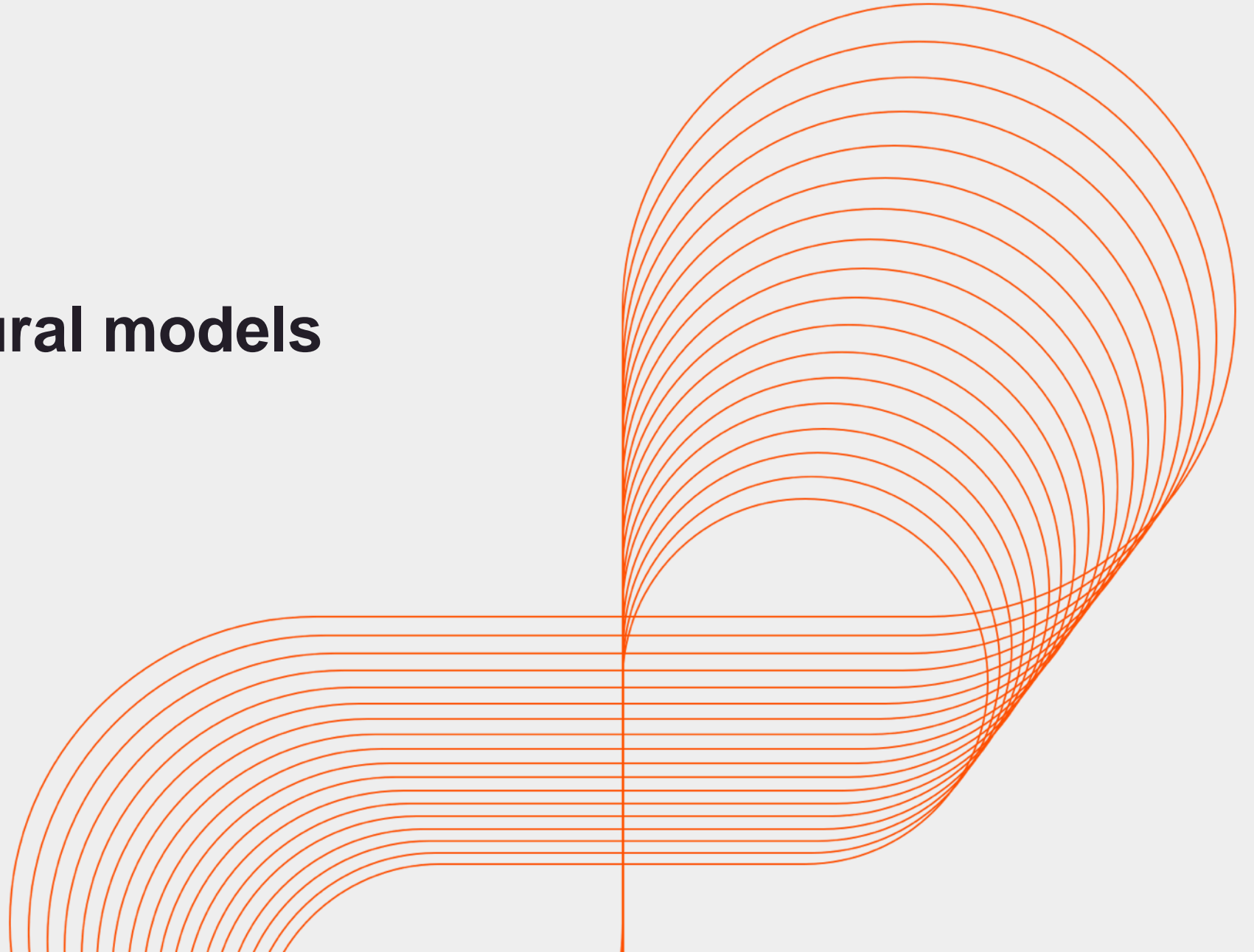




Persistent

JSP architectural models

Persistent University



Agenda

- Separation of concerns
- Describe different JSP-based application models
 - Model I
 - Model II
- The front controller

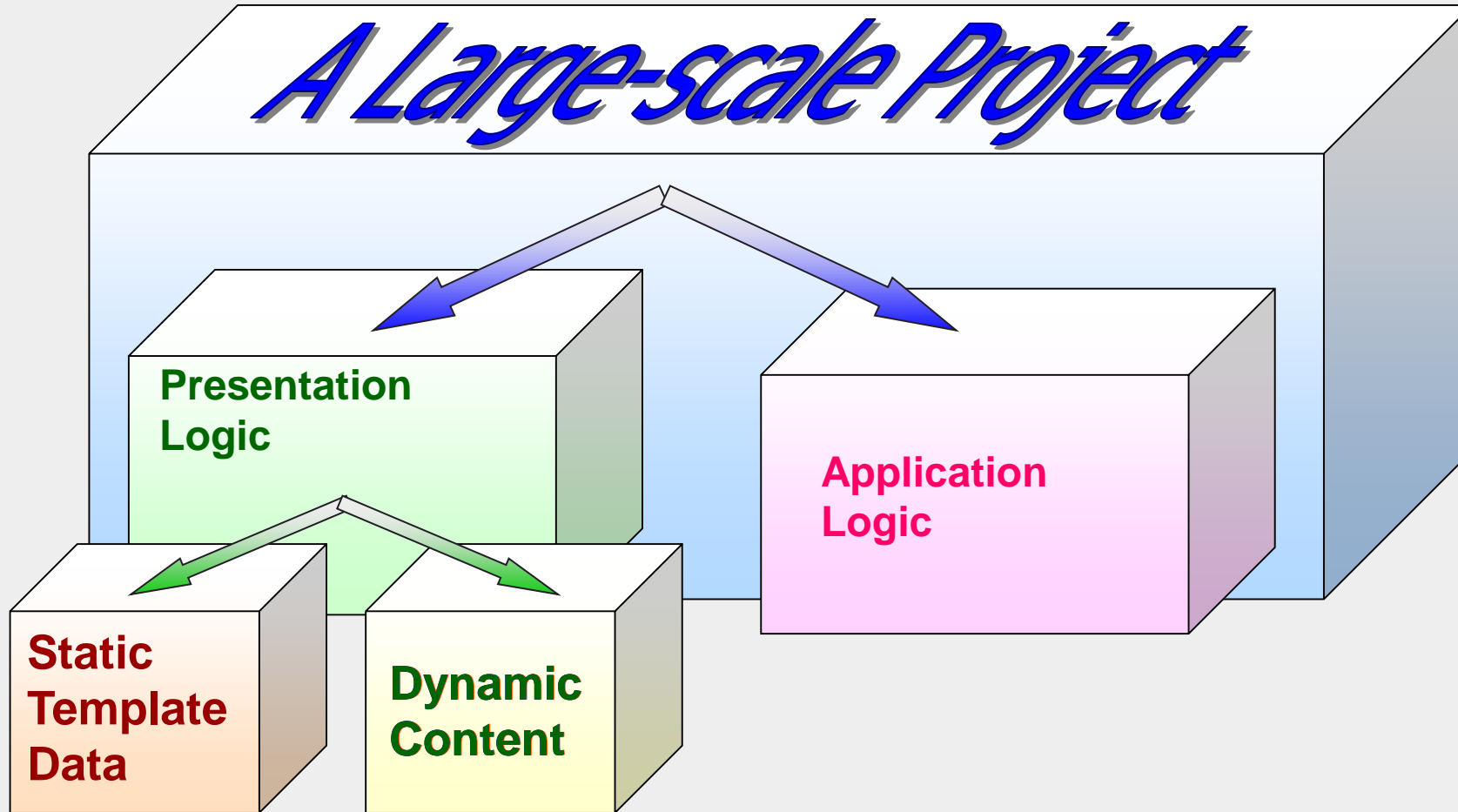
Separation of concerns

- Multiple roles and responsibilities exist in an application development project
- A page designer role –
 - Designs the HTML pages
- A software developer –
 - Creates the software using a programming language
- In small-scale projects the roles may intermingle
- For large-scale projects the main problem is complexity of the application design

The building blocks of a web application

A decorative graphic consisting of a horizontal orange line that extends from the left edge of the slide. At its right end, it meets a vertical orange line that extends downwards to the bottom edge. A large orange circle is positioned in the upper right quadrant, with its left edge touching the horizontal line and its bottom edge touching the vertical line.

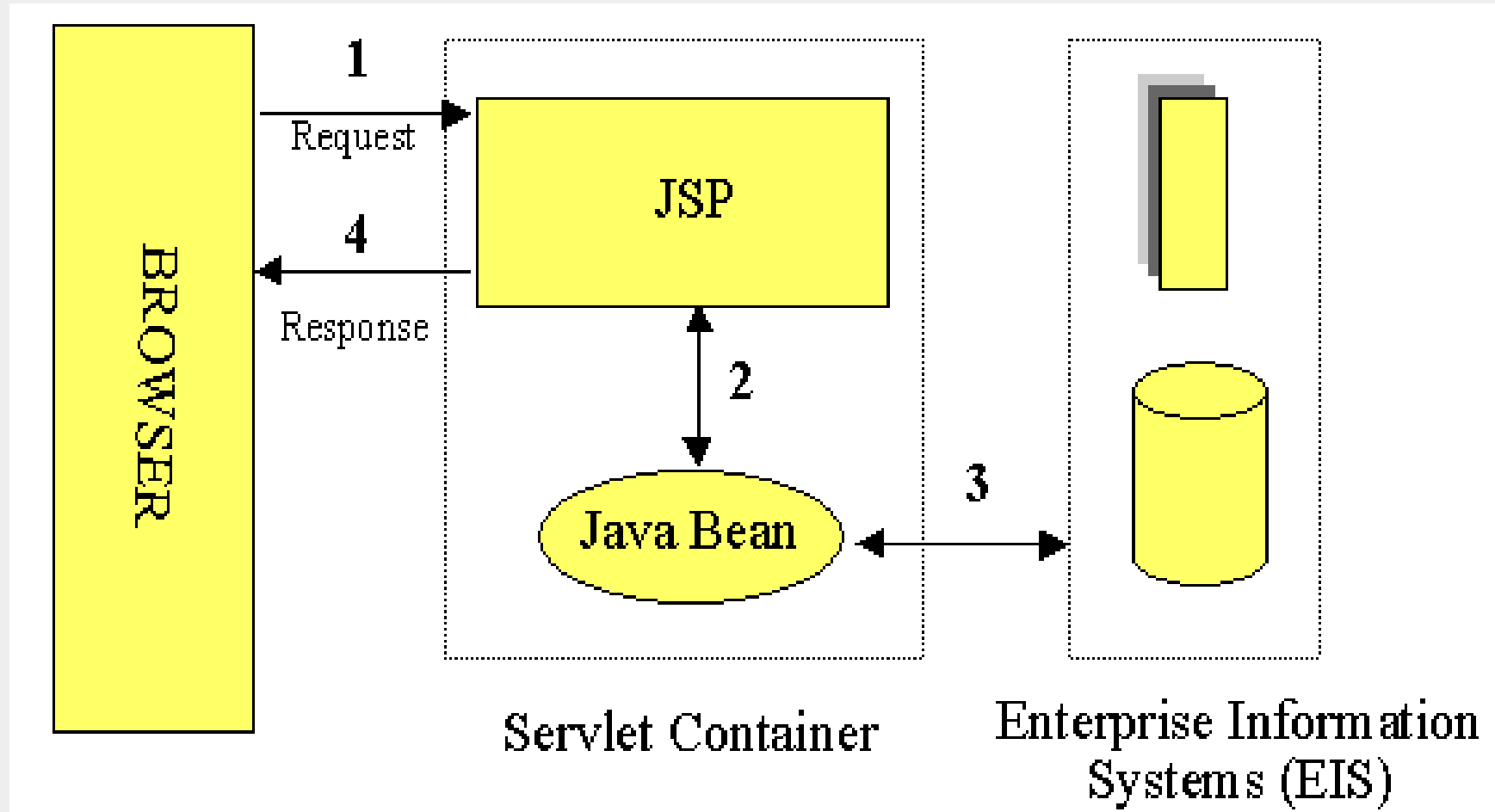
Separation of concerns....



JSP access models

- The early JSP specifications advocated two philosophical approaches, popularly known as Model 1 and Model 2 architectures, for applying JSP technology.
- These approaches differ essentially in the location at which the bulk of the request processing was performed, and offer a useful paradigm for building applications using JSP technology.

Model 1 architecture



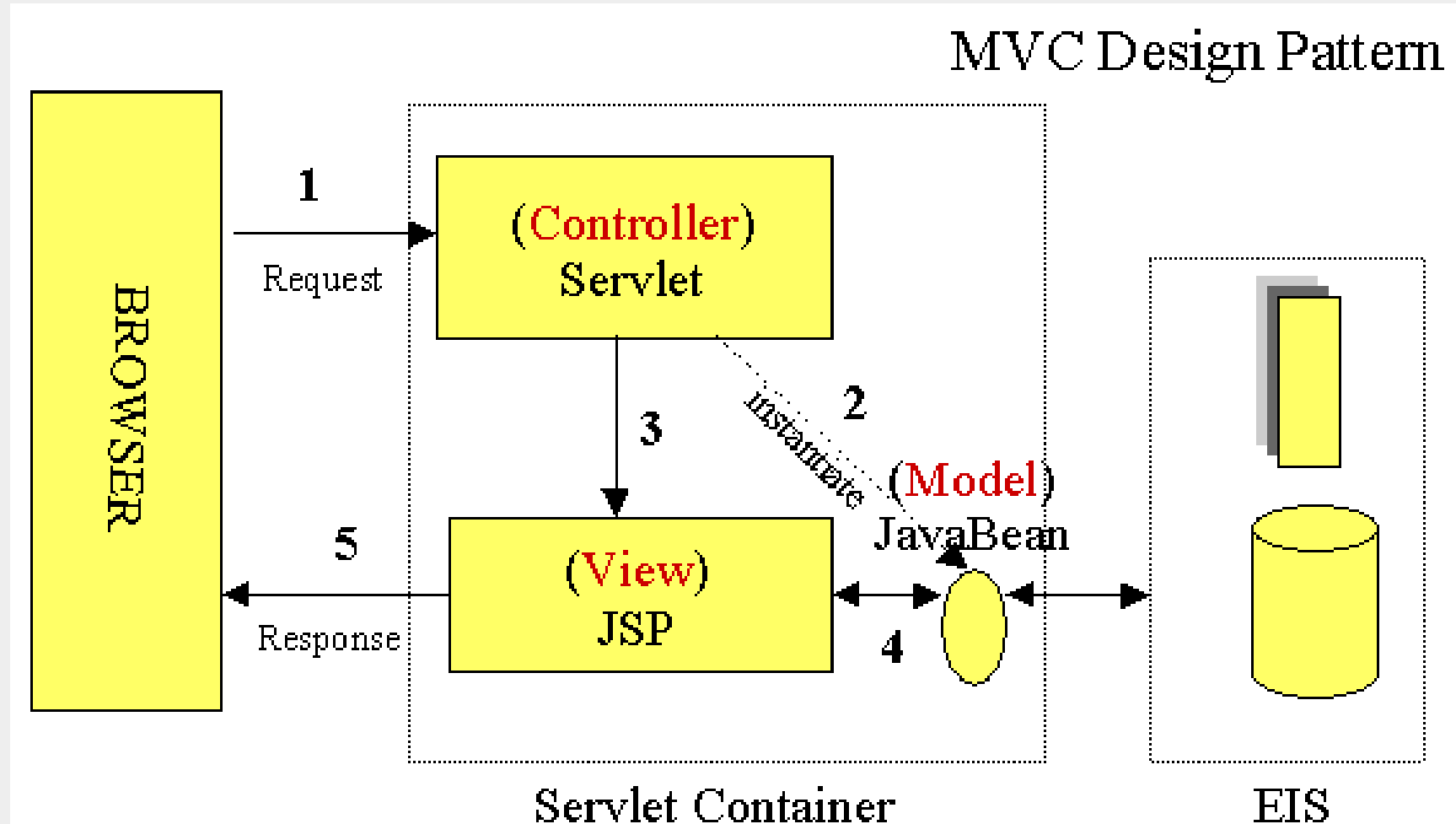
Understanding the model 1 architecture

- In the Model 1 architecture, an incoming request is sent directly to the JSP, which processes it and replies back to the client.
- There is still separation of presentation from content, because all data access is performed using beans.
- Suitable for simple applications, may not be desirable for complex implementations.

Understanding the model 1 architecture....

- Leads to a significant amount of Java code embedded within the JSP page, especially if there is a substantial request processing to be performed.
- This is certainly an issue if your JSP pages are created and maintained by designers -- which is usually the norm on large projects.
- Another downside of this architecture is that each of the JSP pages must be individually responsible for managing application state and verifying authentication and security.

Model 2 architecture (MVC)




Understanding the model 2 architecture

- This is a server-side implementation of the popular Model – View - Controller design pattern.
- The processing is divided between presentation and front components.
- Presentation components are JSP pages that generate the HTML/XML response that determines the user interface.
- Front components (also known as controllers) do not handle any presentation issues, but rather, process all the HTTP requests. They are responsible for creating any beans or objects used by the presentation components, as well as deciding, which presentation component to forward the request to. Front components can be implemented as either a servlet or JSP page.

Understanding the model 2 architecture....

- The advantage of this architecture is that there is no processing logic within the presentation component itself; it is simply responsible for retrieving any objects or beans that may have been previously created by the controller, and extracting the dynamic content within for insertion within its static templates.
- This clean separation of presentation from content leads to a clear delineation of the roles and responsibilities of the developers and page designers on the programming team.
- The front components present a single point of entry into the application, thus making the management of application state, security, and presentation uniform and easier to maintain.

Indicative index of complexity

<p>Simple application or small development team</p>  <p>Complex application or large development team</p>	Call Java code directly. Place all Java code in JSP page. Appropriate for very small amounts of code.
	Call Java code indirectly. Develop separate utility classes. Insert into JSP only the Java code needed to invoke the utility classes.
	Use beans. Develop separate utility classes structured as beans. Use <code>jsp:useBean</code> , <code>jsp:setProperty</code> and <code>jsp:getProperty</code> .
	Use the MVC architecture. Have a servlet response to the original request, look up data and store results in beans. Forward to a JSP page to present results.
	Use EL. Use shorthand syntax to access and output object properties. Used in conjunction with beans and MVC.
	Use custom tags. Develop tag handler classes. Invoke them with XML-like custom tags.

Benefits of MVC

- You can distribute development effort to some extent, so that changes in one part of the web application do not require changes to another. The developers responsible for writing the business logic can work independently of the developers responsible for the flow of control, and web-page designers.
- The MVC design has an organizational structure that better supports scalability (building bigger applications) and ease of modification and maintenance (due to the cleaner separation of tasks).
- You can maintain an environment that comprises different technologies across different locations.

The front controller

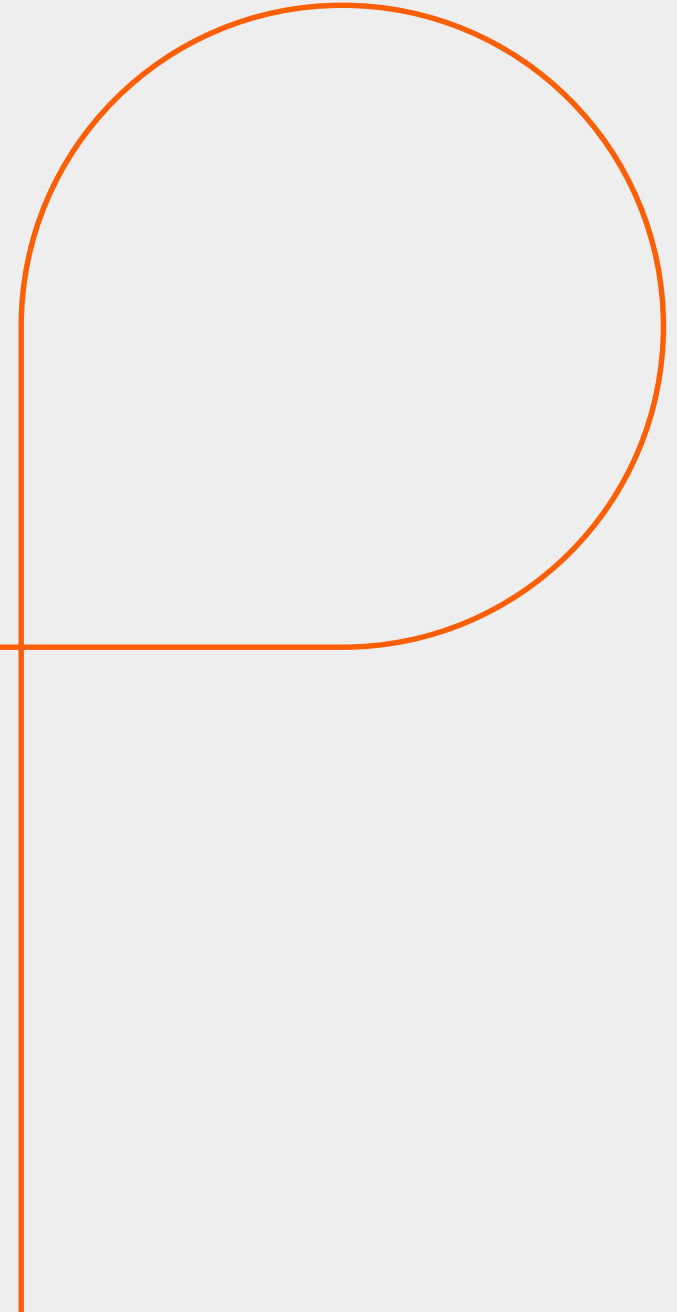
- Front Controller is a core JavaEE design pattern
- Under this pattern a single component, usually a servlet, servlet filter or possibly a JSP, acts as the single control point for the presentation tier of a web application.
- All of the app's requests go through a single controller, which handles dispatching the request to the appropriate places

Summary:

- With this we have come to an end of our session, where we discussed :
 - Separation of concerns
 - Different JSP-based application models
 - Model I
 - Model II
 - The front controller

Appendix

Thank You





Thank you!

Persistent University

