HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG

BÀI GIẢNG MÔN

Lập trình Web

Giảng viên: Nguyễn Trọng Khánh

Điện thoại/E-mail: khanhnt@ptit.edu.vn

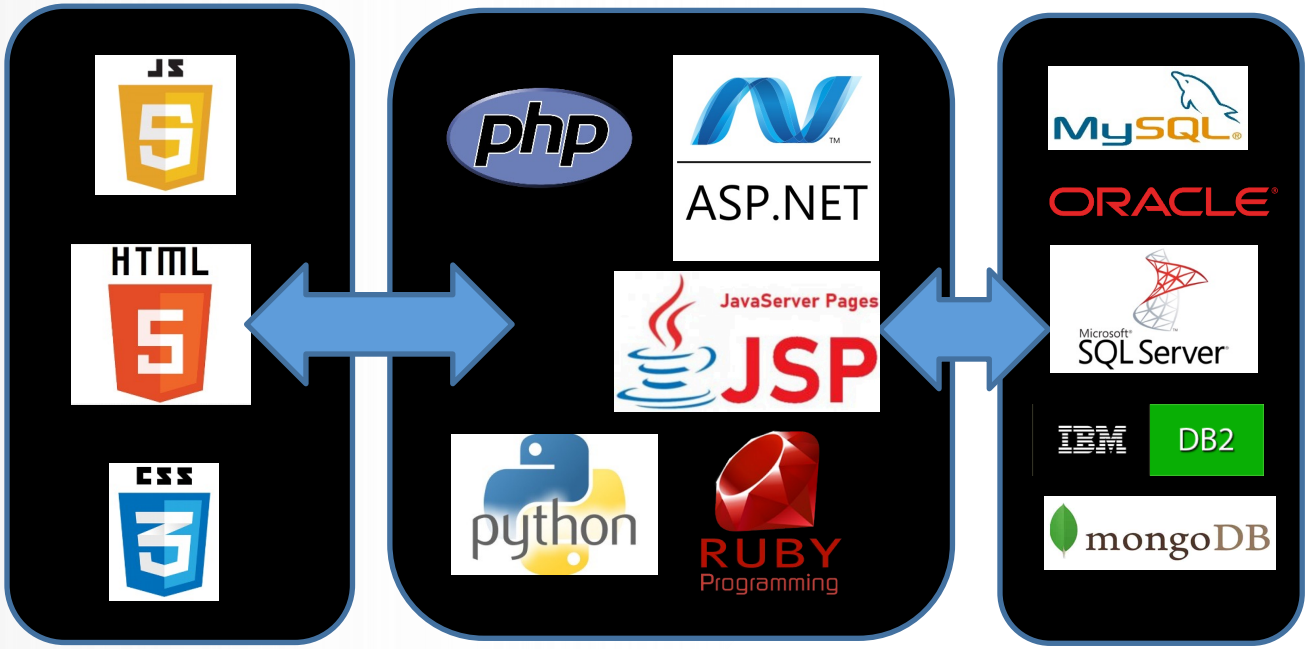
Bộ môn: HTTT - Khoa CNTT1

Học kỳ/Năm biên soạn: 1/2024

Backend Development

Some materials in this slide were inspired by the course CS4640 – The university of Virginia

Web DEVELOPMENT Knowledge



3

History of Backend development

- **Web backend**: "server-side" of a website, handling business logic, database interactions, and server integrations.
- Evolution from static to dynamic web applications.
- Early Days & CGI
  - Early 1990s: Introduction of the World Wide Web with static HTML pages.
  - Mid-1990s: CGI (Common Gateway Interface)
    - First step towards dynamic content generation.
    - Allowed server to interact with other programs to generate dynamic content.

History of Backend development

- Programming Languages & Frameworks
  - Late 1990s to Early 2000s: PHP, Perl, and ASP
    - Simplified creation of dynamic web pages.
  - Java & .NET
    - Gained popularity for enterprise applications.
    - Introduced technologies like Servlets, JSP (Java), and .NET Framework (Microsoft).
  - RESTful APIs & Single Page Applications
  - Mid to Late 2000s: RESTful APIs
    - Enabled more flexible backend development and integration.
  - SPAs (Single Page Applications)
    - Improved user experience with seamless interactions.
    - Backend logic accessed through RESTful APIs.

History of Backend development

- Microservices, Containerization, & Cloud-Native
- Early 2010s: Microservices Architecture
  - Independent development and deployment of application components.
- Containerization (Docker, Kubernetes)
  - Enhanced deployment and management of complex applications.
- Cloud Native & Serverless Computing
  - Shift towards cloud as a platform.
  - Serverless models reduce infrastructure management tasks for developers.

Tools & Environments

- **XAMPP - A Comprehensive PHP Development Environment**
  - **What is XAMPP?** A software package including Apache, MySQL, PHP, Perl.
  - **Application:** Ideal for developing PHP web applications locally.
  - **Advantages:** Easy to install and configure, supports multiple operating systems (Windows, Linux, macOS).
- **WAMP** (Windows, Apache, MySQL, PHP/Perl/Python)
- **LAMP** (Linux, Apache, MySQL, PHP/Perl/Python)
- **MAMP** (Mac, Apache, MySQL, PHP/Perl/Python)

Tools & Environments

- **MEAN/MERN Stack - Modern Full-stack JavaScript Development**
  - **MEAN Stack:** MongoDB, Express.js, AngularJS, and Node.js, entirely in JavaScript.
  - **MERN Stack:** A variation of MEAN – AngularJS → ReactJS

Tools & Environments

- **Apache Tomcat & Java Development Tools**
  - **Apache Tomcat:** An environment for running Java web applications, supporting Servlet and JSP.
  - **Eclipse and IntelliJ IDEA:** Powerful IDEs for Java development, including Tomcat integration, debugging, and project management.
  - **Spring Framework:** A robust framework for web application development, supporting MVC, RESTful services.

How User Interact With Web Apps

https://it.ptit.edu.vn/

Usability

Web client

HTML,JS, CSS

HTTP Request

HTTP Response

Web application

Request

Response

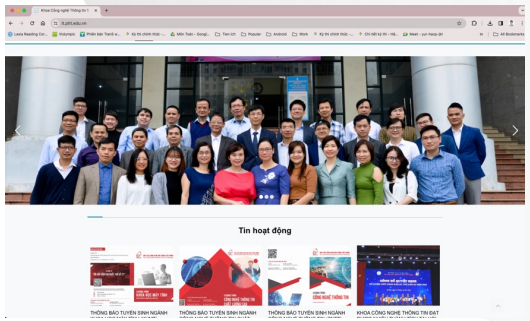
Web component$_1$

Web component$_2$

Web component$_n$

DB or persistent data

Web server



Web Development (General View)

Front End Development

- UI/UX design, usability
- Web design
- Responsive design

- Client side
- HTML, CSS, JS (and JS libraries, frameworks)
- Fixed huge amount of data, no database interaction

Back End Development

- Server side
- Speed, performance, scalability, security, availability, accessibility, reliability
- Business logic
- Java, PHP, Python, Ruby on Rails, and back end frameworks
- Database interaction

- Server administration
- Database, data science

Full-Stack Development

- Variety of skills (both front end and back end)
- Horizontal technology development (+)
- Not expert in particular skill (-)

How the Web Works

PHP: Form Handling

Name:
Awesome Dev
Email:
awesome@virginia.edu
Comment:
PHP is fun!
Submit

Browser

```
<form action="form-handler.php" method="post">
  <label>Name: </label>
  <input type="text" name="name" autofocus /> <br/>
  <label>Email:</label>
  <input type="email" name="emailaddr" /> <br/>
  <label>Comment: </label>
  <textarea rows="5" cols="40" name="comment"></textarea> <br/>

  <input type="submit" value="Submit" />
</form>
```

To server

```
<?php
if ($_SERVER['REQUEST_METHOD'] == 'POST')
{
    # param => value
    $name = $_POST['name'];
    $email = $_POST['emailaddr'];
    $comment = $_POST['comment'];

    echo "Thanks for this comment, $name <br/>";
    echo "<i>$comment</i> <br/>";
    echo "We will reply to $email <br/>";
}
?>
```

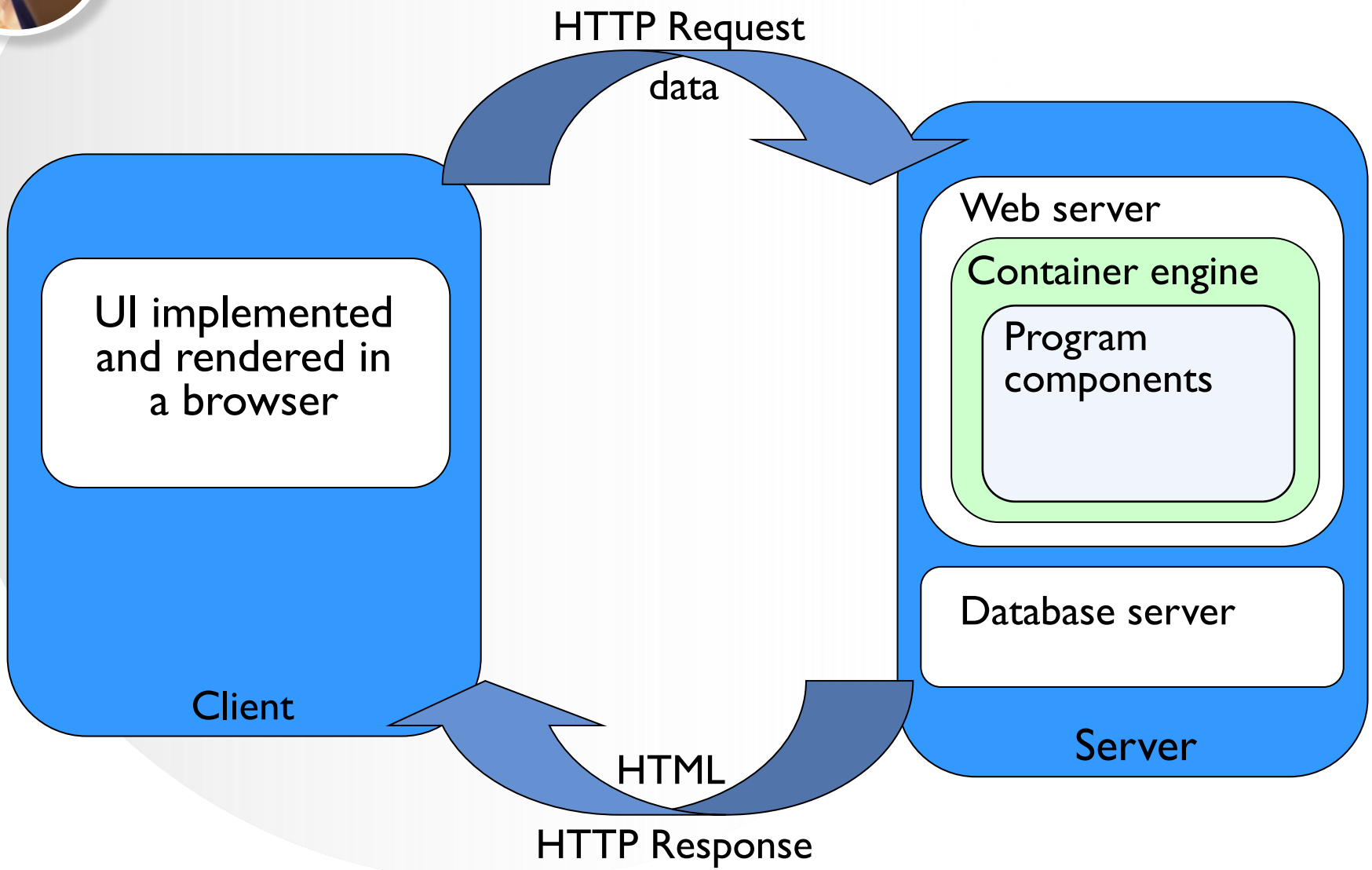
form-handler.php

To client

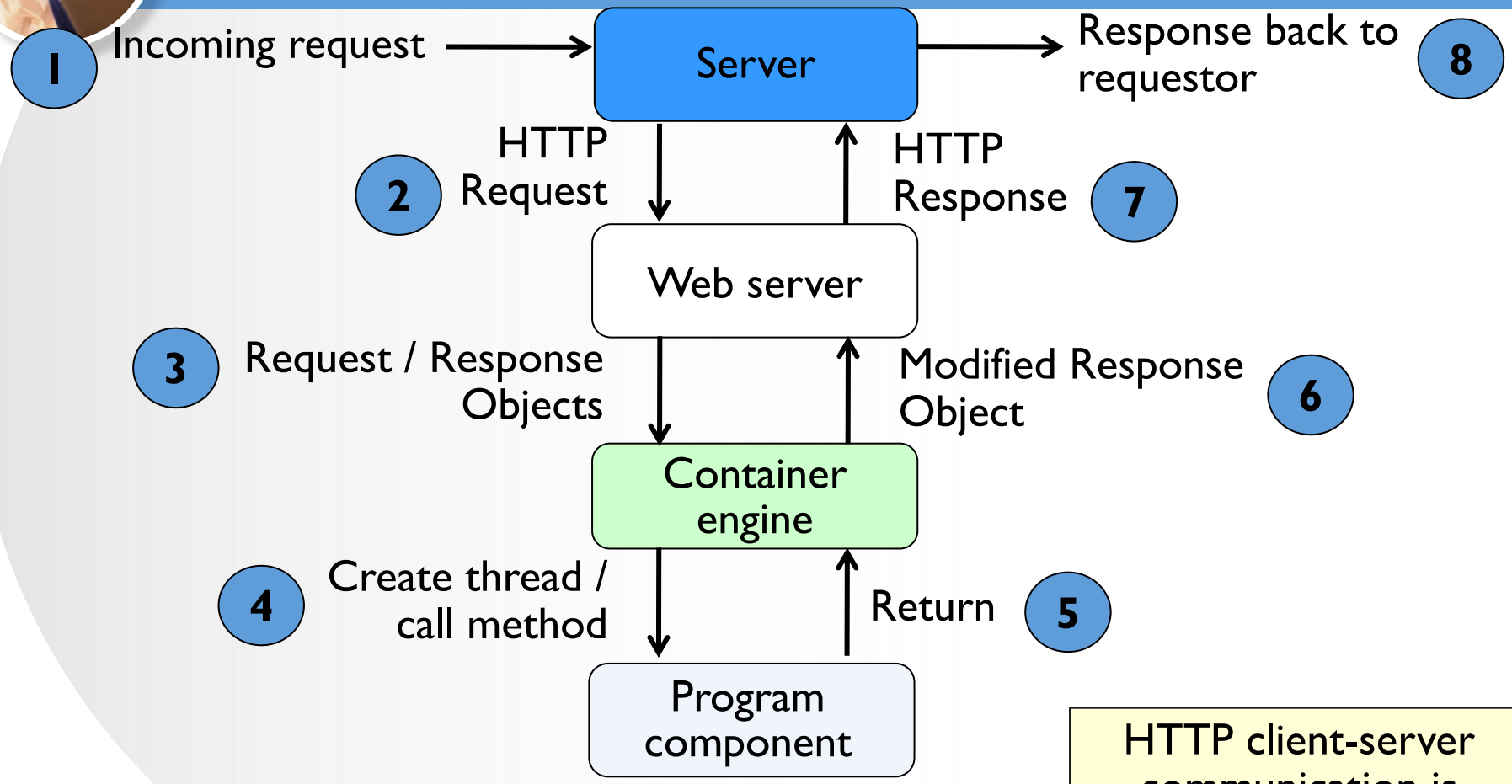
Thanks for this comment, Awesome Dev
PHP is fun!
We will reply to awesome@virginia.edu

Browser

Server Side Processing



Execution Overview



As soon as the request is made and fulfilled, the connection is terminated

HTTP client-server communication is connectionless (stateless)

Session Management

How can servers keep track of state of different clients?

1. *Session* : A single coherent use of the system by the same user
   - Example : shopping carts
2. *Cookies* : A string of characters that a web server places on a browser's client to keep track of a session
   - Usually used as an index into a table (*dictionary*) on the server
   - Most dictionaries expire after a period of time (15 to 30 minutes)

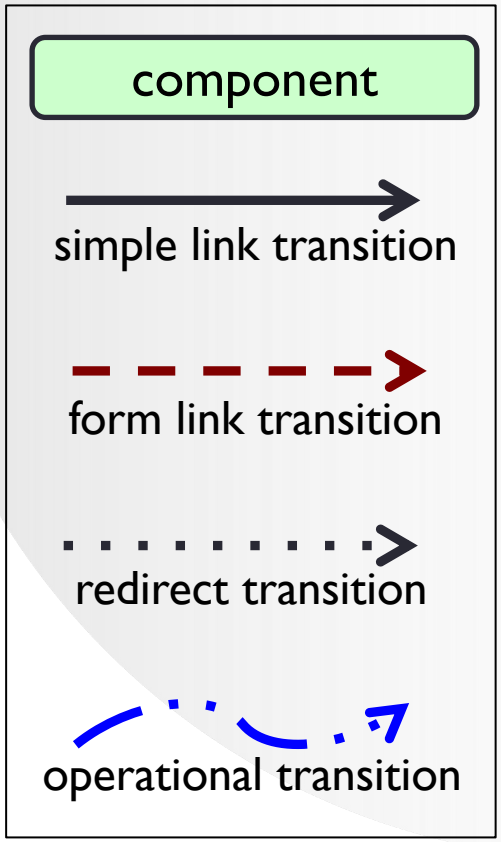
Additional mechanisms

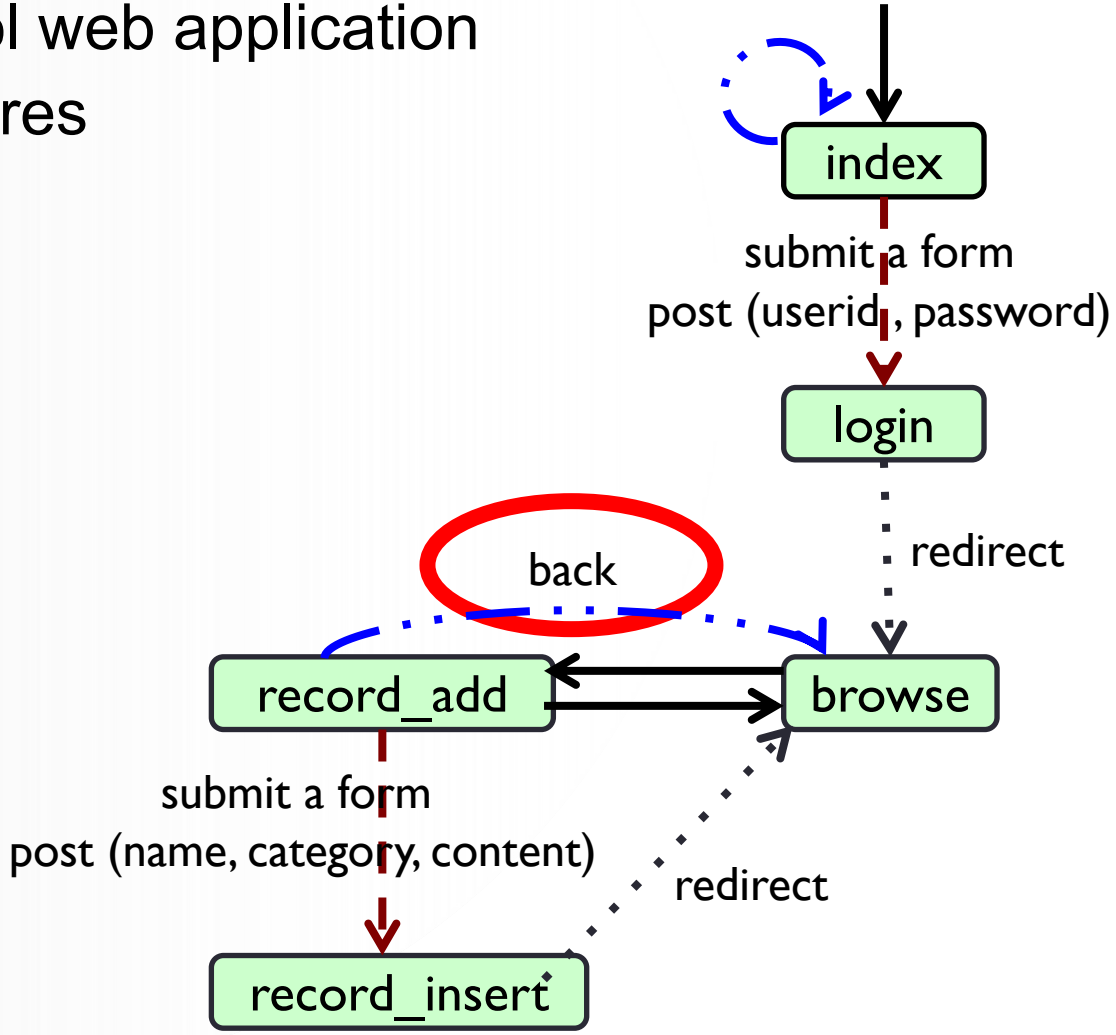
- URL rewriting
- Hidden form control

We will come back to this later …

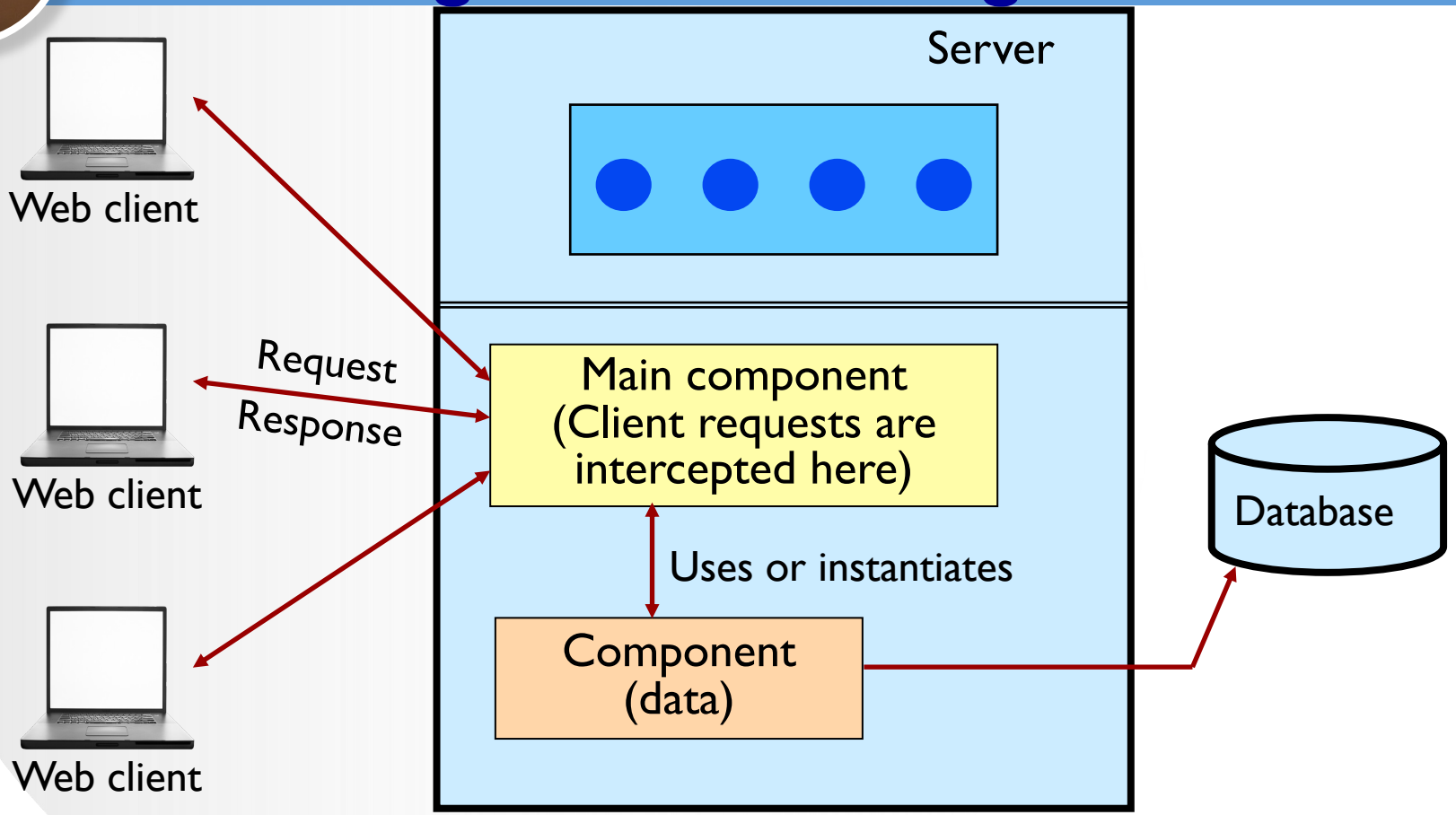
Additional Web Features

User's ability to control web application via web browser features



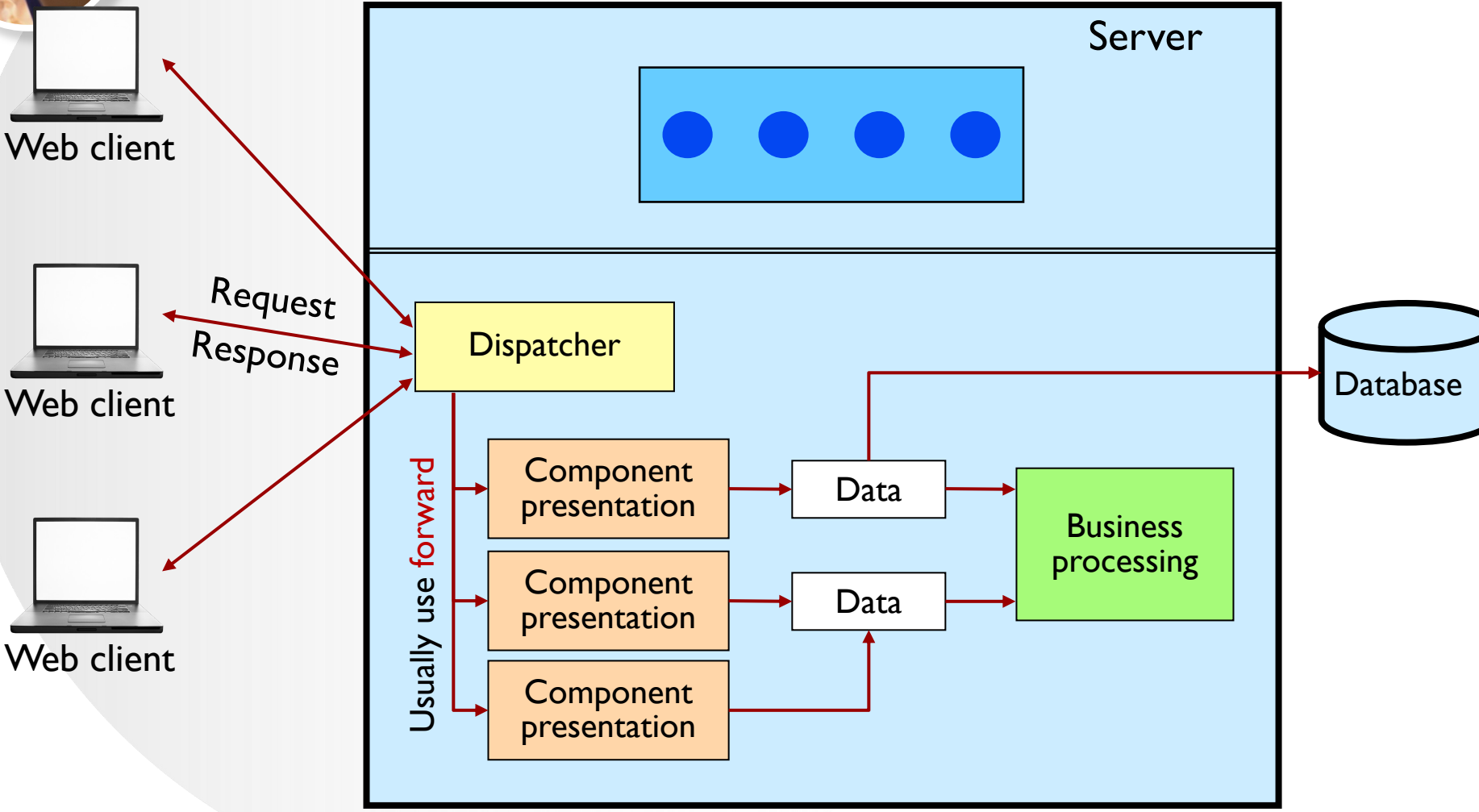


Back End Component and Page-centric Design



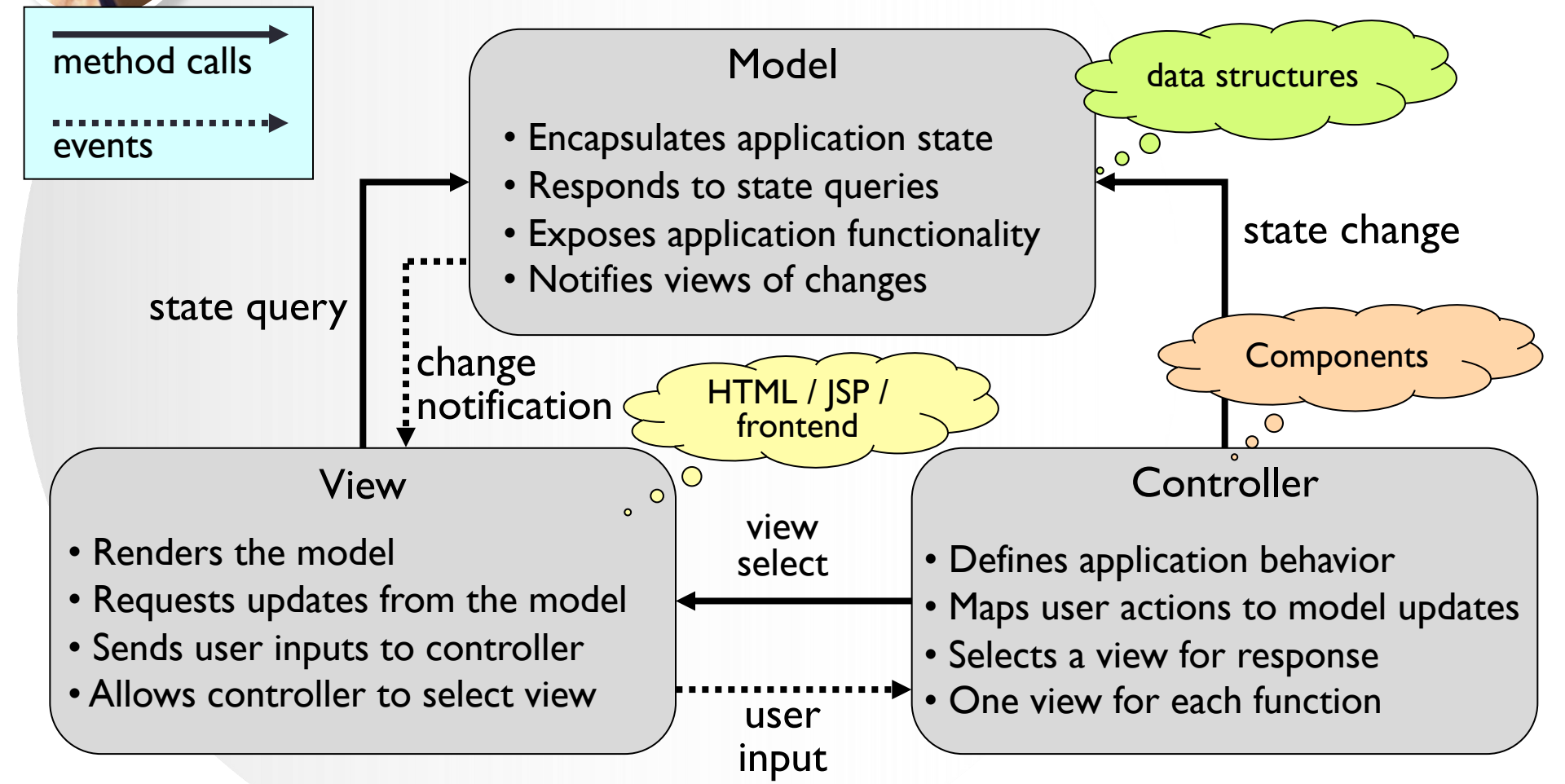
Requests are made to a main component and the main component response to clients

Back End Component and Dispatcher (N-tier) Design



Requests are sent to a dispatcher that then forward the requests to another component (using *forward* or *redirect* control connection)

Back End Component and Model View Controller



[Graphic from Designing Enterprise Applications with the Java 2 Platform, Enterprise Edition, Nicholas Kassem et al., October 2000]