What is the purpose of the core module in AEM?

Ans: **The core module in AEM is basically the brain of the project, handling all the backend stuff like Sling Models, OSGi services, and servlets. It talks to the JCR, processes data, and connects with other systems when needed. It also manages events, workflows, and business logic to keep things running smoothly. In short, it keeps AEM fast, modular, and easy to manage**

What kind of files and code can be found in the core folder?

Ans: **All the backend files are present in the core folder.The sling models OSGI services and serverlet are present in the core folder. Any backend process is done through the core folder.Within core folder ther are two folders src and target along with a pom.xml file.**

Explain the role of ui.apps in AEM projects.

Ans: **The ui.apps folder contains all the frontend related files of the files like css and javascript.**

How are components structured in the ui.apps folder?

Ans:the components folder has all the required assets for the frontend of the project. It contains forms, elements etc.

Hello World Component:

- Where is the Hello World component located in both core and ui.apps?
- Ans: in ui.apps it is located in the components folder whereas in core it is located in the models .
    - Explain the Java class (in core) for the Hello World component.

      Ans :The HelloWorldModel is a Sling Model in AEM that retrieves the text property from JCR and generates a dynamic greeting message. If a value is provided for text, it customizes the message to say "Hello, [text]!". Otherwise, it defaults to "Hello, World!". This model is then used in HTL to render the message dynamically on the frontend

      How does the HTL script work in ui.apps for Hello World?\

      Ans: It retrieves the message property from the model, which is dynamically generated based on the text value stored in JCR. If a value is provided, the script displays "Hello, [text]!"; otherwise, it defaults to "Hello, World!"
- How are properties and dialogs defined for this component?
- Ans: In AEM, properties and dialogs for the Hello World component are defined in the ui.apps module using XML files inside the component’s folder.

What are the different types of AEM modules (core, ui.apps, ui.content, etc.)?

The core module handles backend logic using Sling Models, OSGi services, and servlets. The ui.apps module defines frontend components, dialogs, and client libraries. The ui.content module stores AEM site content, templates, and initial content. Additional modules like ui.frontend, ui.config, it.tests, and dispatcher help with frontend development, configurations, testing, and caching.

How does Maven build these modules?

Ans: Maven kicks things off with the parent POM, which sorts out dependencies and build order. The core module turns Java code into a JAR, while ui.apps and ui.content get packed into AEM installable files. Then, the Content Package Maven Plugin pushes everything into AEM. Finally, the all module bundles it all into one big ZIP for easy deployment

Explain the build lifecycle of Maven in the context of AEM.

Ans: First, it cleans up old build files so everything starts fresh. Then, it checks if all the necessary files and dependencies are in place. After that, the core module's Java code gets compiled, while ui.apps and ui.content get packed as installable AEM files. Next, tests are run to catch any errors early. If everything looks good, it packages everything properly. Then comes the install phase, where the built files get stored locally, and finally, in the deploy phase, the package is sent to AEM for installation

How are dependencies managed in pom.xml?

Ans: dependencies are managed under the <dependencies> section using groupID ,artifactID and version.

Why is Maven used instead of other build tools?

Ans: Maven is used instead of other build tools in AEM because it simplifies dependency management.

What advantages does Maven offer for AEM development?

Ans: Maven makes AEM development easier by handling dependencies automatically, so you don’t have to manually download JAR files. It standardizes project structure, making it easier to manage large AEM projects. With build automation, you can compile code, deploy bundles, and package content with a single command

How does Maven help in managing dependencies and plugins in AEM projects?

Ans; using the pom.xml file

What does mvn clean install do in an AEM project?

Ans/:it installs all the required dependencies for the maven project

How to deploy packages directly to AEM using Maven commands?

Ans;It can be done by using this command “mvn clean install -PautoInstallSinglePackage”

Explain the purpose of different Maven profiles in AEM (autoInstallPackage, autoInstallBundle).

Ans: Maven profiles in AEM help deploy specific parts of the project efficiently. autoInstallPackage installs the full project (backend + frontend), while autoInstallBundle only deploys the core OSGi bundle (Java code). autoInstallSinglePackage does a quick full deployment in one step. These profiles make AEM development faster by avoiding unnecessary re-installations

What is the purpose of dumplibs in AEM?

Ans: dumplibs in AEM is a debugging tool for client libraries (CSS & JS). It helps check if clientlibs are loading correctly and shows their dependencies

How can you view client libraries using dumplibs?

Ans:by using this url http://localhost:4502/libs/granite/ui/content/dumplibs.html

Explain how client libraries are structured in AEM.

Ans: Client libraries in AEM are stored under /apps/project-name/clientlibs and organized into folders like js, css, and resources. Each clientlib has a .content.xml file and a cq:ClientLibraryFolder node to define categories. The allowProxy=true setting enables access via /etc.clientlibs