1. What is usability testing in web testing?

Ans -

Usability Testing is a technique used to evaluate a product (in this case a website) by testing it on users. Most people who set up a usability test carefully construct a scenario wherein a person performs aTesting list of tasks that someone who is using the website for the first time is likely to perform.

2. Explain the difference between HTTP and HTTPS?

Ans -

- HTTP URL in your browser's address bar is http:// and the HTTPS URL is https://.
- HTTP is unsecured while HTTPS is secured.
- HTTP sends data over port 80 while HTTPS uses port 443.
- HTTP operates at application layer, while HTTPS operates at transport layer.
- No SSL certificates are required for HTTP, with HTTPS it is required that you have an SSL certificate and it is signed by a CA.
- HTTP doesn't require domain validation, whereas HTTPS requires at least domain validation and certain certificates even require legal document validation.
- No encryption in HTTP, with HTTPS the data is encrypted before sending.

3. Write the test scenarios for testing a web site?

Ans -

- What is the expected load on the server (e.g., number of hits per unit time)
- What kind of performance is required under each load condition (such as web server response time, database query response times)?
- What kind of tools will be required for performance testing (such as web load testing tools, other tools already in-house that can be adapted, web robot downloading tools, etc.)?
- Who is the target audience? What kind of browsers will they be using? What kind of connection speeds will they be using? Are they intra-organizations (thus likely with high connection speeds and similar browsers) or Internet-wide (thus with a wide variety of connection speeds and browser types)?
- What kind of performance is expected from the client-side (e.g., how fast should pages appear, how fast should animations, applets, etc. load and run)?
- Will the downtime for server and content maintenance/upgrades be allowed? If so, then how much?
- What kind of security (firewalls, encryption, passwords, etc.) will be required and what is it expected to do? How can it be tested?
- How reliable are the site's Internet connections required to be? And how does that affect the backup system or redundant connection requirements and testing?
- What process will be required to manage updates to the web site's content?
- What are the requirements for maintaining, tracking, and controlling page content, graphics, links, etc.?
- Which HTML specification will be adhered to? How strictly? What variations will be allowed for targeted browsers?

• Will there be any standard requirements for page appearance and/or graphics throughout a site or parts of a site??
• How will internal and external links be validated and updated? And how often? Will it happen?
• Can testing be done on the production system, or will a separate test system be required?
• How are browser caching, variations in browser option settings, dial-up connection variability, and real-world internet ‘traffic congestion' problems to be accounted for testing?
• How extensive or customized are the server logging and reporting requirements; are they considered as an integral part of the system and do they require testing?
• How are CGI programs, applets, javascript, ActiveX components, etc. to be maintained, tracked, controlled, and tested?
• Pages should be 3-5 screens max unless the content is highly focused on a single topic. If larger, provide internal links within the page.
• The page layouts and design elements should be consistent throughout a site so that it's clear to the user that they are still on a site.
• Pages should be as browser-independent as possible, or pages should be provided or generated based on the browser type.
• All pages should have links external to the page; there should be no dead-end pages.
• The page owner, revision date, and a link to a contact person or organization should be included on each page.

4. Write a few Test Cases on GMail functionality.
Ans -

https://docs.google.com/spreadsheets/d/1Z8dOm8mxiUnpeTcJUHuCiIUc9xoGL_89RHosnaQi-fc/edit?usp=sharing

5. Write any 5 common ATM Machine functionality.
Ans -
• Users should be able to withdraw some cash if an amount is present in account.
• Users should be able to check account balance.
• Users should be able to Transfer some amount of money from his/her account to another account
• Users should be able to change their ATM pin.
• Users should be able to change their registered mobile number.

6. Give some examples of web applications that are used in our day to day life.
Ans -
• Newers World
• Gmail
• Youtube
• Facebook
• Whatsapp

7. What are the advantages of Using Cookies?

Ans -

- Cookies are simple to use and implement.
- Occupies less memory, do not require any server resources and are stored on the user's computer so no extra burden on the server.
- We can configure cookies to expire when the browser session ends (session cookies) or they can exist for a specified length of time on the client's computer (persistent cookies).
- Cookies persist a much longer period of time than Session state.

8. What is XSS and how can we prevent it?

Ans -

XSS is known as Cross-site scripting, it is a web security vulnerability that allows an attacker to compromise the interactions that users have with a vulnerable application. It allows an attacker to circumvent the same origin policy, which is designed to segregate different websites from each other. Cross-site scripting vulnerabilities normally allow an attacker to masquerade as a victim user, to carry out any actions that the user is able to perform, and to access any of the user's data. If the victim user has privileged access within the application, then the attacker might be able to gain full control over all of the application's functionality and data.

There are two types of XSS :-

1. Reflected XSS : It is the simplest variety of cross-site scripting. It arises when an application receives data in an HTTP request and includes that data within the immediate response in an unsafe way.
2. Stored XSS : It is also known as persistent or second-order XSS. It arises when an application receives data from an untrusted source and includes that data within its later HTTP responses in an unsafe way.

We can prevent XSS by :-

- Filter input on arrival. At the point where user input is received, filter as strictly as possible based on what is expected or valid input.
- Encode data on output. At the point where user-controllable data is output in HTTP responses, encode the output to prevent it from being interpreted as active content. Depending on the output context, this might require applying combinations of HTML, URL, JavaScript, and CSS encoding.

9. Write a few Cross Browsing Testing TCs for any website.

Ans -

https://docs.google.com/spreadsheets/d/1GOPHm25x_UA6FG4pYNXxwgonI60yCq3QLHIX6re1ilM/edit?usp=sharing