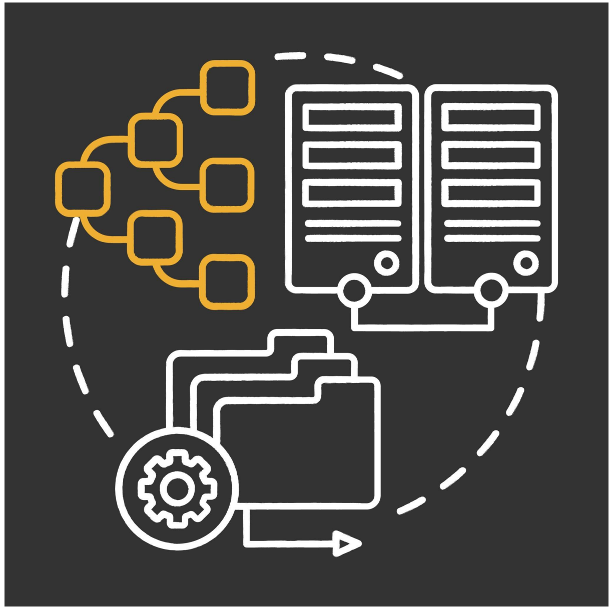
JSF(JavaServer Faces)

JSF is a MVC framework used to create the UI for server side applications. It provides a set of reusable components.



Merits and Demerits

Merits:

- Provides reusable UI components
- Allows implementation of custom components
- Provides event-driven programming model
- Manages the UI state across multiple requests

Demerits:

- Steep learning curve due to complex life cycle
- Tightly coupled to server-side state, limiting scalability
- Verbose and slow development process
- Difficult to debug with unclear error messages

Life cycle of JSF

Restore View:

This is the first phase of JSF life cycle. In this phase the component tree is created and is stored in the FacesContext instance.

Apply Request Values:

In this phase component in the component tree will retrieve their values from request.

Process Validation:

In this phase all convertors and validators are executed. If validation passes all other phases executes normally otherwise JSF implementation adds an error message to the FacesContext instance and render response phase executes directly.

Update Model Values:

In this phase bean properties are updated using corresponding components values.

Invoke Application:

In this phase the business logic is executed after form submission

Render Response:

In this phase response is sends to browser.

Managed Bean

A managed bean is a java bean class which contains data and business logic. A managed bean is managed by JSF framework and acts as a model for JSF framework and act as a model for JSF application. Before using a managed bean we have to configure it either in faces-config.xml file or using annotations.

Managed Bean Scope

Request Scope:

Bean created when a request for this bean comes and destroyed after complete response.

Merits:

- Fresh instance created per HTTP request-lightweight
- No memory overhead after response is sent

Demerits:

- Data is lost after page reload or redirect.
- Not suitable for multi-page forms or wizards.

View Scope:

Bean Created when a request involving this bean comes and destroyed when view changed.

Merits:

- Bean lives as long as the user stays on the same view.
- Ideal for AJAX-based partial updates.

Demerits:

- Requires Java serialization (must implement Serializable).
- Slightly more memory than request scope.

Session Scope:

Bean created when a request for this bean comes and destroyed when session terminates.

Merits:

- Bean persists across multiple pages and requests.
- Useful for login sessions or shopping carts.

Demerits:

- Higher memory usage (one bean per user session).
- Can lead to stale data if not managed properly.

Application Scope:

Bean created when a request involving this bean comes and remains for whole duration of web application.

Merits:

- Shared across all users → memory efficient for static data.
- Ideal for config or shared resources.

Demerits:

- Not thread-safe by default (must handle concurrency).
- Data is same for all users → no personalization.

AJAX(Asynchronous Javascript and XML)

AJAX is a web development technique that allows a web page to send and receive data from the server **asynchronously**, without reloading the entire page. It uses JavaScript for communication and was originally designed to work with XML, though now **JSON** is commonly used. AJAX improves performance and enhances the user experience by allowing partial updates to web content.

Benefits of ajax:

- Only a specific part of the page is updated, not the whole page.
- Makes web applications faster and more interactive.
- Transfers only necessary data, minimizing traffic and load.
- Enables real-time content updates without delay.
- Developers can decide exactly which components to update.

Use of AJAX in JSF

Basically AJAX is used in JSF by two ways:

- Using<f:ajax>(Standard JSF)

  In JSF, `<f:ajax>` allows partial page updates. When you set `event="keyup"`, it triggers an AJAX call every time a key is released. The `render="output"` ensures only the specified component is updated, keeping the rest of the page unchanged.

- Using <p:ajax> in PrimeFaces

  In PrimeFaces, `<p:ajax>` simplifies AJAX by working directly inside components like `<p:inputText>`. It listens for events like `keyup` and updates the specified target using `update="output"`, enabling dynamic and responsive UI behavior.