1. What do you understand by Device and OS fragmentation in mobile application testing?

Ans:- Mobile device fragmentation is a phenomenon that occurs when some mobile users are running older versions of an operating system, while other users are running newer versions.

OS Fragmentation arises due to different versions of OSs. Both Android & iOS have around 10+ different versions available. Developers have to make sure that their app is running on all different versions of OS(whether it is Android, iOS or any other OS).

2. List down the things you cannot test on an Emulator but on Real Devices?

Ans:-

The things you cannot test on an Emulator but on Real Devices are:

* The Emulator cannot emulate battery issues, network connectivity and other real-time data like GPS, sensors, gestures, touch force, etc.
* It cannot emulate the performance of an app in terms of incoming calls, SMS, etc.
* There’s no way to check the color/contrast of display under different weather conditions
* Touchscreen issues cannot be emulated either
* Not all mobile apps can be tested on emulators

3. What All Major Networks to Be Considered While Performing Application Testing?

Ans:-

You should test the application on 4G, 3G, 2G and WIFI. 2G is a slower network, it's good if you verify your application on a slower network also to track your application performance.

4. What are the google core quality guidelines for Mobile Apps, Explain guidelines for Functionality?

Ans:- The google core quality guidelines for Mobile Apps are

Visual Experience, Functionality, Performance & Stability, Privacy & Security, and Google Play. Google says an ideal app should offer a polished user experience, load quickly without dropping too many frames, handle user data with care and ensure compliance with Google Play Store policy.

In the Functionality Section, Google has several recommendations for media apps, including using the HEVC video compression standard for video encoding and supporting picture-in-picture mode. It also says apps should use the Android Share Sheet for content sharing, minimize their background activity and ensure their compatibility with Doze and App Standby. On this note, Google also warns that developers should expect restrictions on background activities to continue to tighten in the future Android versions.

5.Write down test scenarios for Push notification.

Ans:- Test cases for Push Notification are:

* Verify the notification received even if the app is open.
* Verify the notification received even if the app is closed.
* Verify the notification received even if the device is locked.
* Verify the notification received even if the user is playing the game.
* Verify the notification received even if another app is in use.
* Verify the notification received in multiple time zones.
* Verify the notification is clickable & it opens up the notification.
* Verify the sound, vibration & blinking.
* Verify the notification when the user is not SignIn.
* Verify the notification should get removed from the notification bar once the notification is opened.
* Verify when the notification comes & device is locked, if the user opens the device with sliding the notification then the application should get launched & navigate to that specific notification (App in the background).
* Verify when notification comes in foreground, then the notification will display directly on the app (App in foreground).
* Verify repeated notification multiple times and check notification is in the order.
* Verify recent notification messages always should be on the top in stack.
* Verify when the user didn't click on a banner then the notification count should get updated on the icon of the application.

6. How to record a log from an Android/iOS device?

Ans:- To record a log from an Android/iOS device :

For Android: ADB(Android Debug Bridge)

For IOS Device : Xcode

Steps to record a log from an Android Device:-

Follow the steps below to obtain debug-logs from Android devices on your Windows PC.

On your Windows Computer, download and Install Android SDK.

Once the installation completes, Android SDK will launch automatically. Select and Install Android Platform Tools.

Connect your Android device to your Windows PC via USB cable.

Open a new command prompt session, change the working directory to the path where Android SDK is installed (cd [path where Android SDK resides]/platform-tools).

First, run the following command

adb.exe kill-server

Next, run the command to list the devices connected to your PC. Note down the device id of your device.

adb.exe devices

Run,

adb.exe logcat -v threadtime [device id] > C:\android-debug.log

Enter the device id obtained in the previous step in place of [device id].

Perform any actions on your Android device.

To stop the adb process, press Ctrl + C on the command prompt window.

The debug data can be viewed from c:\android-debug.log file.

Steps to record a log from an Android Device:-

First, install Xcode on your mac machine. Next, launch Xcode.

Connect your iOS device to the Mac through USB.

Launch Xcode. Go to Windows > Devices and Simulators.

Reproduce the problem you encountered.

Choose your device from the devices section on the left side of the screen.

Click on the up-triangle on the bottom of the screen to view device logs.

Click on the down arrow on the bottom right of the screen to save the device logs as a file.

Select View Device Logs button under the Device Information section on the right-hand panel to view crash logs.

Under the Process column on the left, identify and select your app and click on Crash Log to see the contents.

Right click the corresponding app entry on the Process Column and click on Export Log to save the crash log.

7. What is deep linking? What do you understand by deferred deep linking?

Ans:- Deep links are a type of link that send users directly to an app instead of a website or a store. They are used to send users straight to specific in-app locations, saving users the time and energy locating a particular page themselves – significantly improving the user experience.

DEFERRED DEEP LINKING:

Deep links are a smart way to drive conversions while offering a positive user experience, but what happens if the user is deep linked into an app they don’t have installed, This is when deferred deep links come into play. If a user clicks on a deep link and doesn’t have the app installed, they can be deferred to the App Store instead. The genius of deferred deep linking is that when that user installs and opens up the app, they can still be sent to the in-app location where you initially wanted them to land. For example, this could be a specific level in your gaming app, or a page from your product catalog for e-commerce.