Angular has two main compilers:

AOT (Ahead-of-Time) Compiler: The AOT compiler compiles the Angular components and templates at build time, before they are served to the browser. This results in faster loading times, since the browser doesn't have to spend time compiling the code. It also enables Angular to catch many errors during the build process, instead of at runtime.

JIT (Just-in-Time) Compiler: The JIT compiler compiles the Angular components and templates in the browser at runtime. This makes development easier and faster, as you can see the changes you make in your code reflected immediately in the browser. However, this can result in slower initial loading times, since the browser has to compile the code before it can run it.

In general, it's recommended to use the AOT compiler for production builds and the JIT compiler for development builds, as the AOT compiler provides better performance and security benefits.

Angular provides several build tools that can be used to create and manage the build process for Angular applications. Some of the most commonly used build tools in Angular are:

ng build: This is the basic command used to build an Angular application. It takes the source files of an Angular application, compiles them, and creates a production-ready build of the application in the "dist" folder.

ng serve: This command builds the application and starts a development server to serve the application in the browser. The server automatically reloads the page whenever changes are made to the source files.

ng test: This command runs the unit tests for an Angular application. Unit tests are used to test individual components and services in an Angular application, and ensure that they work as expected.

ng lint: This command checks the source code of an Angular application against a set of coding standards, and reports any violations. This helps to maintain the quality and consistency of the codebase.

ng e2e: This command runs end-to-end tests for an Angular application. End-to-end tests simulate user interactions with the application and test the entire application flow, from start to finish.

These build tools are part of the Angular CLI (Command Line Interface), which is a command-line tool that makes it easy to create, build, and manage Angular applications. The Angular CLI provides a consistent interface for managing various aspects of an Angular application, making it easier for developers to work with Angular and reducing the need for manual configuration.

Webpack is a popular module bundler for JavaScript applications. In Angular, it is often used to bundle and build the application for production.

Webpack takes the various dependencies and modules in an Angular application and packages them into a single file (or a few files) that can be efficiently served to the browser. This helps to reduce the number of requests the browser needs to make to load the application, resulting in faster loading times.

In Angular, Webpack is used to compile and bundle the application's TypeScript, HTML, and CSS files, as well as any other assets such as images and fonts. Webpack can also be configured to optimize the bundled code, such as minifying and compressing it, to reduce the size of the final build.

Webpack also provides a development server that can be used during development, similar to the Angular CLI's ng serve command. This server can be configured to automatically reload the page whenever changes are made to the source files, making it easier for developers to see their changes in the browser.

In summary, Webpack is an important tool for building and deploying Angular applications in production. It helps to optimize the performance and reduce the size of the final build, making it easier for the browser to load and run the application.

EsLint: Code Quality check

ESLint is a popular linting tool for JavaScript code. In an Angular application, it can be used to enforce a consistent coding style and catch potential errors in the code before they cause problems.

ESLint works by analyzing the JavaScript code in an Angular application and checking it against a set of rules. These rules can be customized to suit the specific coding style and standards used by the development team. For example, the rules can enforce the use of semicolons, require specific naming conventions, or disallow the use of certain features that are considered problematic.

By using ESLint, developers can catch potential problems in their code early, before they cause problems in the application. This helps to maintain the quality and stability of the codebase, and reduces the need for debugging and fixing problems later in the development process.

In Angular, ESLint is often used in conjunction with other tools, such as the Angular CLI or Webpack, to automate the process of linting the code. The linting process can be run as part of the build process, or it can be run on-demand using the command line.

In summary, ESLint is a valuable tool for maintaining the quality and consistency of an Angular application's codebase. By using ESLint, developers can catch potential problems early, reducing the time and effort needed to fix them later in the development process.

An Angular application consists largely of components and its HTML templates. The components and templates must be converted to executable JavaScript before they are rendered inside the browser. This conversion will be taken care by Angular compiler.

There are two ways to compile an Angular application:

1. **Just-In-time (JIT):** Compiles Angular app in the browser at run time

2. **Ahead-Of-Time(AOT):** compiles Angular app at build time

JIT compilation is the default when we run any of the following CLI commands

1. ng build
2. ng serve

For AOT compilation, we need to append --aot option to the commands as shown below

1. ng build --aot
2. ng serve --aot

**Why AOT Compilation?**

**Faster Rendering:**With AOT compilation, the browser loads the pre-compiled application which will be rendered immediately in the browser.

**Fewer Asynchronous Requests:**AOT compilation inlines external HTML templates and CSS stylesheets into the application itself, eliminating separate ajax requests for those files.

**Smaller Framework Download Size:**As the app is pre-compiled, there is no need of downloading Angular compiler in the browser which is roughly half of the size of Angular itself. This dramatically reduces the application payload.

**Detects Template Errors Early:**The AOT compiler detects the template binding errors during the build setup itself.

**Better Security:**AOT compiles components and HTML templates before it is served to the client. With no templates to read and no risky client side HTML or Javascript evaluation, there are less chances for injection attacks.

**Deploying an application for production**

To generate an optimized build, use the option --prod in  build commands

1. ng build --prod

--prod flag will do the following during build process.

* Enables AOT compiler and pre-compiles Angular components and templates by default
* Enables production mode
* Concatenates many files into few bundles
* Minifies the app by removing extra spaces, comments etc.
* Uglifies the app by rewriting variable and function names with short names
* Removes the unused code and modules which is termed as dead code elimination

Once the build process is completed, it creates a separate folder called dist in the project structure. Production build command will bundle all the files and places it in dist folder.

You need to copy the dist folder to the folder on a remote server for deployment.