SOFTWARE TESTING

CLIENT SIDE WEB APPLICATION TESTING (BYPASS TESTING)

Submitted by:

Abhishek Garg (MT2020021)

Nitesh Jain (MT2020118)

Mohd Asad Ansari (MT2020147)

Link to the code :
https://drive.google.com/file/d/120W_NUpwilmdIebyn0yXN9MjheUDelam/view?usp=sharing

Introduction

Web software applications are increasingly being used in critical situations. Web applications are used to transfer, receive and store personal, corporate and sensitive data. Input validation testing (IVT) tests user inputs to ensure compliance with system requirements, most importantly in user-dependent software inputs, including web applications. A common approach to web applications is to perform input on the client for scripting languages such as JavaScript. The hidden problem of verifying client side input is that end users can skip this verification. Verifying authentication may result in software failure, and may violate the security of Web applications, resulting in unauthorized access to data, system failures, unauthorized purchases and installation of fake data. We developed a strategy called bypass testing to create client side testing for web applications that deliberately violates explicit and implicit testing of user input. This paper describes the strategy, explains the specific rules and conditions for testing, describes the default concept of conceptual evidence, and presents the initial test results from using bypass testing.

About Webapp

A marketplace auction platform is exactly what it sounds like: auctioneers list their catalogues in an online auction marketplace right alongside their competitors. The competitors interested in the product, bid for the product with the amount they are willing to pay for that product. The owner gets to know the bids placed on the product and chooses the one who bids with the maximum amount. The online bidding platform enables the product to be sold worldwide without being physically present at the local. The current existing structure doesn't provide an efficient platform for this bidding platform .So we are building a website which will provide a platform to share complete details which will be useful to others to bid for products in live scenario.We are also using DevOps tools to build a high quality website in less time.Our online auction platform i.e. Bidding Application lets your customers sell their product by auction, an interested person will bid on the available products, and the winner will get the product. It has been deployed as iaas or developed further to meet your specific business requirements. The architecture of our project demands three layers.

The front end of the project is handled by “React” Framework.The middle layer is built on “SpringBoot” Framework and communicates with mysql database.The back end is swiftly handled by “MySQL”, using “Azure database” for interaction between back end and middle layer.

1. The UI of the application enables the user to register itself on the platform to either sell or buy products of its wish. Once registered, the user is capable of putting ads for the product that he wishes to sell. The product can be anything such as antiques, used electronics, paintings etc.
2. For putting an ad, the parameters required are product name, description of the product and minimum amount for bid. Once the ad has been posted it displays on the products list page with all other products. The user can select the product of his wish from the list of products in front of him.
3. By clicking on the product, the user is able to get all the details about the product as well as the owner of the product. By clicking on the bid button, the user enters the page where he can fill in the amount he is willing to pay. The amount should be obviously greater than the minimum bid amount fixed by the owner else the bid won’t be set showing an alert regarding the same.
4. Once the bid has been placed, the bid is now visible in the product details dashboard. The user has the option to select for the bid he wishes, it's obvious that the user opts for the bid with a maximum bid amount. Once selected, the product becomes inactive and thus no further bids can be placed on the product.

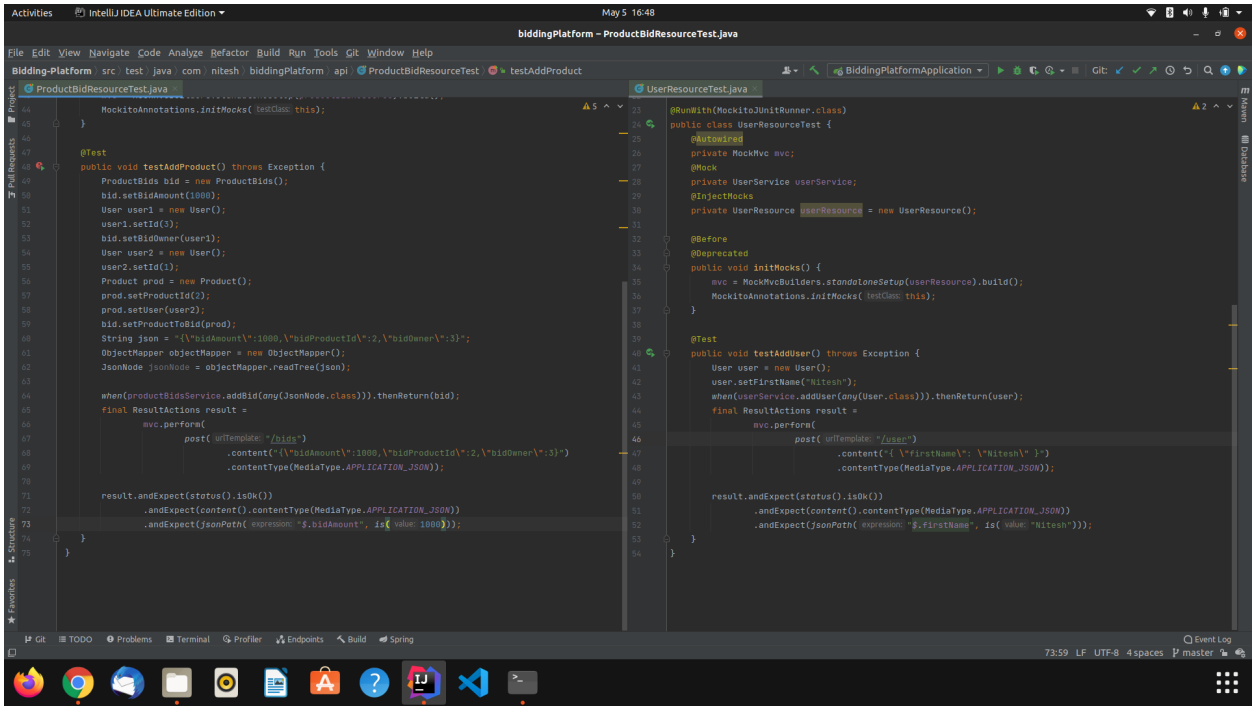
Initial Unit Testing

A unit test is a piece of code written by a developer that executes a specific functionality in the code to be tested and asserts a certain behavior or state. The percentage of code which is tested by unit tests is typically called test coverage. A unit test targets a small unit of code, e.g., a method or a class. External dependencies should be removed from unit tests, e.g., by replacing the dependency with a test implementation or a (mock) object created by a test framework. Unit tests are not suitable for testing complex user interfaces or component interaction. For this, you should develop integration tests.

Junit : It is an open-source framework, which is used for writing test cases and running test cases. Provides annotations to identify test methods, assertions for testing expected results, etc. It is useful for java developers to write and run repeatable tests. Erich Gamma and Kent Beck initially developed it. It is an instance of xUnit architecture. As the name implies, it is used for unit testing of a small chunk of code.Developers who are following test-driven methodology must write and execute unit tests first before any code.

Unit testing of code

Mockito : It is a mocking framework that tastes really good. It lets you write beautiful tests with a clean & simple API. Mockito doesn't give you a hangover because the tests are very readable and they produce clean verification errors.The framework allows the creation of test double objects in automated unit tests for the purpose of test-driven development or behavior-driven development. The framework's name and logo are a play on mojitos, a type of drink



Code for test cases

Bypass Testing

Verification of multiple inputs focuses on individual parameters. This works well for traditional software, where patterns of interaction between users and software are modified and cannot be modified by users. What is interesting is that the use of dynamic Web pages means that the same URL can produce different forms at different times, depending on the parameters provided, status on server, client features, and more environmental knowledge. In addition, Web users applications can not only change the value of input parameters, but can also change the number of input parameters and control flow. This makes it easier to break the boundaries between the different parameters and between the components of the software. This section describes a systematic way of identifying barriers between inputs boundaries. The rules for conducting bypass testing are then provided conditions for testing a Web application to verify these issues are adequately evaluated.

Input validations to bypass

1• Data type and value modification. HTML inputs are initially strings, but they are often converted to other data types on the server. Data type conversion testing uses values of different types to evaluate the server-side processing, including general strings, integers, real numbers, and dates.

2• HTML built-in length violation. The HTML tag input has an attribute maxlength. Invalid values are generated to violate these restrictions.

3• HTML built-in value violation. Pre-defined input restrictions from HTML select, check and radio boxes are violated by modifying the submission to submit values that are not in the pre-defined set.

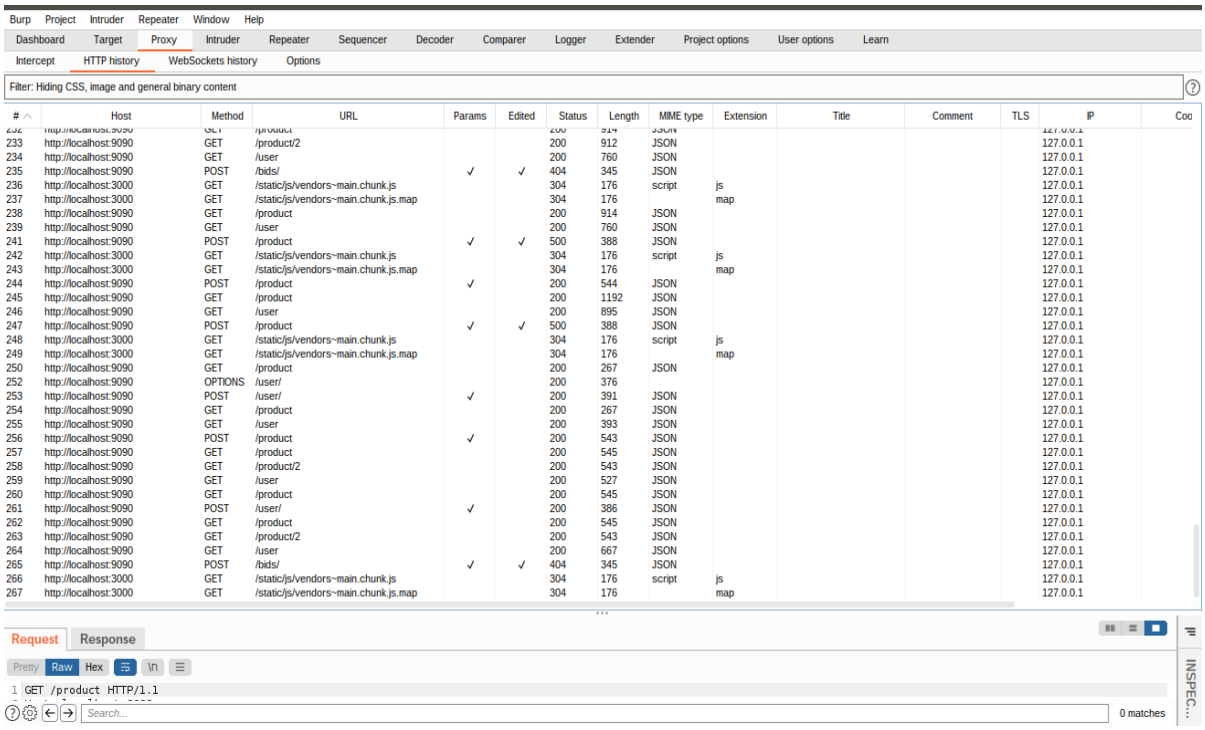
4• HTML constraint violation. Pre-defined input constraints HTML tags, front end scripts and other restrictions added by the developer for deployment of application.

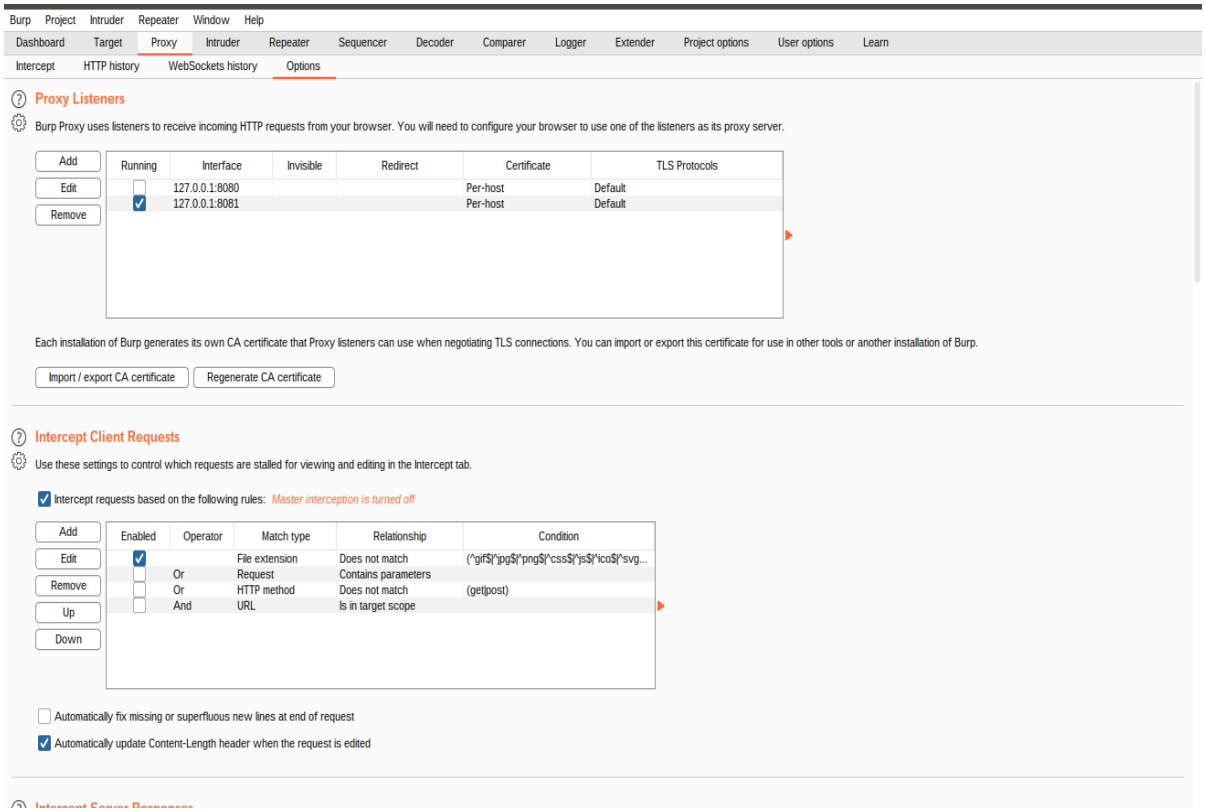
Tool Used : Burp Suite

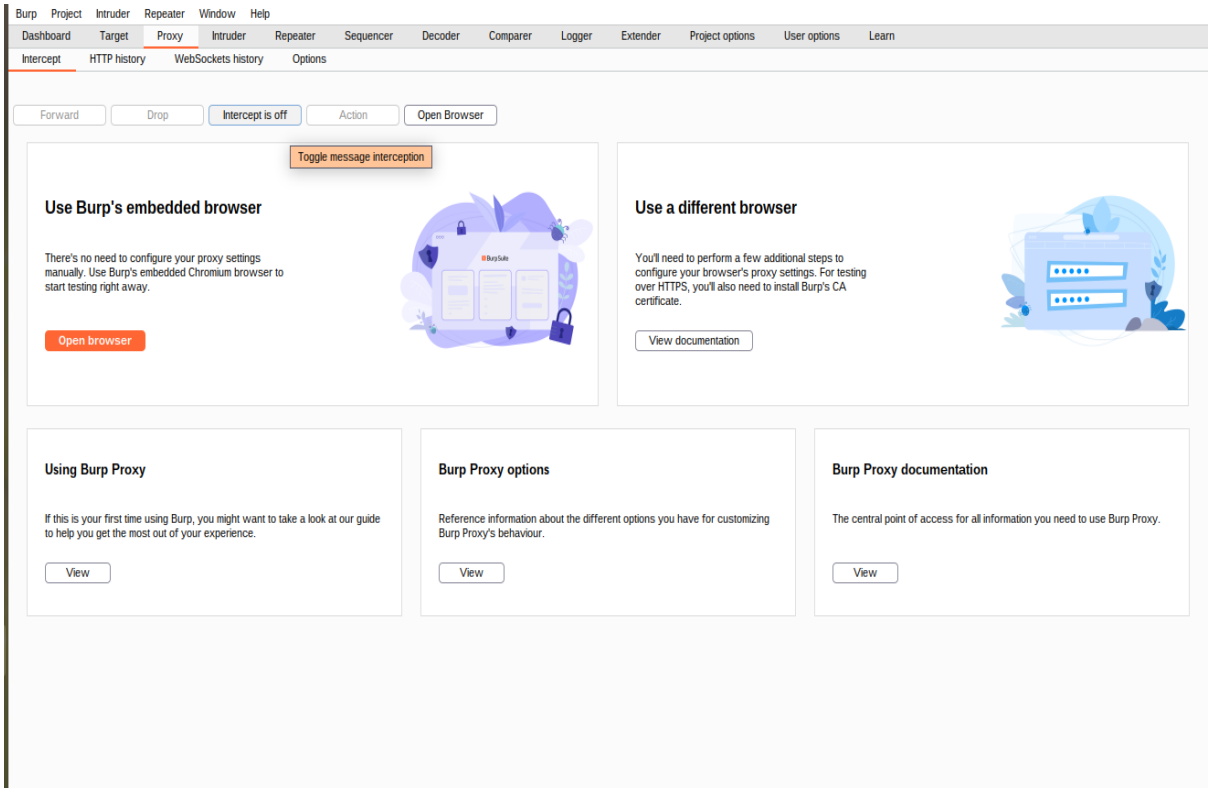
Burp or Burp Suite is a set of tools mostly used to do penetration testing of web applications. Developed by a company called Portswigger, also named after its founder Dafydd Stuttard. BurpSuite aims to have it all in one set of tools and its capabilities can be enhanced by adding add-ons called BApps. It is a very popular tool among web application security researchers and bug bounty hunters. Its ease of use makes it even more special than other free ones like OWASP ZAP.

For installation or configuration visit here for linux.

Working : With the help of this tool we can verify the path of a packet through the layers of the network to its destination and modify the request sent by the client and by altering the packet information and sending the updated packet to the server. It performs network interface level packet tracing and issues output on the packet trace.



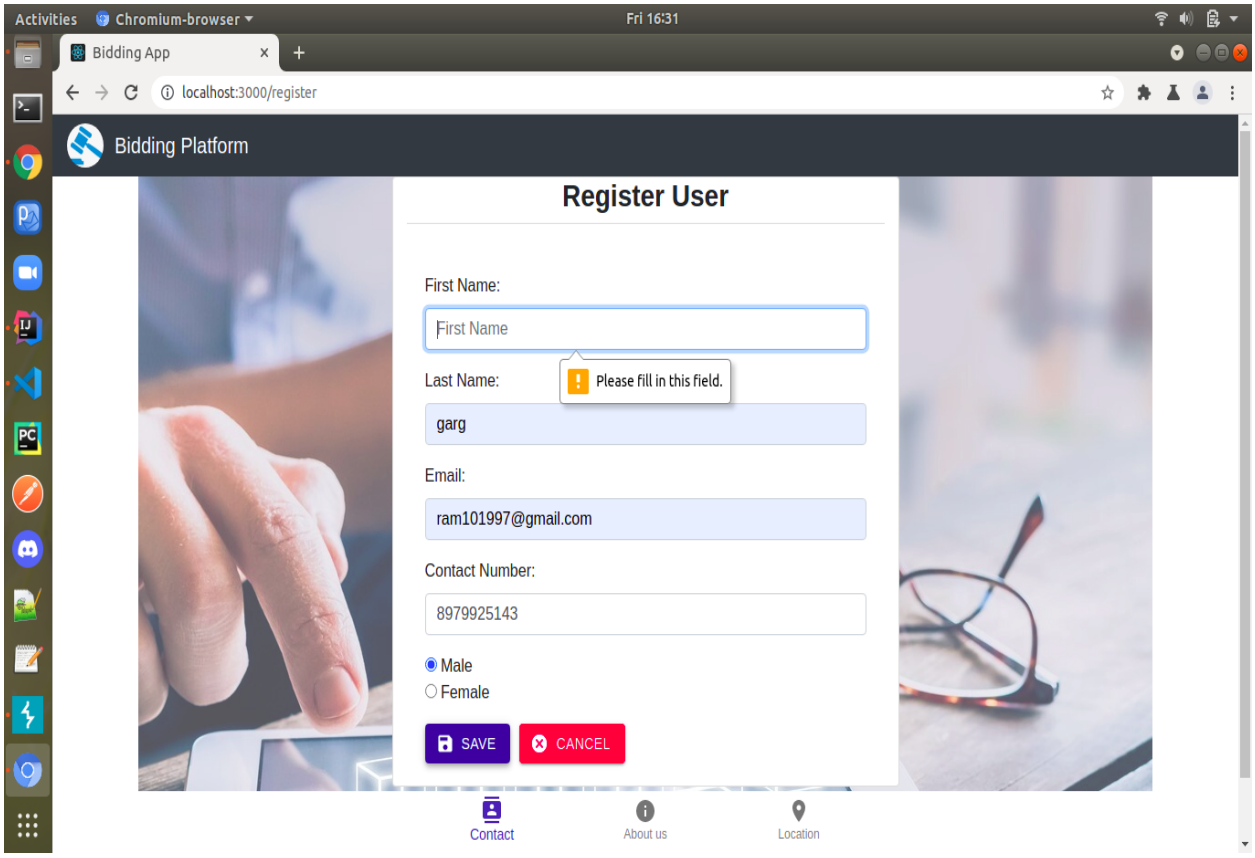

Burp Project Intruder Repeater Window Help
Dashboard Target Proxy Intruder Repeater Sequencer Decoder Comparer Logger Extender Project options User options Learn
Intercept HTTP history WebSockets history Options
Proxy Listeners
Burp Proxy uses listeners to receive incoming HTTP requests from your browser. You will need to configure your browser to use one of the listeners as its proxy server.
Add
Edit
Remove
Running Interface Invisible Redirect Certificate TLS Protocols
127.0.0.1:8080 Per-host Default
127.0.0.1:8081 Per-host Default
Each installation of Burp generates its own CA certificate that Proxy listeners can use when negotiating TLS connections. You can import or export this certificate for use in other tools or another installation of Burp.
Import / export CA certificate
Regenerate CA certificate
Intercept Client Requests
Use these settings to control which requests are stalled for viewing and editing in the Intercept tab.
Intercept requests based on the following rules: Master interception is turned off
Add
Edit
Remove
Up
Down
Enabled Operator Match type Relationship Condition
File extension Does not match (^gif$|^jpg$|^png$|^css$|^js$|^ico$|^svg...
Or Request Contains parameters
Or HTTP method Does not match (get|post)
And URL Is in target scope
Automatically fix missing or superfluous new lines at end of request
Automatically update Content-Length header when the request is edited

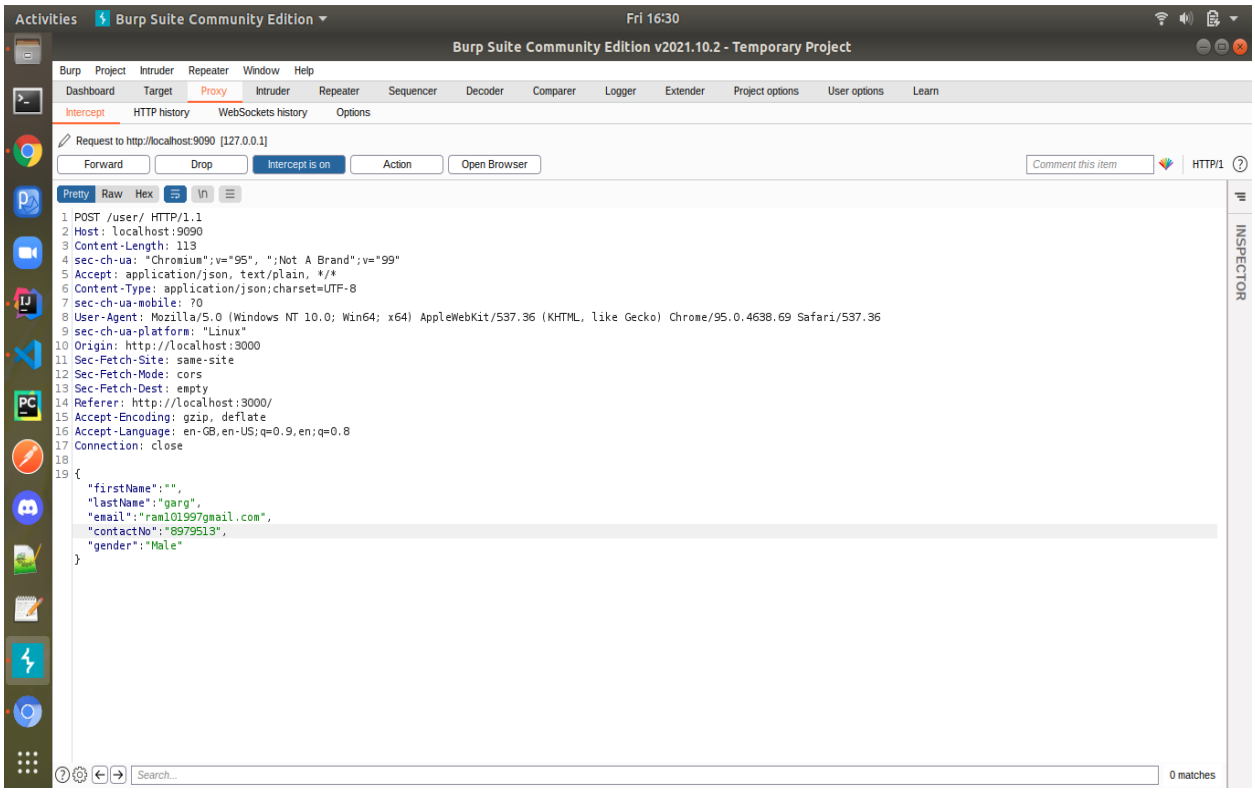

Burp Project Intruder Repeater Window Help
Dashboard Target Proxy Intruder Repeater Sequencer Decoder Comparer Logger Extender Project options User options Learn
Intercept HTTP history WebSockets history Options
Forward Drop Intercept is off Action Open Browser
Toggle message interception
Use Burp's embedded browser
There's no need to configure your proxy settings manually. Use Burp's embedded Chromium browser to start testing right away.
Open browser
Use a different browser
You'll need to perform a few additional steps to configure your browser's proxy settings. For testing over HTTPS, you'll also need to install Burp's CA certificate.
View documentation
Using Burp Proxy
If this is your first time using Burp, you might want to take a look at our guide to help you get the most out of your experience.
View
Burp Proxy options
Reference information about the different options you have for customizing Burp Proxy's behaviour.
View
Burp Proxy documentation
The central point of access for all information you need to use Burp Proxy.
View

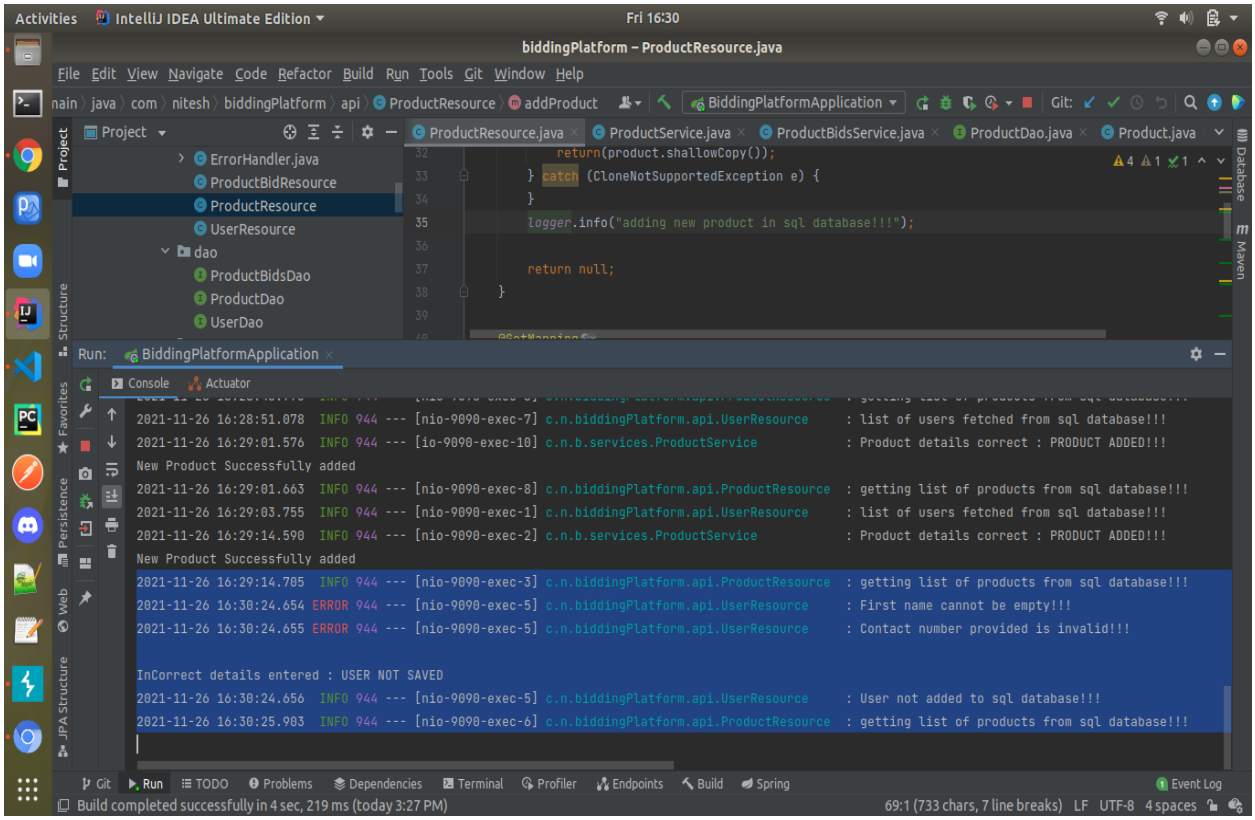
Test Cases:

Test 1: In the first test case we are trying to register a new user and following all the client side validations provided by the developer. We enter all the details correctly. Then using the Burp tool we intercept the request and alter the packet by deleting and modifying values. This alteration is caught by server side validations and does not allow incorrect data to pass through the server side thereby confirming the correctness of the application.

As we are trying to pass an empty first name,it is not allowed on the client side but a request is sent by the tool to the server side which is handled by the developer using server side checking.

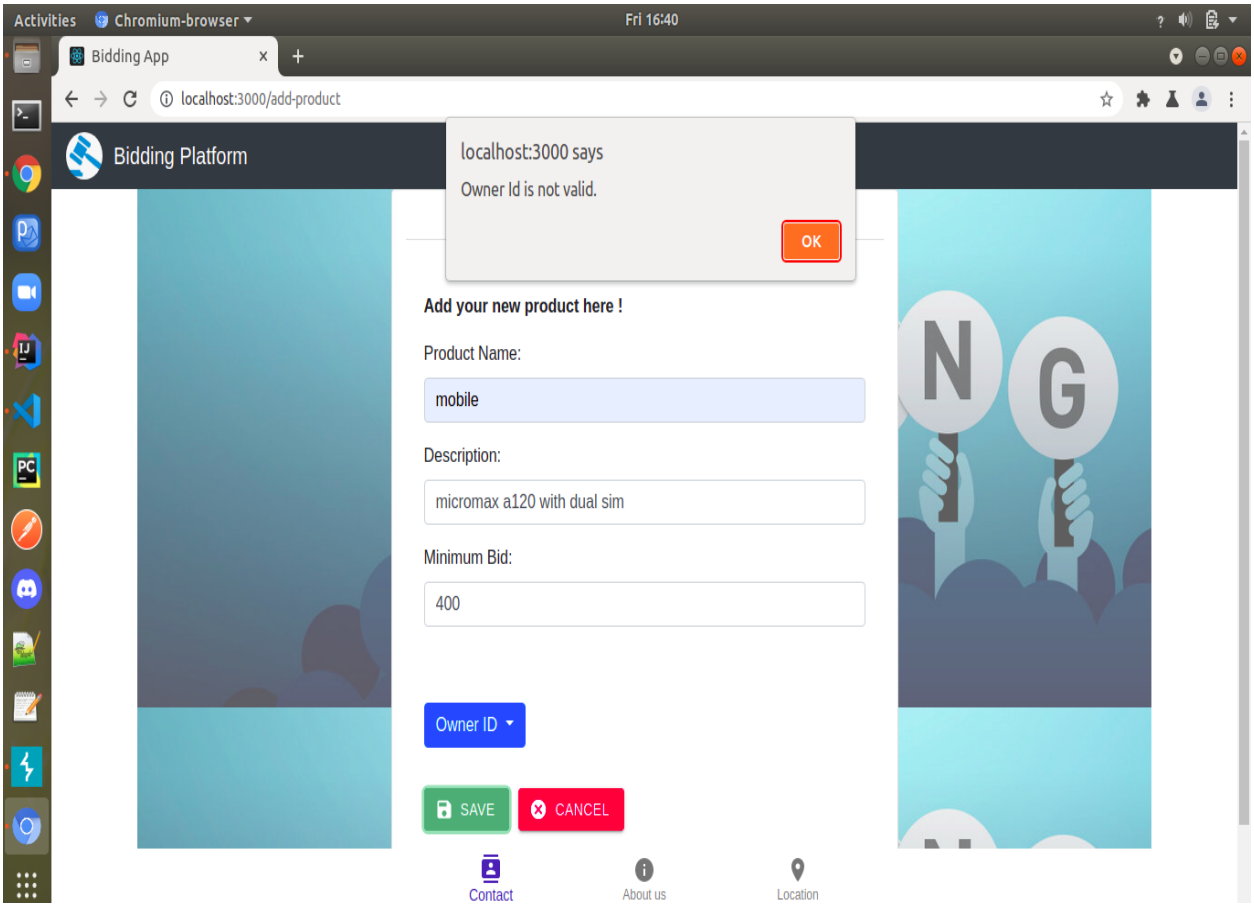


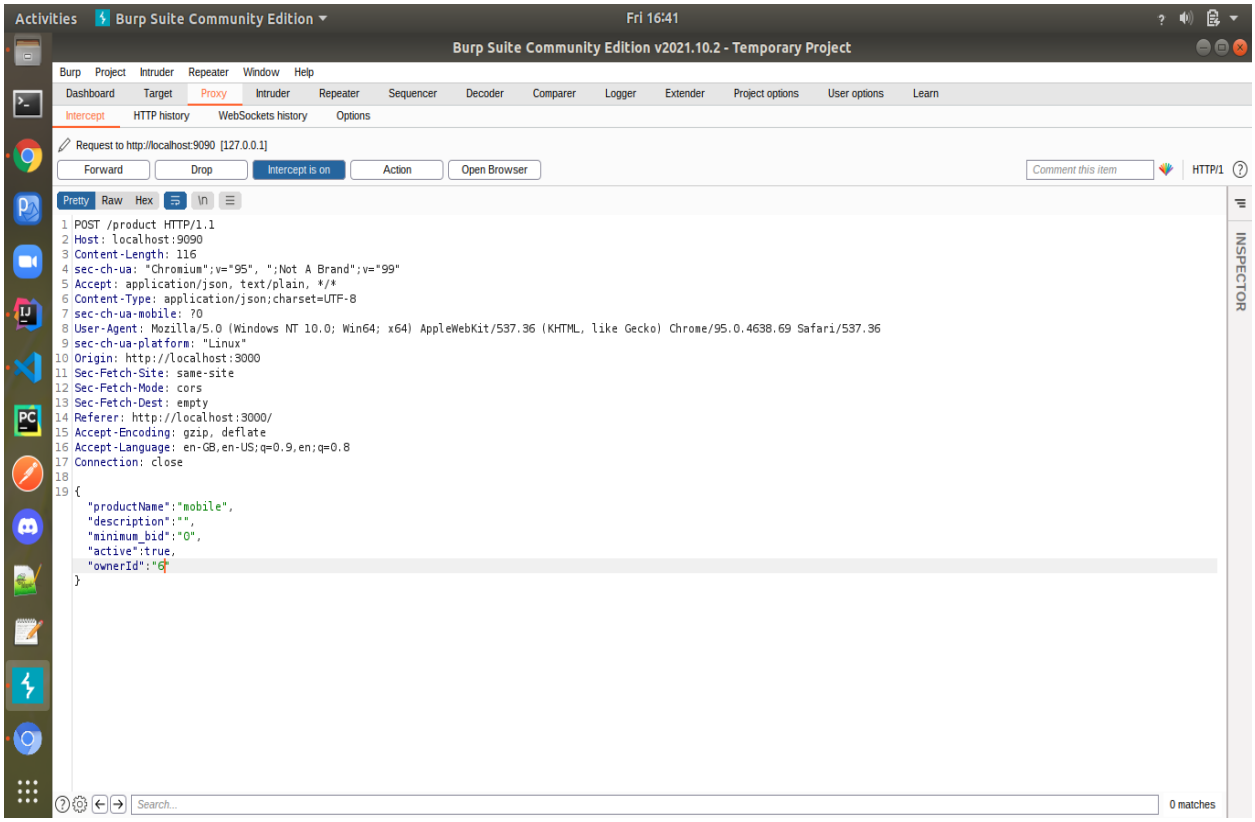

Activities
Burp Suite Community Edition
Fri 16:30
Burp Suite Community Edition v2021.10.2 - Temporary Project
Burp Project Intruder Repeater Window Help
Dashboard Target Proxy Intruder Repeater Sequencer Decoder Comparer Logger Extender Project options User options Learn
Intercept HTTP history WebSockets history Options
Request to http://localhost:9090 [127.0.0.1]
Forward Drop Intercept is on Action Open Browser
Comment this item
HTTP/1
Pretty Raw Hex
1 POST /user/ HTTP/1.1
2 Host: localhost:9090
3 Content-Length: 113
4 sec-ch-ua: "Chromium";v="95", ";Not A Brand";v="99"
5 Accept: application/json, text/plain, */*
6 Content-Type: application/json;charset=UTF-8
7 sec-ch-ua-mobile: ?0
8 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/95.0.4638.69 Safari/537.36
9 sec-ch-ua-platform: "Linux"
10 Origin: http://localhost:3000
11 Sec-Fetch-Site: same-site
12 Sec-Fetch-Mode: cors
13 Sec-Fetch-Dest: empty
14 Referer: http://localhost:3000/
15 Accept-Encoding: gzip, deflate
16 Accept-Language: en-GB,en-US;q=0.9,en;q=0.8
17 Connection: close
18
19 {
"firstName":"",
"lastName":"garg",
"email":"raml01997gmail.com",
"contactNo":"8979513",
"gender":"Male"
}
INSPECTOR
Search...
0 matches

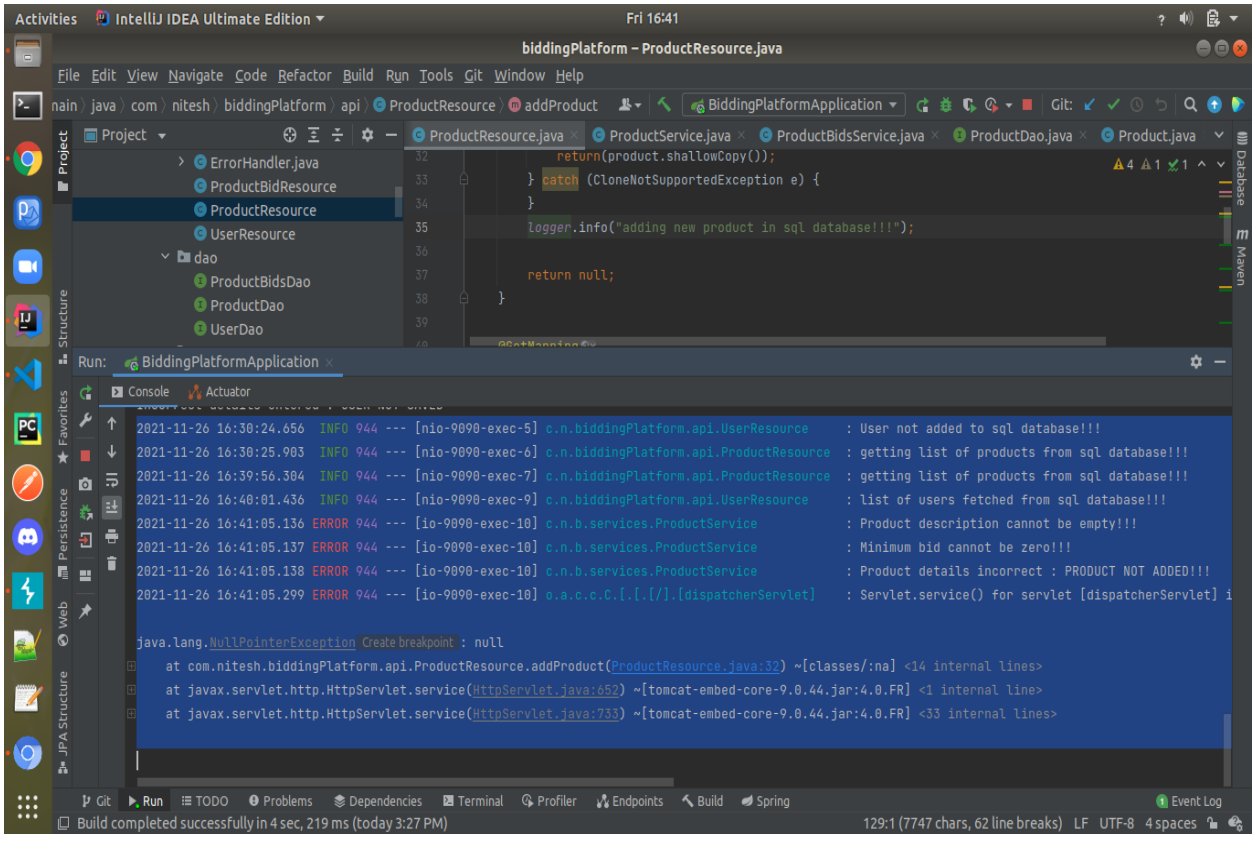

Activities
IntelliJ IDEA Ultimate Edition
Fri 16:30
biddingPlatform – ProductResource.java
File Edit View Navigate Code Refactor Build Run Tools Git Window Help
nain java com nitesh biddingPlatform api ProductResource addProduct
BiddingPlatformApplication
Git:
Project
ProductResource.java
ProductService.java
ProductBidsService.java
ProductDao.java
Product.java
ErrorHandler.java
ProductBidResource
ProductResource
UserResource
dao
ProductBidsDao
ProductDao
UserDao
return(product.shallowCopy());
} catch (CloneNotSupportedException e) {
}
logger.info("adding new product in sql database!!!");
return null;
}
Run: BiddingPlatformApplication
Console
Actuator
2021-11-26 16:28:51.078 INFO 944 --- [nio-9090-exec-7] c.n.biddingPlatform.api.UserResource : list of users fetched from sql database!!!
2021-11-26 16:29:01.576 INFO 944 --- [io-9090-exec-10] c.n.b.services.ProductService : Product details correct : PRODUCT ADDED!!!
New Product Successfully added
2021-11-26 16:29:01.663 INFO 944 --- [nio-9090-exec-8] c.n.biddingPlatform.api.ProductResource : getting list of products from sql database!!!
2021-11-26 16:29:03.755 INFO 944 --- [nio-9090-exec-1] c.n.biddingPlatform.api.UserResource : list of users fetched from sql database!!!
2021-11-26 16:29:14.590 INFO 944 --- [nio-9090-exec-2] c.n.b.services.ProductService : Product details correct : PRODUCT ADDED!!!
New Product Successfully added
2021-11-26 16:29:14.705 INFO 944 --- [nio-9090-exec-3] c.n.biddingPlatform.api.ProductResource : getting list of products from sql database!!!
2021-11-26 16:30:24.654 ERROR 944 --- [nio-9090-exec-5] c.n.biddingPlatform.api.UserResource : First name cannot be empty!!!
2021-11-26 16:30:24.655 ERROR 944 --- [nio-9090-exec-5] c.n.biddingPlatform.api.UserResource : Contact number provided is invalid!!!
InCorrect details entered : USER NOT SAVED
2021-11-26 16:30:24.656 INFO 944 --- [nio-9090-exec-5] c.n.biddingPlatform.api.UserResource : User not added to sql database!!!
2021-11-26 16:30:25.903 INFO 944 --- [nio-9090-exec-6] c.n.biddingPlatform.api.ProductResource : getting list of products from sql database!!!
Git Run TODO Problems Dependencies Terminal Profiler Endpoints Build Spring
Event Log
Build completed successfully in 4 sec, 219 ms (today 3:27 PM)
69:1 (733 chars, 7 line breaks) LF UTF-8 4 spaces
Database
Maven
Structure
Favorites
Persistence
Web
JPA Structure

Test 2 : In the second test case, we are trying to add a new product that needs to be bid by entering product name, description and bidding amount but don't provide owner id which is checked and validated by client side.

Then using the Burp tool we intercept the request and alter the packet. This alteration is caught by server side validations which in turn throws an error of product description being empty and the product bidding amount having a value zero that was bypassed.

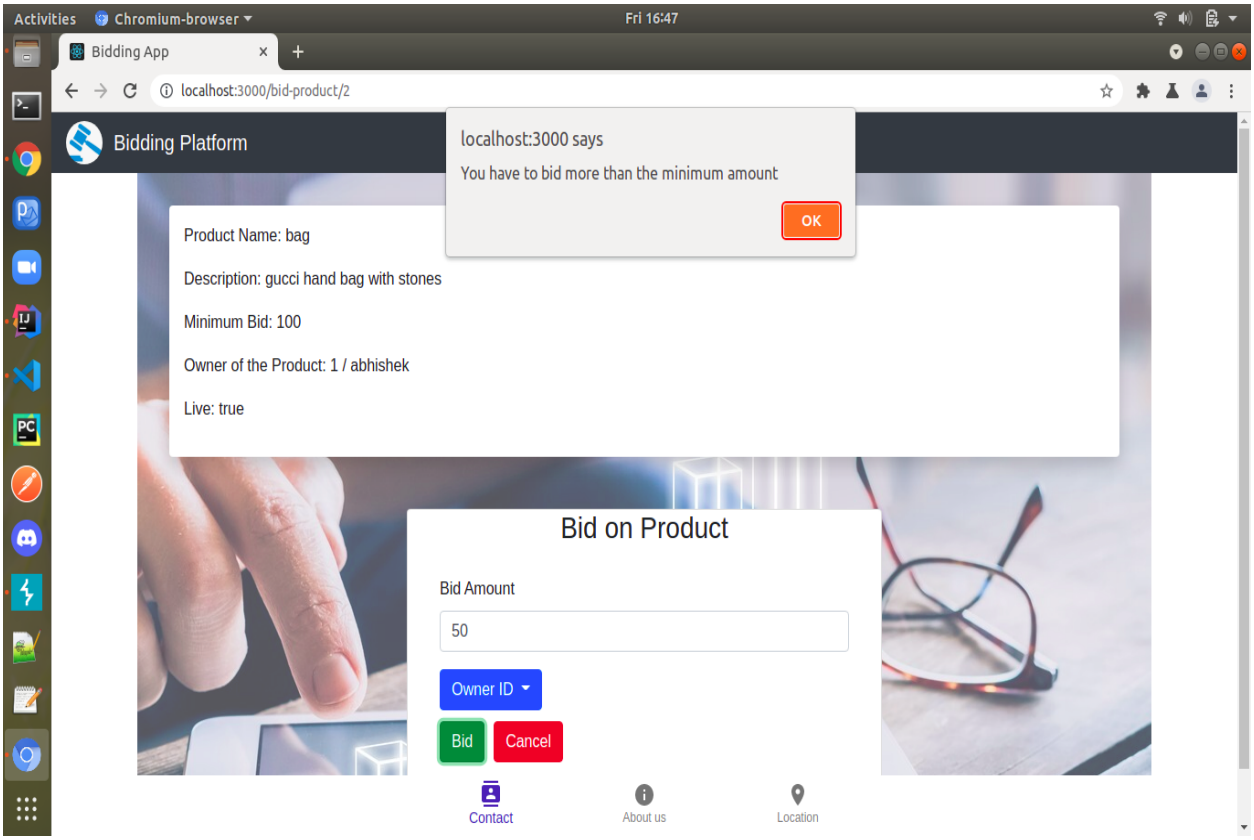


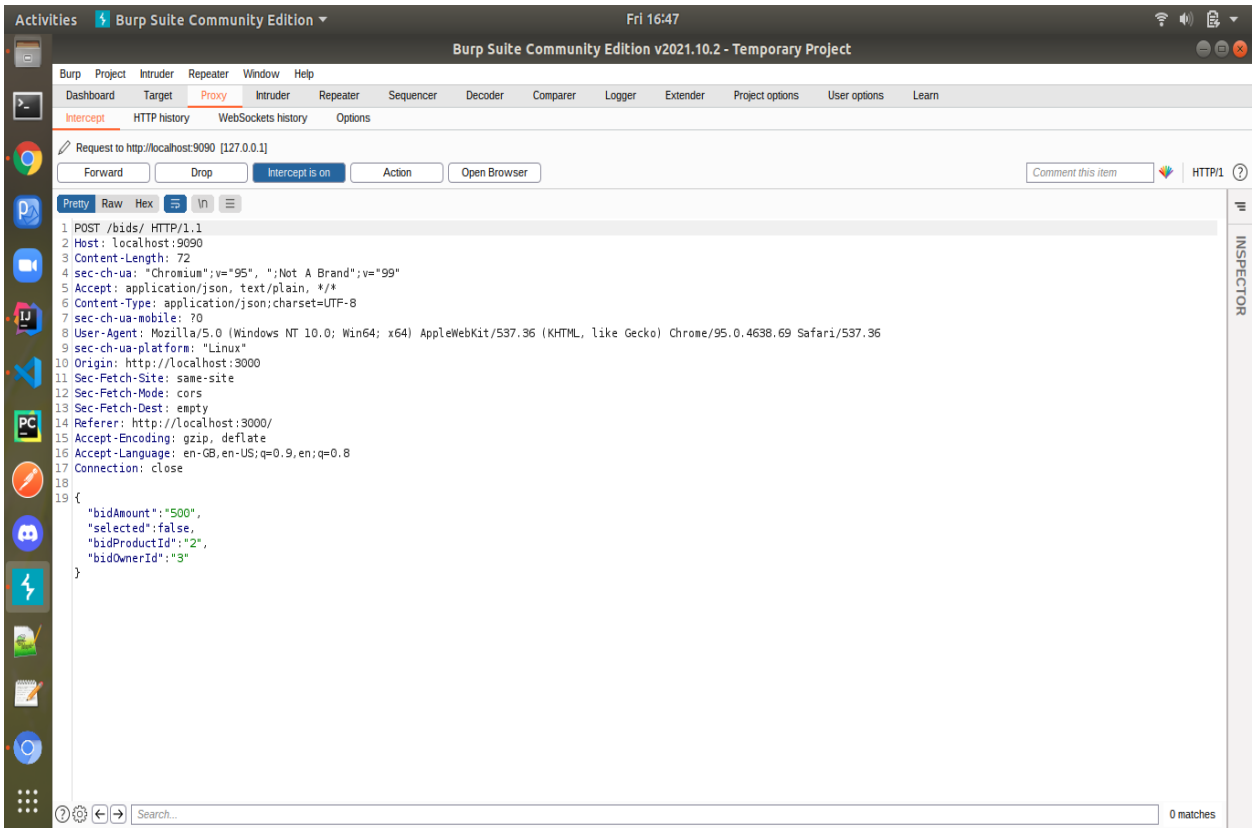

Activities
Burp Suite Community Edition
Fri 16:41
Burp Suite Community Edition v2021.10.2 - Temporary Project
Burp Project Intruder Repeater Window Help
Dashboard Target Proxy Intruder Repeater Sequencer Decoder Comparer Logger Extender Project options User options Learn
Intercept HTTP history WebSockets history Options
Request to http://localhost:9090 [127.0.0.1]
Forward Drop Intercept is on Action Open Browser
Comment this item
HTTP/1
Pretty Raw Hex
1 POST /product HTTP/1.1
2 Host: localhost:9090
3 Content-Length: 116
4 sec-ch-ua: "Chromium";v="95", ";Not A Brand";v="99"
5 Accept: application/json, text/plain, */*
6 Content-Type: application/json;charset=UTF-8
7 sec-ch-ua-mobile: ?0
8 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/95.0.4638.69 Safari/537.36
9 sec-ch-ua-platform: "Linux"
10 Origin: http://localhost:3000
11 Sec-Fetch-Site: same-site
12 Sec-Fetch-Mode: cors
13 Sec-Fetch-Dest: empty
14 Referer: http://localhost:3000/
15 Accept-Encoding: gzip, deflate
16 Accept-Language: en-GB,en-US;q=0.9,en;q=0.8
17 Connection: close
18
19 {
"productName":"mobile",
"description":"",
"minimum_bid":"0",
"active":true,
"ownerId":"6"
}
INSPECTOR
Search...
0 matches

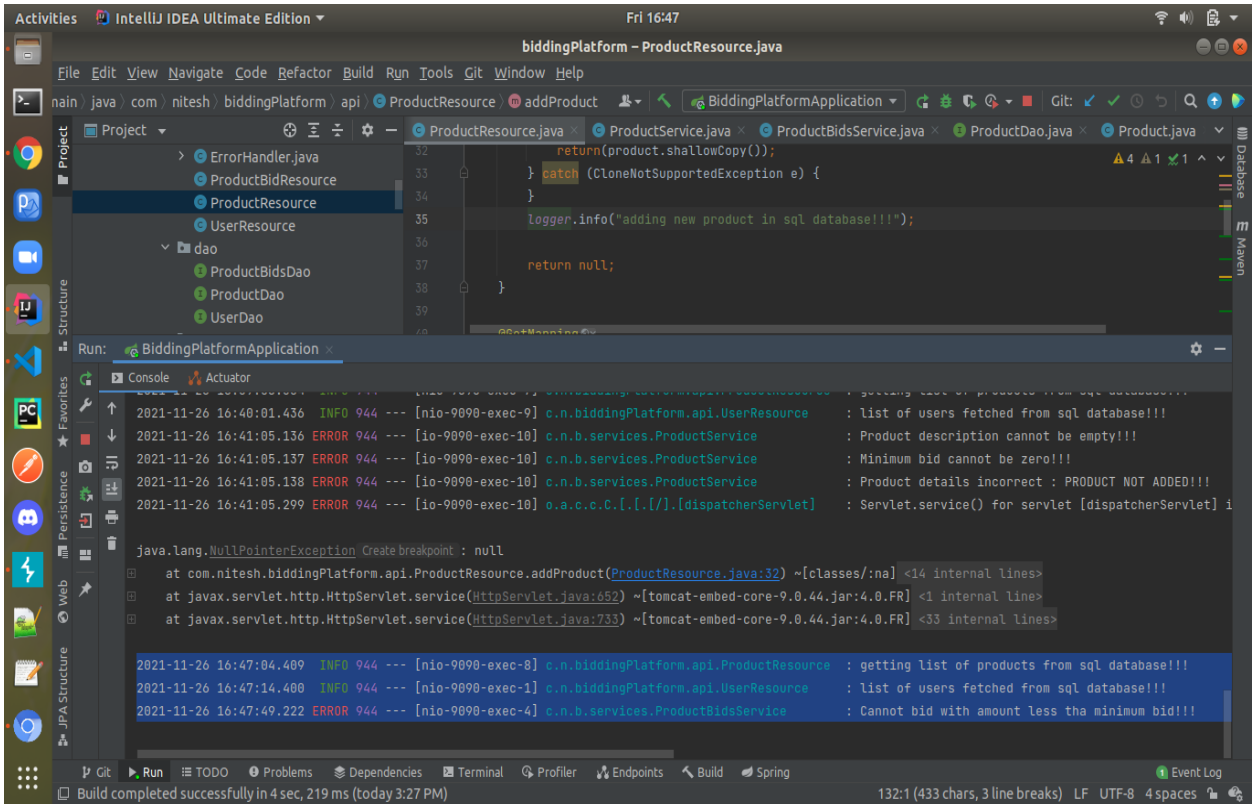

Activities
IntelliJ IDEA Ultimate Edition
Fri 16:41
biddingPlatform – ProductResource.java
File Edit View Navigate Code Refactor Build Run Tools Git Window Help
nain java com nitesh biddingPlatform api ProductResource addProduct
BiddingPlatformApplication
Git:
Project
ErrorHandler.java
ProductBidResource
ProductResource
UserResource
dao
ProductBidsDao
ProductDao
UserDao
ProductResource.java
ProductService.java
ProductBidsService.java
ProductDao.java
Product.java
return(product.shallowCopy());
} catch (CloneNotSupportedException e) {
}
logger.info("adding new product in sql database!!!");
return null;
}
Run: BiddingPlatformApplication
Console
Actuator
2021-11-26 16:30:24.656 INFO 944 --- [nio-9090-exec-5] c.n.biddingPlatform.api.UserResource : User not added to sql database!!!
2021-11-26 16:30:25.903 INFO 944 --- [nio-9090-exec-6] c.n.biddingPlatform.api.ProductResource : getting list of products from sql database!!!
2021-11-26 16:39:56.304 INFO 944 --- [nio-9090-exec-7] c.n.biddingPlatform.api.ProductResource : getting list of products from sql database!!!
2021-11-26 16:40:01.436 INFO 944 --- [nio-9090-exec-9] c.n.biddingPlatform.api.UserResource : list of users fetched from sql database!!!
2021-11-26 16:41:05.136 ERROR 944 --- [io-9090-exec-10] c.n.b.services.ProductService : Product description cannot be empty!!!
2021-11-26 16:41:05.137 ERROR 944 --- [io-9090-exec-10] c.n.b.services.ProductService : Minimum bid cannot be zero!!!
2021-11-26 16:41:05.138 ERROR 944 --- [io-9090-exec-10] c.n.b.services.ProductService : Product details incorrect : PRODUCT NOT ADDED!!!
2021-11-26 16:41:05.299 ERROR 944 --- [io-9090-exec-10] o.a.c.c.C.[.[.[/].[dispatcherServlet] : Servlet.service() for servlet [dispatcherServlet] i
java.lang.NullPointerException Create breakpoint : null
at com.nitesh.biddingPlatform.api.ProductResource.addProduct(ProductResource.java:32) ~[classes/:na] <14 internal lines>
at javax.servlet.http.HttpServlet.service(HttpServlet.java:652) ~[tomcat-embed-core-9.0.44.jar:4.0.FR] <1 internal line>
at javax.servlet.http.HttpServlet.service(HttpServlet.java:733) ~[tomcat-embed-core-9.0.44.jar:4.0.FR] <33 internal lines>
Git Run TODO Problems Dependencies Terminal Profiler Endpoints Build Spring
Event Log
Build completed successfully in 4 sec, 219 ms (today 3:27 PM)
129:1 (7747 chars, 62 line breaks) LF UTF-8 4 spaces

Test 3 : In the third test case we are trying to place a new bid for the product and following the client side validations provided by the developer. We enter a bid amount less than the minimum bid amount which gives an error due to client side scripting.

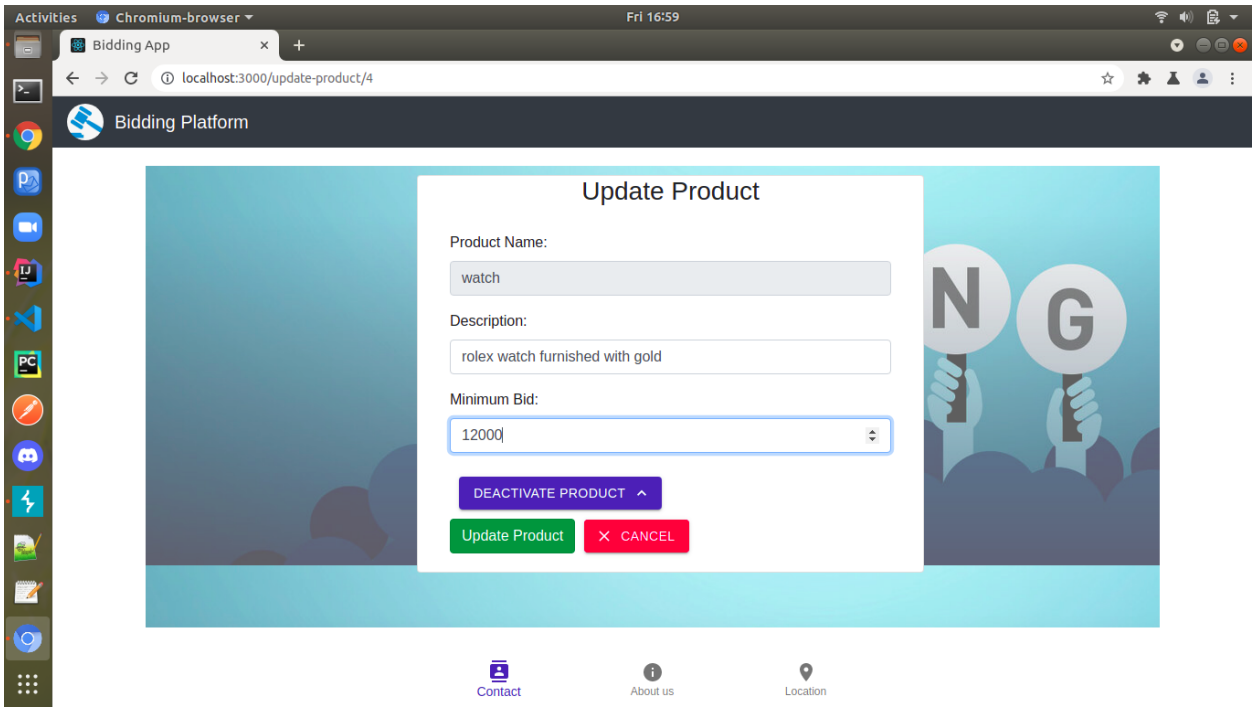
After entering the correct details we use our Burp tool and intercept the request and alter the packet. This alteration is caught by server side validations throwing an error for the bid amount to be less than the minimum amount.

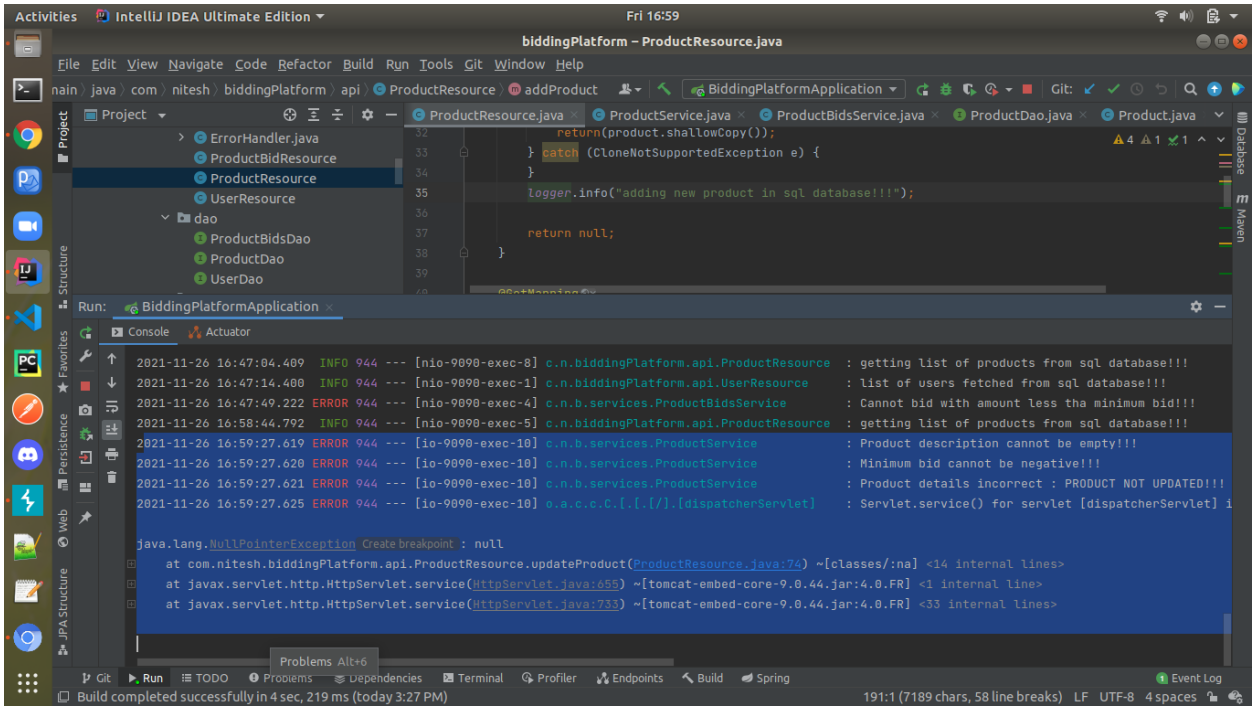



Activities
Burp Suite Community Edition
Fri 16:47
Burp Suite Community Edition v2021.10.2 - Temporary Project
Burp Project Intruder Repeater Window Help
Dashboard Target Proxy Intruder Repeater Sequencer Decoder Comparer Logger Extender Project options User options Learn
Intercept HTTP history WebSockets history Options
Request to http://localhost:9090 [127.0.0.1]
Forward Drop Intercept is on Action Open Browser
Comment this item
HTTP/1
Pretty Raw Hex
POST /bids/ HTTP/1.1
Host: localhost:9090
Content-Length: 72
sec-ch-ua: "Chromium";v="95", ";Not A Brand";v="99"
Accept: application/json, text/plain, */*
Content-Type: application/json;charset=UTF-8
sec-ch-ua-mobile: ?0
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/95.0.4638.69 Safari/537.36
sec-ch-ua-platform: "Linux"
Origin: http://localhost:3000
Sec-Fetch-Site: same-site
Sec-Fetch-Mode: cors
Sec-Fetch-Dest: empty
Referer: http://localhost:3000/
Accept-Encoding: gzip, deflate
Accept-Language: en-GB,en-US;q=0.9,en;q=0.8
Connection: close
{
"bidAmount":"500",
"selected":false,
"bidProductId":"2",
"bidOwnerId":"3"
}
INSPECTOR
Search...
0 matches


Activities
IntelliJ IDEA Ultimate Edition
Fri 16:47
biddingPlatform – ProductResource.java
File Edit View Navigate Code Refactor Build Run Tools Git Window Help
nain java com nitesh biddingPlatform api ProductResource addProduct
BiddingPlatformApplication
Git:
Project
ErrorHandler.java
ProductBidResource
ProductResource
UserResource
dao
ProductBidsDao
ProductDao
UserDao
ProductResource.java
ProductService.java
ProductBidsService.java
ProductDao.java
Product.java
return(product.shallowCopy());
} catch (CloneNotSupportedException e) {
}
logger.info("adding new product in sql database!!!");
return null;
}
Run: BiddingPlatformApplication
Console
Actuator
2021-11-26 16:40:01.436 INFO 944 --- [nio-9090-exec-9] c.n.biddingPlatform.api.UserResource : list of users fetched from sql database!!!
2021-11-26 16:41:05.136 ERROR 944 --- [io-9090-exec-10] c.n.b.services.ProductService : Product description cannot be empty!!!
2021-11-26 16:41:05.137 ERROR 944 --- [io-9090-exec-10] c.n.b.services.ProductService : Minimum bid cannot be zero!!!
2021-11-26 16:41:05.138 ERROR 944 --- [io-9090-exec-10] c.n.b.services.ProductService : Product details incorrect : PRODUCT NOT ADDED!!!
2021-11-26 16:41:05.299 ERROR 944 --- [io-9090-exec-10] o.a.c.c.C.[.[.[/].[dispatcherServlet] : Servlet.service() for servlet [dispatcherServlet] i
java.lang.NullPointerException Create breakpoint : null
at com.nitesh.biddingPlatform.api.ProductResource.addProduct(ProductResource.java:32) ~[classes/:na] <14 internal lines>
at javax.servlet.http.HttpServlet.service(HttpServlet.java:652) ~[tomcat-embed-core-9.0.44.jar:4.0.FR] <1 internal line>
at javax.servlet.http.HttpServlet.service(HttpServlet.java:733) ~[tomcat-embed-core-9.0.44.jar:4.0.FR] <33 internal lines>
2021-11-26 16:47:04.409 INFO 944 --- [nio-9090-exec-8] c.n.biddingPlatform.api.ProductResource : getting list of products from sql database!!!
2021-11-26 16:47:14.400 INFO 944 --- [nio-9090-exec-1] c.n.biddingPlatform.api.UserResource : list of users fetched from sql database!!!
2021-11-26 16:47:49.222 ERROR 944 --- [nio-9090-exec-4] c.n.b.services.ProductBidsService : Cannot bid with amount less tha minimum bid!!!
Git Run TODO Problems Dependencies Terminal Profiler Endpoints Build Spring
Event Log
Build completed successfully in 4 sec, 219 ms (today 3:27 PM)
132:1 (433 chars, 3 line breaks) LF UTF-8 4 spaces

Test 4 : Following the previous test cases this test was done for updating the product details.





Conclusion:

The basic idea in bypass testing is to let a tester save and modify the HTML. This way, client side checking/validation done routinely is by-passed and the modified data is sent to the server. It can be used to see if the server crashes on the modified data. Checks for security and robustness.Also checks for common mistakes in inputs.

Bypass testing modifies inputs and can be done at the client side or server side. Client side inputs are safer and easier to handle and server side inputs can be modified too, but can be risky if they corrupt data in the server.

The effectiveness of the test cases for a web application can be determined by the response of the server on the test cases as represented below :

1. Valid responses: Invalid inputs are adequately processed by the server. Server provides an explicit message regarding the violation or server provides a generic error message.
2. Effectiveness: Server ignores the invalid input.Invalid inputs cause abnormal server error.
3. Exposure: Invalid inputs are not recognized by the server and abnormal software behavior is exposed to users. Could even result in corruption of data in the server.

THANKS !!!

Abhishek Garg	**Nitesh Jain**	**Mohd Asad Ansari**
(MT2020021)	**(MT2020118)**	**(MT2020147)**