Q1: What is the test automation framework? What is selenium? how does it work? and why do you need it?

Test Automation Framework:

A testing framework is a set of guidelines or rules used for creating and designing test cases. A framework is comprised of a combination of practices and tools that are designed to help QA professionals test more efficiently.

These guidelines could include coding standards, test-data handling methods, object repositories, processes for storing test results, or information on how to access external resources.

While these are not mandatory rules and testers can still script or record tests without following them, using an organized framework typically provides additional benefits that would otherwise be missed out on.

Selenium:

Selenium Framework is a suite of automation testing tools that is based on the JavaScript framework. It could run the tests directly on the target browser, drive the interactions on the required web page and rerun them without any manual input.

Working:

The WebDriver protocol has a local end ('client') which sends the commands (test script) to a browser-specific driver. The driver executes these commands on its browser-instance. So, if the test script calls for execution on Chrome and Firefox, the Chrome Driver will execute the test on Chrome, the Gecko Driver will do the same on Firefox.

Need Of Selenium:

When there are a large number of data sets to be tested for the web application. Then you must opt for a data-driven framework, which separates them from the actual code.

For instance, when there are more functionalities to be tested for the web application. It is suggested that the team goes for a keyword driven framework, where the operations are stored in a separate table in the form of keywords. These keywords are called to use a specific functional operation.

On occasions when data sets and functionalities both are high in number, use a hybrid driven framework to avoid complexity.

Q2: The most common tools that are used for configuration management are packer and ansible. You need to concisely compare both of them.

Packer	Ansible
Create identical machine images for multiple platforms from a single source configuration. Packer automates the creation of any type of machine image.	Radically simple configuration-management, application deployment, task-execution, and multi-node orchestration engine. Ansible is an IT automation tool.
It embraces modern configuration management by encouraging you to use automated scripts to install and configure the software within your Packer-made images.	It can configure systems, deploy software, and orchestrate more advanced IT tasks such as continuous deployments or zero downtime rolling updates. Ansible’s goals are foremost those of simplicity and maximum ease of use.
Super fast infrastructure deployment. Packer images allow you to launch completely provisioned and configured machines in seconds, rather than several minutes or hours.	Ansible's natural automation language allows sysadmins, developers, and IT managers to complete automation projects in hours, not weeks.
Multi-provider portability. Because Packer creates identical images for multiple platforms, you can run production in AWS, staging/QA in a private cloud like OpenStack, and development in desktop virtualization solutions such as VMware or VirtualBox.	Ansible uses SSH by default instead of requiring agents everywhere. Avoid extra open ports, improve security, eliminate "managing the management", and reclaim CPU cycles.
Improved stability. Packer installs and configures all the software for a machine at the time the image is built. If there are bugs in these scripts, they'll be caught early, rather than several minutes after a machine is launched.	Ansible automates app deployment, configuration management, workflow orchestration, and even cloud provisioning all from one system.

References:

https://smartbear.com/learn/automated-testing/test-automation-frameworks/

https://www.browserstack.com/selenium

https://www.browserstack.com/guide/selenium-framework#:~:text=Selenium%20Framework%20is%20a%20suite,them%20without%20any%20manual%20input.