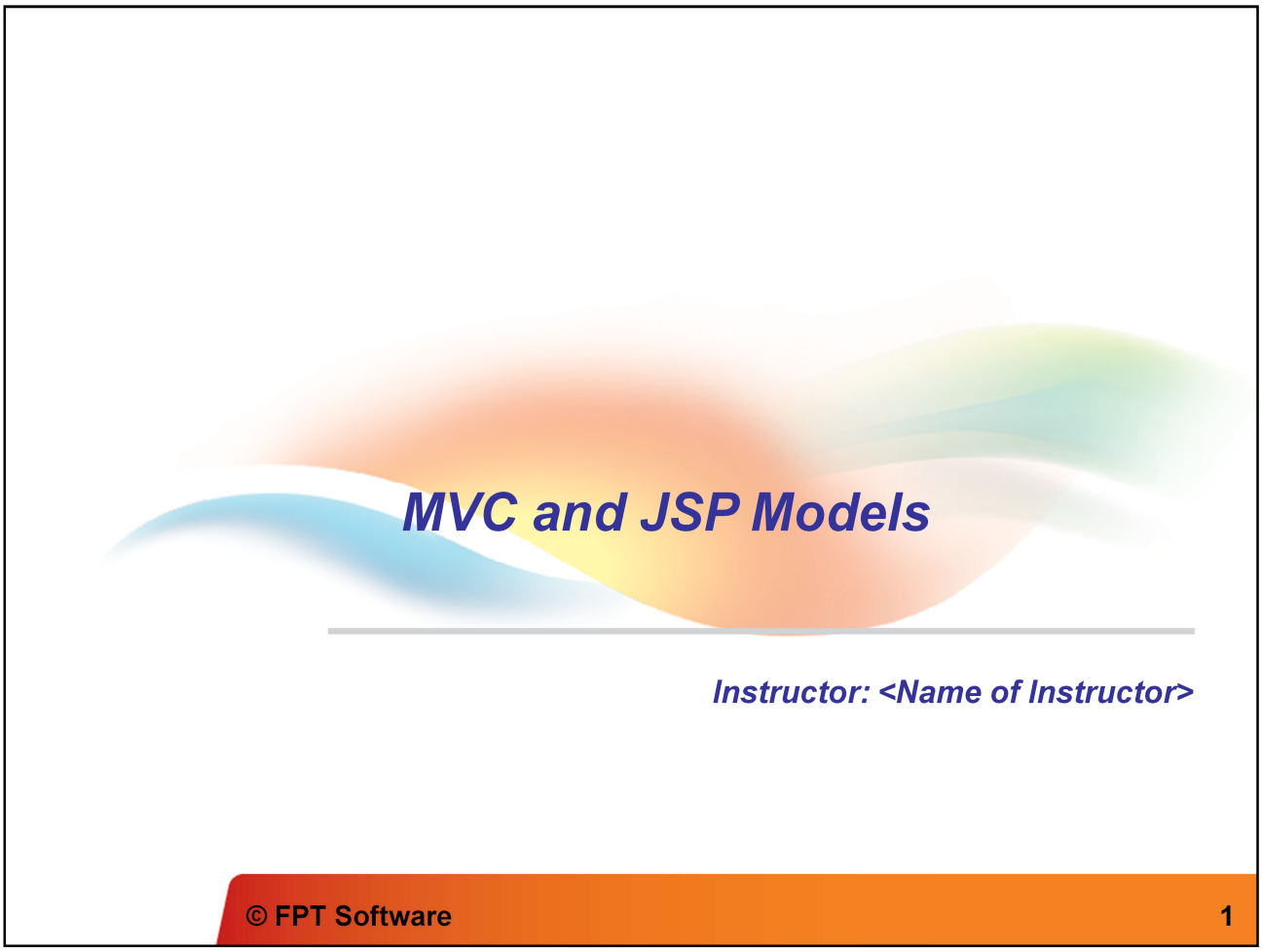
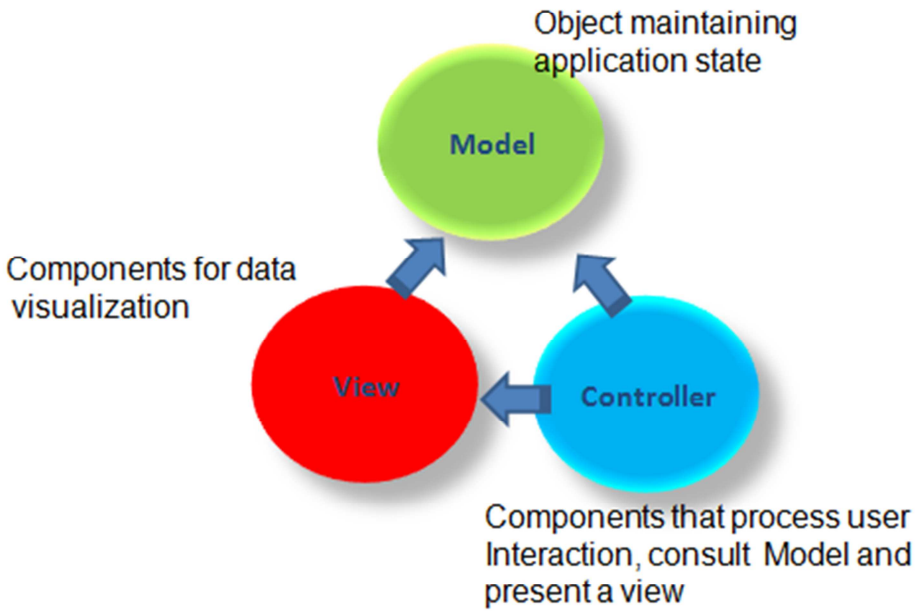



Latest updated by: HanhTT1

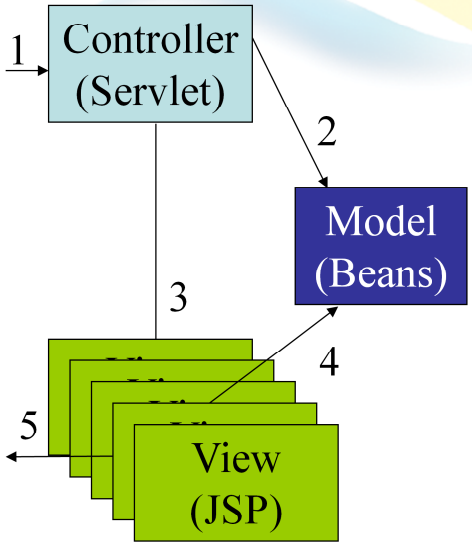
## What Is a MVC?

- MVC stands for Model / View / Controller.
  - A software pattern where logic is separated from the model and view in order to provide for better reuse possibilities.
  - A software pattern recognized in the early days of small talk.

## MVC Architecture



## Web Application MVC Pattern




Model
- Information is provided in objects or beans

View
- The JSP provide the view

Controller
- Servlet provides control logic and becomes the controller



StudentListController
- doGet

StudentModal
- List<Student> students;

BusinessLogic
- StudentBO
    - listOfStudent
- StudentDAO

StudentListView.jsp
- Modal

---

StudentAddController
- doGet

## MVC - Model

- Models data and behavior behind business process
- Manages Information - If Changes
- Contains data and Related Functionality
- Maps Real-World Entities
- Performing DB Queries
- Calculating Business Process
- Encapsulates Domain Logic which are independent of Presentation

## MVC - Controller

- Serves logical connection between user's interaction and the business process
- It receives and Translates input to request on model or view
- Input from user and instructs the model and view to perform action
- Responsible for making decision among multiple presentation
- Maps the end-user action to the application response

## MVC - View

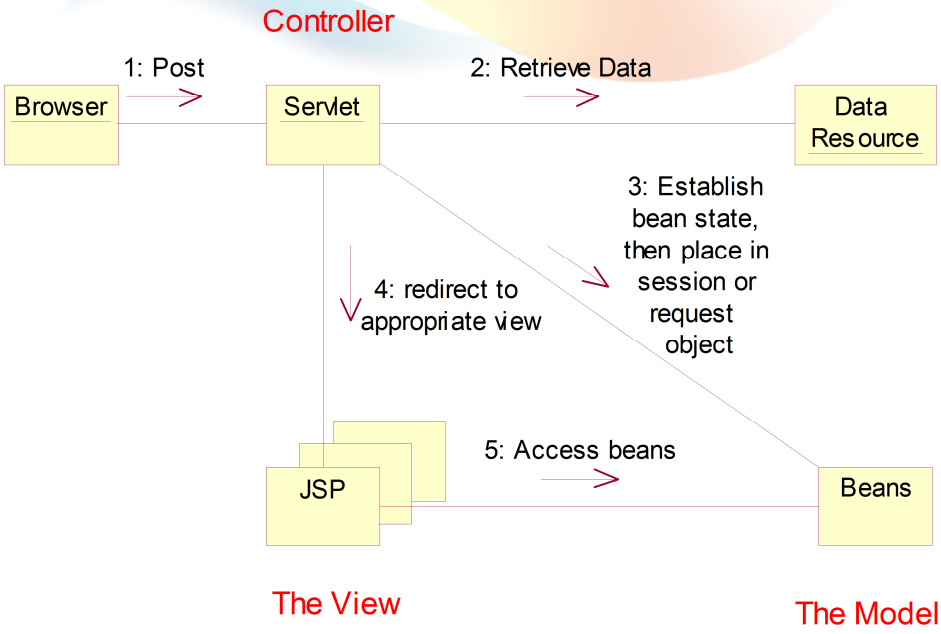
- Obtains data from model & presents to the user
- Represents Output/Input of the application
- Display results of Business Logic
- Free Access to Model
- Reads Data from Model – Using Query Methods

## Relationship between Components

- View and Controller
  - Controller is responsible for creating or selecting view
- Model and Controller
  - Controller depends on model
  - If a change is made to the model then there might be required to make parallel changes in the Controller
- Model and View
  - View depends on Model
  - If a change is made to the model then there might be required to make parallel changes in the view

# Logical Layers in Web Application

# MVC Collaboration Diagram



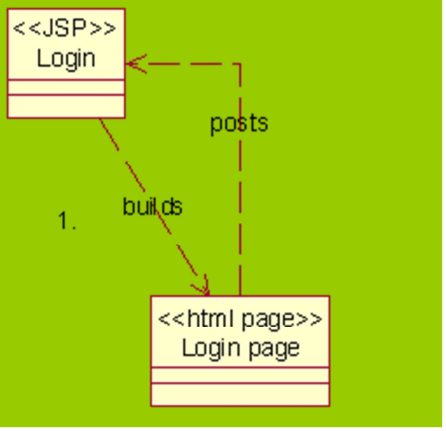
## Java 2 Web Applications

- Java 2 web application options:
  - Servlets
    - + Great for Java people
    - - Difficult to manage graphical changes in HTML layout.
  - JSP
    - + Great for web developers
    - - Seductive tendency to write logic in the JSP page.

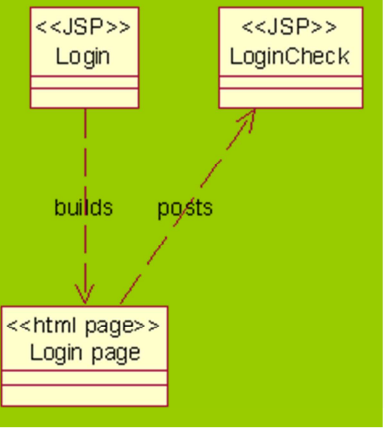
## JSP Model 1

- Web applications where JSP pages are used for every aspect of the development.

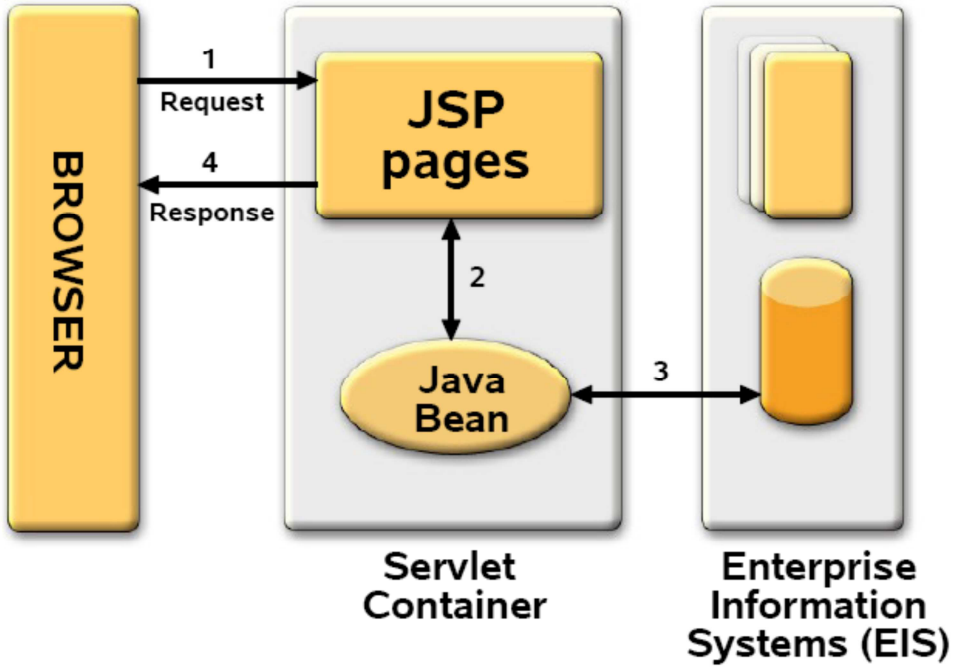
**Option 1**




**Option 2**

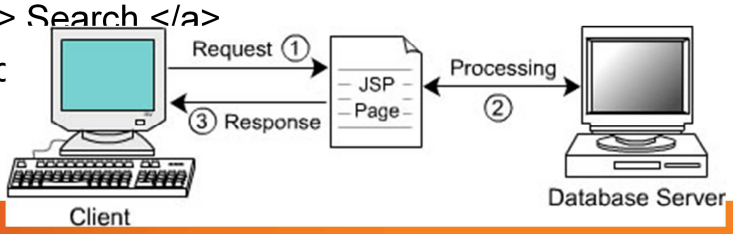


## JSP Model 1 (cont.)

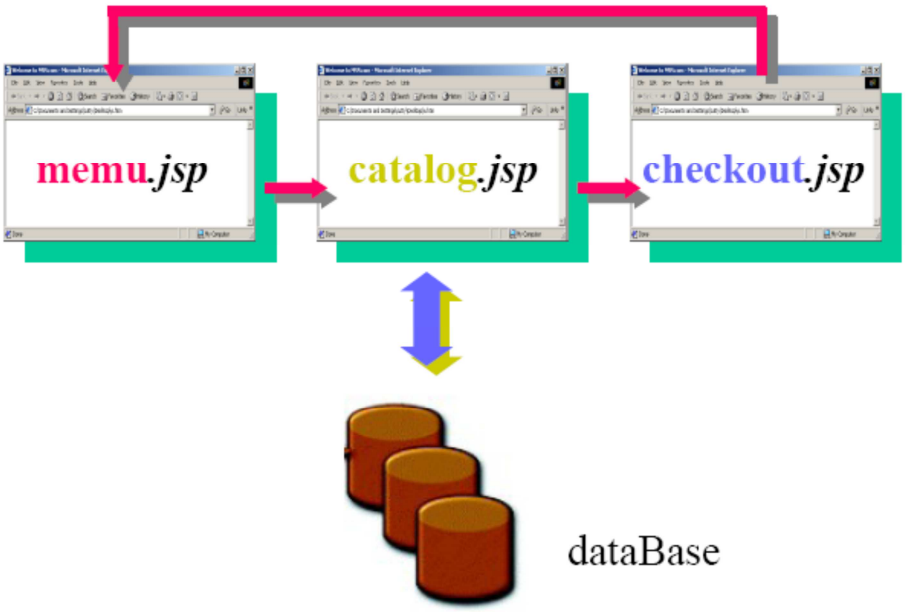


## JSP Model 1 (cont.)

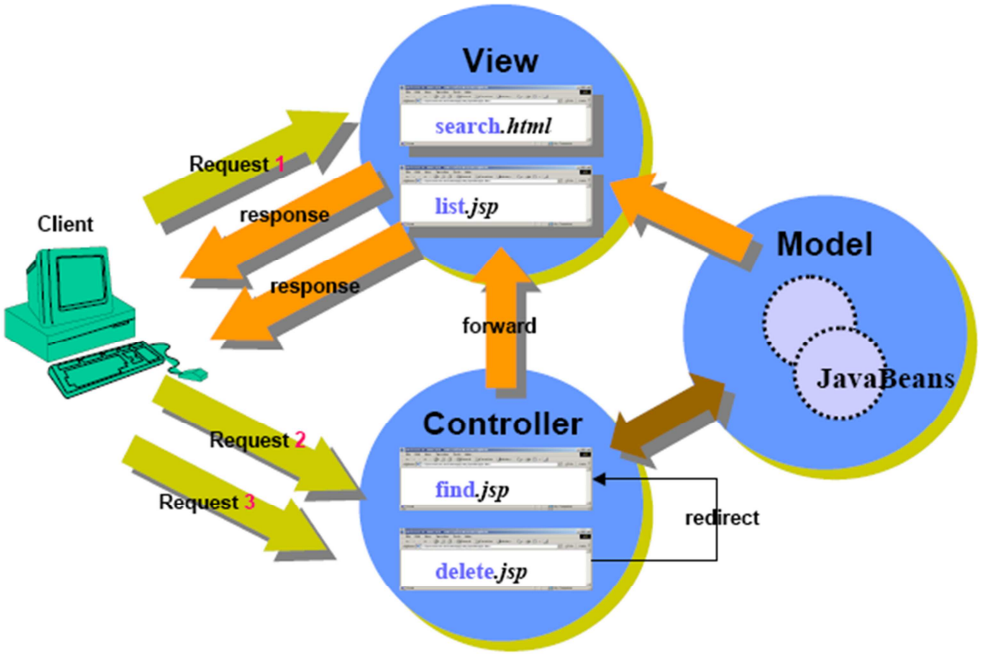
- Represents a page-centric design
  - Composed of a series of interrelated JSP pages
  - Client request is directly processed by JSP page: handle all aspects of the application like presentation, control, and business logics
  - JSP page accesses the DB through JavaBeans to generate a response
- Developed using scripting elements, custom tags, and a scripting language; Next page selection is determined by hyperlink or action of submitting a form
  - <a href="find.jsp"> Search </a>
  - <form action="finc



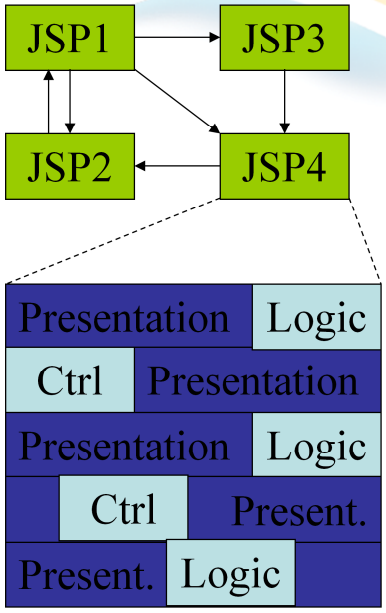
## JSP Model 1 (cont.)



# JSP Model 1 (cont.)



# JSP Model 1 Observation



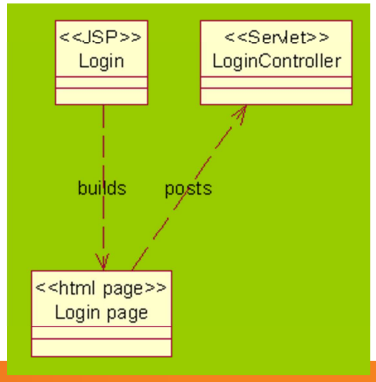

- The Good
  - Easiest Solution
- The Bad
  - Presentation and Logic are mixed.
- The Ugly
  - No reuse possibilities

## JSP Model 1 Observation (cont.)

- Advantages
  - Lightweight design – for small, static application
  - Suitable for small applications having very simple page flow, little need for centralized security control and logging
  - Separation of presentation from content
- Limitations
  - Navigation Problem – to change name of JSP file have to change in many location
  - Applications are difficult to modify – large Java code being embedded in JSP page
  - Not suitable for large and complex applications

## JSP Model 2

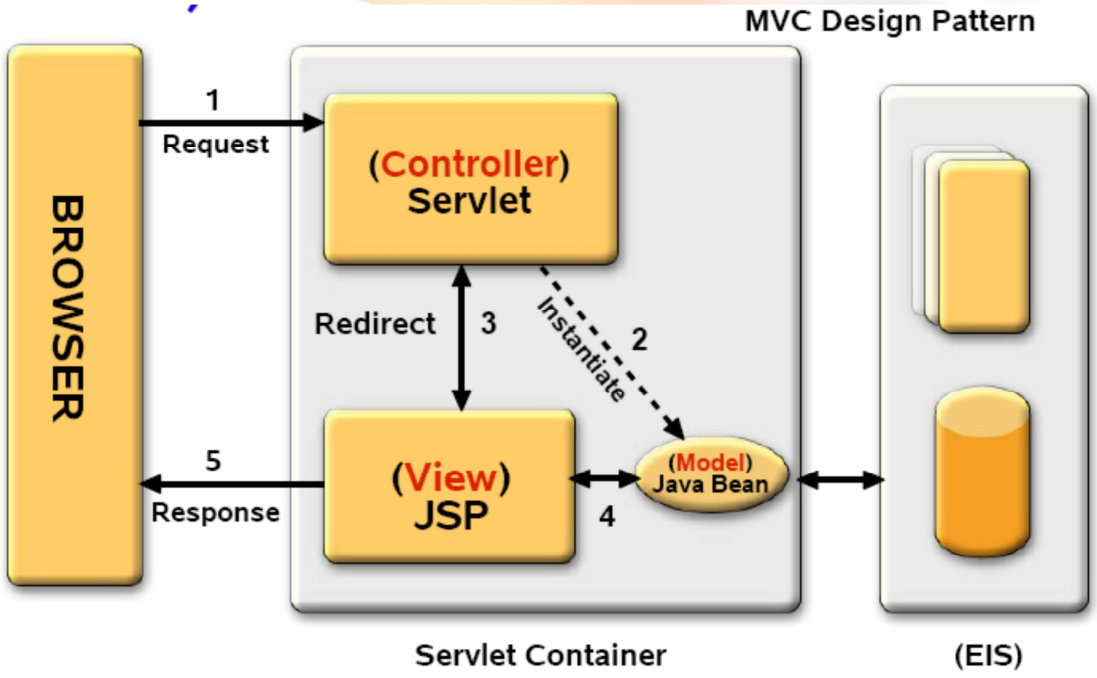
- Web applications where JSP pages are used for the GUI aspect of the web development and the logic of the application is placed in the servlets it posts to.






This provides the base for a model view controller architecture.

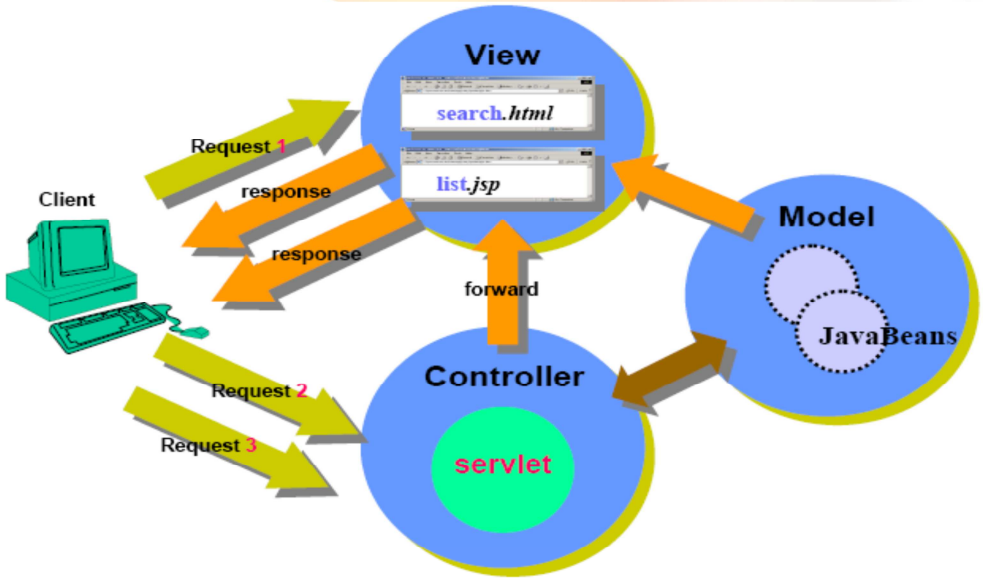
## JSP Model 2 (cont.)



## JSP Model 2 (cont.)

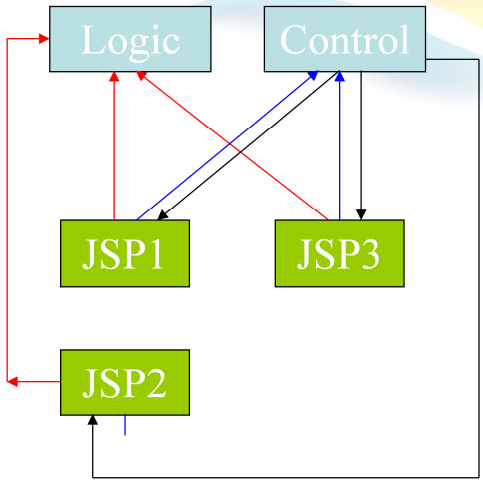
- Represents a controller design
- Based on Model-View-Controller (MVC) pattern
- A combination of servlets, JSP and Java Beans
  - JSP pages are used only for presentation
  - Servlet handles initial request, partially process the data, set up beans, then forward the results to one of a number of different JSP pages
- Servlet serves as a gatekeeper
  - Provides common services, such as authentication, authorization, login, error handling, and etc
- Servlet serves as a central controller
  - Act as a state machine or an event dispatcher to decide upon the appropriate logic to handle the request

# JSP Model 2 (cont.)




Servlet-centric Scenario

# JSP Model 2 – Observation




- The Good
  - Reuse opportunities
    - Other application may be able to use the same code.
- The Bad
  - There is no longer a one to one mapping from a view to a single source of code.
- The Ugly
  - Takes more forethought.

## JSP Model 2 - Observation (cont.)

- Advantages
  - Easier to build, maintain and extend: Suitable for large and complex applications
  - Single point of control (Servlet) for security and logging
- Limitations
  - Increase Design Complexity