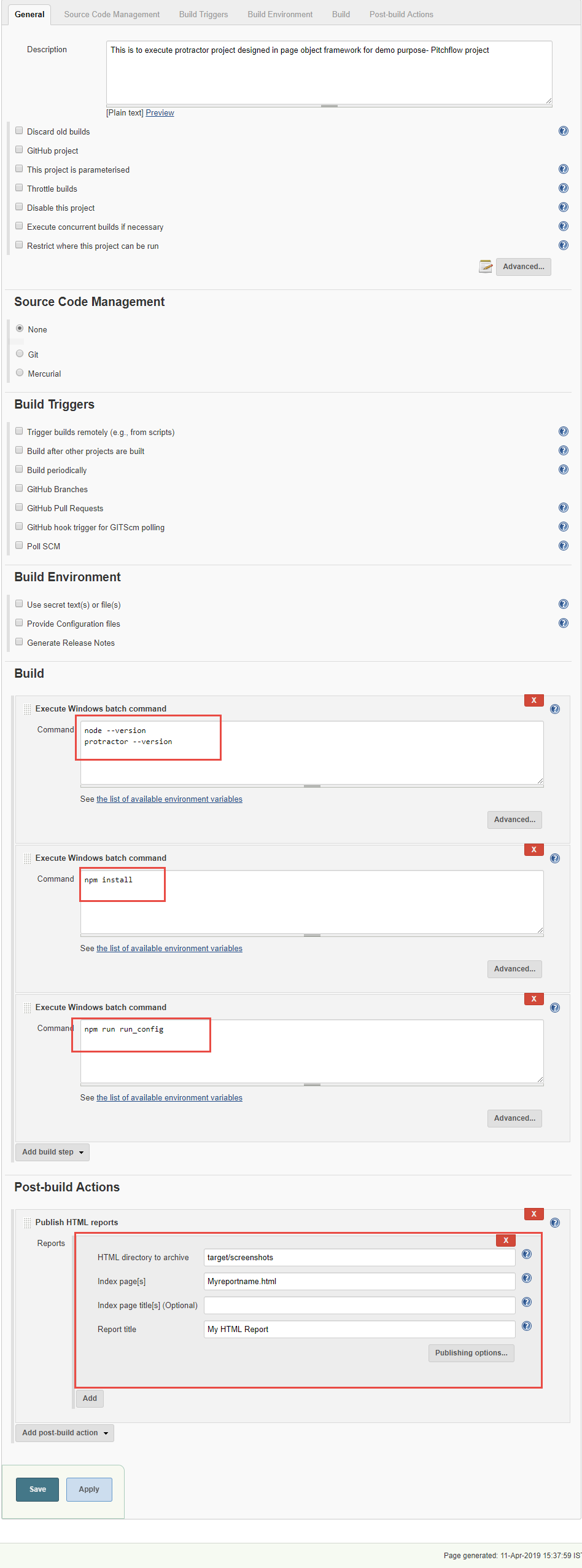
Similar to the [GET request](http://www.java67.com/2014/08/difference-between-post-and-get-request.html), PUT request is also idempotent in HTTP, which means it will produce the same results if executed once more multiple times.  
  
Unlike PSOT method, which will execute request multiple times.

**Github-**

Use git mergetool to run one of several merge utilities to resolve merge conflicts.  It is typically run after git merge.

**Jenkins-**

* Protractor Freestyle project with batch commands and html report



* Protractor pipeline project from Git

