Java DDWA

Lindsay Nickalo bradleyl4@winthrop.edu

1. HTTP

1. What is HTTP?

   Stands for **Hypertext Transfer Protocol**. It's the set of rules that specify how the web browser sends a request to the web server and how the web server sends the response back.
   From a high-level, HTTP is what is known as a Request/Response protocol. The client (usually a browser) sends a Request to the web server. The web server receives the Request, processes it, creates a Response, and sends the Response back to the client.

Request

Here's an HTTP request for the Hacker News homepage from a web browser:

```
GET / HTTP/1.1
Host: news.ycombinator.com
Connection: keep-alive
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14_0) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/7
0.0.3538.102 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.9
Cookie: __cfduid=dfbe07cb1c609155c811135d7085155561538157611; __unam=3267369-16621568c13-799b9d7d-3
```

Figure 1: Example of an HTTP Request

2. What are the four common HTTP methods?

   GET: a request asking for data from the server
   POST: a request sending data to the server for it to create something
   PUT: a request sending data to the server to update existing data
   DELETE: a request to delete data in the server

3. What HTTP method is usually associated with pressing a button on a form?

POST method is most commonly associated with forms since most forms are used for creating things.

4. What HTTP method is usually associated with clicking a link?

   GET method is usually associated with clicking a link since links are used to access other pages and data that must be retrieved from the server for rendering.

5. What is a URL?

   Short for **Uniform Resource Locator**, it is the location of a web resource on a server and the mechanism for retrieving it. The format consists of the **protocol**, the **hostname**, and the **path** to the file. For example:

   http://www.example.com/departments/humanresources.html

   Here, 'http://' is the protocol, 'www.example.com' is the hostname, and the rest is the path.

6. What are the three main parts of the HTTP response?

   The **HTTP Status Code**, the **headers**, and the **content**. The HTTP Status Code denotes the result of the request (success, failure, not found, etc.). The headers denote the information being sent back necessary for rendering and record-keeping, and the content can be anything (html page, css, etc.). For example:

```
HTTP/1.1 200 OK
Server: nginx
Date: Sun, 18 Nov 2018 20:23:00 GMT
Content-Type: text/html; charset=utf-8
Transfer-Encoding: chunked
Connection: keep-alive
Vary: Accept-Encoding
Cache-Control: private; max-age=0
X-Frame-Options: DENY
X-Content-Type-Options: nosniff
X-XSS-Protection: 1; mode=block
Referrer-Policy: origin
Strict-Transport-Security: max-age=31556900
Content-Security-Policy: default-src 'self'; script-src 'self' 'unsafe-inline' https://www.google.com/recapt
cha/ https://www.gstatic.com/recaptcha/ https://cdnjs.cloudflare.com/; frame-src 'self' https://www.google.c
om/recaptcha/; style-src 'self' 'unsafe-inline'
Content-Encoding: gzip

<html op="news">
    <head>
        <title>Hacker News</title>
    ... lots more HTML here.
</html>
```

Figure 2: Example of an HTTP Response

7. What is the HTTP response status code for success?

   200 - 'OK'

8. What is the HTTP response status code for a page that is not found?

   404 - 'Not found'

9. What is the HTTP response status code for an internal service error?

   500 - 'Internal Server Error'

10. What is the HTTP response status code for forbidden access?

    403 - 'Forbidden'

11. What is the HTTP response status code for redirect?

    302 - 'Found'

12. What are URL parameters?

    These are parameters that are set dynamically in the page's URL and can be accessed both client and server side.

13. How are URL parameters specified?

    They are added to the URL as a key-value pair, denoted with a question mark. For example:

    http://www.example.com/employee?id=5

    The URL parameter is 'id' and it is passed with a value of 5.

14. What is REST?

    REST, or **Representational State Transfer**, is a software architecture for web services. It uses HTTP to fetch data and execute actions. It's not a protocol and, to be honest, not even a true standard. Roy Fielding defined REST in his Ph.D. thesis. The thesis outlines several principles for REST architecture including statelessness (REST shouldn't rely on what just happened to decide what to do next) and cacheability (if we ask for the same data twice, the server may suggest we use what we already have). The principles are a bit academic. They serve more as guidance than formal rules.
    A REST resource can be any logical chunk of data: customers, shipping containers, chess moves... anything. A REST API exposes some set of **CRUD (create, read, update, and delete) operations** that act on each resource. It's possible to expose non-CRUD operations – maybe an action kicks off a background process, so there's nothing limiting REST to CRUD. But generally, each resource exposes one or more of the CRUD operations.
    A REST request is simply an HTTP request. It uses one of the five standard HTTP request methods to determine which CRUD method to activate:

    - **GET** → **Read** operation (single or multiple entities)
    - **POST** → **Create** operation, includes data to be added to some form of a data store
    - **PUT**, **PATCH** → **Update** operation, can be a full update or partial update
    - **DELETE** → **Delete** operation (single or multiple entities)

    REST doesn't specify the serialization format to be used for fetching and submitting data. **Both XML and JSON are common.** JSON's advantages include a smaller payload and easy conversion to JavaScript. That's why we introduce JSON above.

    Some servers allow the client to choose a serialization format using the Accept HTTP header. The value application/xml returns XML and application/json returns JSON.

It's also important to tell the server which format is used in the request. We specify using the Content-Type HTTP header. The value application/x-www-form-urlencoded tells the server to expect standard HTML form data, application/json indicates a JSON payload, application/xml is not commonly used in the HTTP request.

2. JavaScript

1. What is a function?

   A JavaScript function is a set of statements that performs a task or calculates a value. To use a function, you must define it somewhere in the scope from which you wish to call it.

2. What are the main data types of JavaScript?

Data Type	Category	Description
String	primitive	used to represent textual data, created with single or double quotes
Number	primitive	used to represent positive or negative numbers with or without decimal place, or numbers written using exponential notation
Boolean	primitive	can hold only two values: true or false
Object	composite	complex data type that allows you to store collections of data, an object contains properties, defined as a key-value pair. A property key is always a string, but the value can be any data type, like strings, numbers, booleans, or complex data types like arrays, function and other objects.
Array	composite	used for storing multiple values in single variable, each value has an index, and it may contain data of any data type
Function	composite	callable object that executes a block of code

Table 1: JavaScript Data Types

3. What's the difference between '==' and '==='?

   '==' compares two variables, ignoring the data type, whereas '===' will check the data type and if it doesn't match, return false.

4. What is AJAX?

   Stands for **Asynchronous JavaScript and XML**. Clients (usually JavaScript code running in a web browser) use Ajax to exchange data with web services. This approach is used extensively in modern web applications to help give them the feel of a desktop application. This smoother desktop feel is possible because the HTML page does not reload when making an Ajax call. The Ajax call is made, data is returned from the web service, and then JavaScript code dynamically updates only the portions of the page that need to be updated.
   The pieces that make up AJAX are XMLHttpRequest (XHR), JavaScript, HTML and CSS, and JSON or XML.

5. What is XMLHttpRequest?

   XHR is an application programming interface, or API, that allows JavaScript code to exchange data with web services by exchanging requests and responses over HTTP or HTTPS. Although we could make these calls directly in our JavaScript code, there are several libraries available, including jQuery, that wrap XHR with code that makes it more convenient to make Ajax calls.
   XHR was created by Microsoft to support web access to Outlook email and originally appeared in Internet Explorer. Mozilla Firefox added support soon after. The two implementations were quite similar but there were several small differences. In order to smooth the differences and promote interoperability, the Firefox team created the XMLHttpRequest JavaScript API, which became the de facto standard for XHR.

6. What is JSON?

   JSON stands for **JavaScript Object Notation** and it is one of the encoding schemes used to ferry data between web services clients and their clients. JSON is made up of a combination of key/value pairs.

```
{
    "title": "Design Patterns -- Elements of Reusable Object Oriented Software",
    "authors": ["Eric Gamma", "Richard Helm", "Ralph Johnson", "John Vlissides"],
    "publisher": "Addison-Wesley",
    "price": "$45.00"
}
```

Figure 3: Example of JSON

7. What is a callback function?

   Simply put: A callback is a function that is to be executed after another function has finished executing — hence the name 'call back'.

More complexly put: In JavaScript, functions are objects. Because of this, functions can take functions as arguments, and can be returned by other functions. Functions that do this are called higher-order functions. Any function that is passed as an argument is called a callback function.

```
function doHomework(subject, callback) {
  alert(`Starting my ${subject} homework.`);
  callback();
}

function alertFinished(){
  alert('Finished my homework');
}

doHomework('math', alertFinished);
```

Figure 4: Example of Callback Function in JavaScript

8. What is an anonymous function?

   A function that was **declared without any named identifier to refer to it.** As such, an anonymous function is usually not accessible after its initial creation. We typically use these within AJAX calls, events, etc.

```
setTimeout(function() {
  alert('hello');
}, 1000);
```

Figure 5: Example of an Anonymous Function in JavaScript

9. What is jQuery?

   jQuery is one of the most popular, and most stable, open-source JavaScript libraries used for web development today; it is used by over half of the most visited sites on the web. jQuery allows developers to manipulate HTML, respond to browser events, and make Ajax calls in a consistent way across different browser types. This makes web development much easier, allowing the developer to concentrate on application features rather than special code to account for the differences in browsers. The library is also popular because its plug-in architecture makes it easy to extend the library

and add your own features.
jQuery uses events to know when to execute commands. An event is a signal or message that gets sent only after a particular condition is met. We can tell jQuery which of the events we're interested in and then jQuery will let us know when that particular thing has happened.

10. What are some benefits to using JQuery?

    The ready event is specific to jQuery and, as described above, fires after all of the HTML elements have been rendered but possibly before all of the content (for example images) have been loaded. The JavaScript load event fires only after all of the elements have been rendered and all the content has been loaded. We use the jQuery ready event so that our code is available to run at the earliest possible moment.
    jQuery makes it very easy to interact with HTML elements. We can do everything from getting values from a form element to changing the class attributes or CSS styles of an element to showing and hiding - and even adding and removing – elements on the page.
    jQuery can also interact with Browser Events, such as click and hover. This allows us to do stuff in response to user interaction with the browser window.

11. How does Bootstrap use JavaScript?

    Bootstrap uses jQuery for JavaScript plugins (like modals, tooltips, etc).

12. What is AngularJS?

    AngularJS is a JavaScript-based open-source front-end web framework mainly maintained by Google and by a community of individuals and corporations to address many of the challenges encountered in developing single-page applications. It is what HTML would have been, had it been designed for building web-apps.
    AngularJS is a JavaScript framework. It can be added to an HTML page with a ¡script¿ tag. AngularJS extends HTML attributes with Directives, and binds data to HTML with Expressions.

13. What are some benefits to using AngularJS?

    AngularJS lets you extend HTML vocabulary for your application. The resulting environment is extraordinarily expressive, readable, and quick to develop.
    AngularJS is a toolset for building the framework most suited to your application development. It is fully extensible and works well with other libraries. Every feature can be modified or replaced to suit your unique development workflow and feature needs.
    Data-binding is an automatic way of updating the view whenever the model changes,

as well as updating the model whenever the view changes. This is awesome because it eliminates DOM manipulation from the list of things you have to worry about.
Controllers are the behavior behind the DOM elements. AngularJS lets you express the behavior in a clean readable form without the usual boilerplate of updating the DOM, registering callbacks or watching model changes.
AngularJS models are plain old JavaScript objects. This makes your code easy to test, maintain, reuse, and again free from boilerplate.
Directives are a unique and powerful feature available in AngularJS. Directives let you invent new HTML syntax, specific to your application.
We use directives to create reusable components. A component allows you to hide complex DOM structure, CSS, and behavior. This lets you focus either on what the application does or how the application looks separately.
AngularJS's locale aware filters and stemming directives give you building blocks to make your application available in all locales.
A deep link reflects where the user is in the app. This is useful so users can bookmark and email links to locations within the app. Round trip apps get this automatically, but AJAX apps by their nature do not. AngularJS combines the benefits of deep linking with desktop app-like behavior.
AngularJS lets you declare the validation rules of the form without having to write JavaScript code.
AngularJS provides built-in services on top of XHR as well as various other backends using third party libraries. Promises further simplify your code by handling asynchronous return of data.
The dependency injection in AngularJS allows you to declaratively describe how your application is wired. This means that your application needs no main() method which is usually an unmaintainable mess. Dependency injection is also a core to AngularJS. This means that any component which does not fit your needs can easily be replaced.
AngularJS was designed from ground up to be testable. It encourages behavior-view separation, comes pre-bundled with mocks, and takes full advantage of dependency injection. It also comes with end-to-end scenario runner which eliminates test flakiness by understanding the inner workings of AngularJS.

3. Tomcat/Servlets/JSP

1. What is Tomcat?

   It is an open source implementation of the Java Servlet, JavaServer Pages, Java Expression Language and Java WebSocket technologies. An Apache Software Foundation project, Tomcat was first released in 1998, just four years after Java itself. Tomcat started as a reference implementation for the first Java Servlet API and the JSP spec. While it's no longer the reference implementation for either of these technologies, Tomcat remains the most widely used Java server, boasting a well-tested and proven core engine with good extensibility.
   At heart, Tomcat is a servlet and JSP container. A Java servlet encapsulates code

and business logic and defines how requests and responses should be handled in a Java server. JSP is a server-side view rendering technology. As the developer, you write the servlet or JSP page, then let Tomcat handle the routing.
Tomcat also contains the Coyote engine, which is a web server. Thanks to Coyote, it's possible to extend Tomcat to include a variety of Java enterprise specs and capabilities, including the Java Persistence API (JPA). Tomcat also has an extended version, called TomEE, that includes more enterprise features.

2. What is a servlet?

   A servlet is a Java class that is used to extend the capabilities of servers that host applications accessed by means of a request-response programming model.
   Servlets are the Java programs that runs on the Java-enabled web server or application server. They are used to handle the request obtained from the web server, process the request, produce the response, then send response back to the web server.

3. What are 5 benefits of a servlet container?

   Servlet containers, sometimes referred to as servlet engines, execute and manage servlets. The servlet container calls servlet methods and provides services that the servlet needs while executing. This gives us many benefits, including:

   - **Platform independence** → the servlets have everything they need within the container and thus are not dependent on the platform they are ran on
   - **Performance** → servlets outperform other technology and can run on a thread once loaded into memory.
   - **Extendable** → servlets can be extended and polymorphed by other classes as necessary
   - **Safety** → Servlets utilize Java's garbage collection service so memory management problems are non-existent
   - **Security** → everything is server-side so the security from the web-server is naturally inherited and they can only be invoked server-side

4. What is JSP?

   JavaServer Pages is a Java standard technology that enables you to write dynamic, data-driven pages for your Java web applications. JSP is built on top of the Java Servlet specification. The two technologies typically work together, especially in older Java web applications. From a coding perspective, the most obvious difference between them is that with servlets you write Java code and then embed client-side markup (like HTML) into that code, whereas with JSP you start with the client-side script or markup, then embed JSP tags to connect your page to the Java backend.

5. What is JSP Expression Language?

   It makes it possible to easily access application data stored in JavaBeans components, allowing you to create expressions that are arithmetic and/or logical. Within a JSP EL expression, you can use integers, floating point numbers, strings, booleans, and null. For example:

```
<jsp:setProperty name = "box" property = "perimeter"
value = "${2*box.width+2*box.height}"/>

```

6. What is JSTL?

   The **JavaServer Pages Standard Tag Library** is a collection of useful JSP tags which encapsulates the core functionality common to many JSP applications. It has support for common, structural tasks such as iteration and conditionals, tags for manipulating XML documents, internationalization tags, and SQL tags. It also provides a framework for integrating the existing custom tags with the JSTL tags.
   You must include this to use them:

```
<%@ taglib prefix = "c" uri = "http://java.sun.com/jsp/jstl/core" %>

```

7. What does 'Reference Implementation' mean?

   It means the implementation of the specification that should demonstrate the concepts. Mostly it is implemented by the same guys/company that designed the specification.
   You may think of specification as a standard that is needed to allow other possible implementation to be compatible with the rest of the world. And the reference implementation is a proof-of-concept piece of software that should show how to do that and encourage others to create their own implementations.
   In the JDBC context it means, that there are some interfaces (CachedRowSet) prescribing some methods and there is a reference implementation to these interfaces done in Sun/Oracle.
   Basically, it's the basic bare-minimum implementation that demonstrates the usability of an interface as a reference to the user using it or overriding it.

8. What is a Model-1 application?

   With the Model 1 architecture, the JSP page handles all of the processing of the request and is responsible for displaying the output to the client.

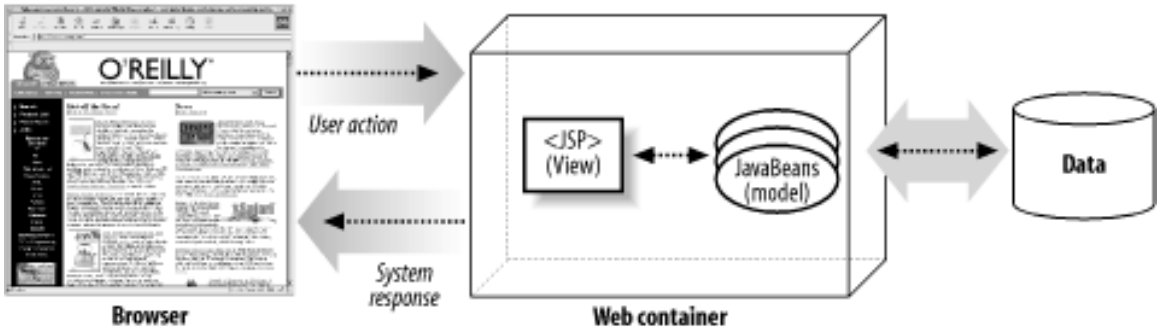

Figure 6: Model 1 Application

Notice that there is no servlet involved in the process. The client request is sent directly to a JSP page, which may communicate with JavaBeans or other services, but ultimately the JSP page selects the next page for the client. The next view is determined based on either the JSP selected or parameters within the client's request.

9. What is a Model-2 application?

   The client request is first intercepted by a servlet, commonly referred to as a controller servlet. This servlet handles the initial processing of the request and determines which JSP page to display next.

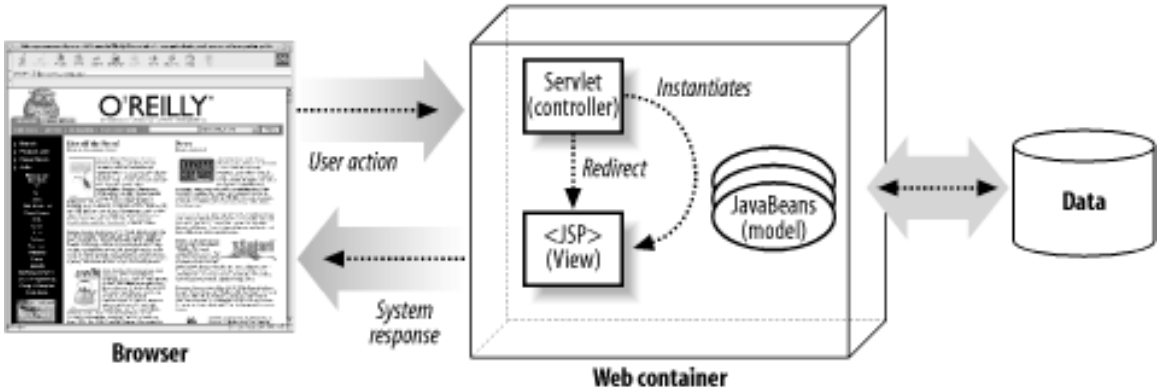

Figure 7: Model 2 Application

As shown in the figure, a client never sends a request directly to a JSP page in the Model 2 architecture. This allows the servlet to perform front-end processing, including authentication and authorization, centralized logging, and help with internationalization. Once request processing has completed, the servlet directs the request to the appropriate JSP page. How the next page is determined varies widely across different applications. For example, in simpler applications, the next JSP page to display may

be hardcoded in the servlet based on the request, parameters, and current application state. In more sophisticated web applications, a workflow/rules engine might possibly be used.

10. What is a WAR file?

    These files are with the .war extension. The war file contains the web application that can be deployed on the any servlet/jsp container. The .war file also contains jsp, html, javascript and other files that are necessary for the development of the web applications.
    Whereas with JAR files, they contain the libraries, resources and accessories files like property files.

11. What is a Web Service?

    Web services are similar to web applications; the main difference is that **web services return data** rather than entire HTML documents. We also learned that web services (like web applications) **are served over HTTP** and that they can have many different types of clients.

12. What does API stand for?

    Application Programming Interface - basically its the building blocks already created for you to use in your project, such as a file uploader, or fingerprint authentication.
    An API isn't the same as the remote server — rather it is the part of the server that receives requests and sends responses.
    Think of an API like a menu in a restaurant. The menu provides a list of dishes you can order, along with a description of each dish. When you specify what menu items you want, the restaurant's kitchen does the work and provides you with some finished dishes. You don't know exactly how the restaurant prepares that food, and you don't really need to.
    Similarly, an API lists a bunch of operations that developers can use, along with a description of what they do. The developer doesn't necessarily need to know how, for example, an operating system builds and presents a "Save As" dialog box. They just need to know that it's available for use in their app.
    This isn't a perfect metaphor, as developers may have to provide their own data to the API to get the results, so perhaps it's more like a fancy restaurant where you can provide some of your own ingredients the kitchen will work with.

4. Spring MVC

1. What is Spring MVC?

   Spring MVC is more than a library. It's a framework. Spring MVC supports a wide range of web applications, from apps that present a complex user interface rendered in HTML, CSS, and JavaScript, to web services that function as a data backend.
   It allows us to write simple Java code that generates HTTP responses from requests.
   It implements the MVC pattern and used Spring dependency injection.

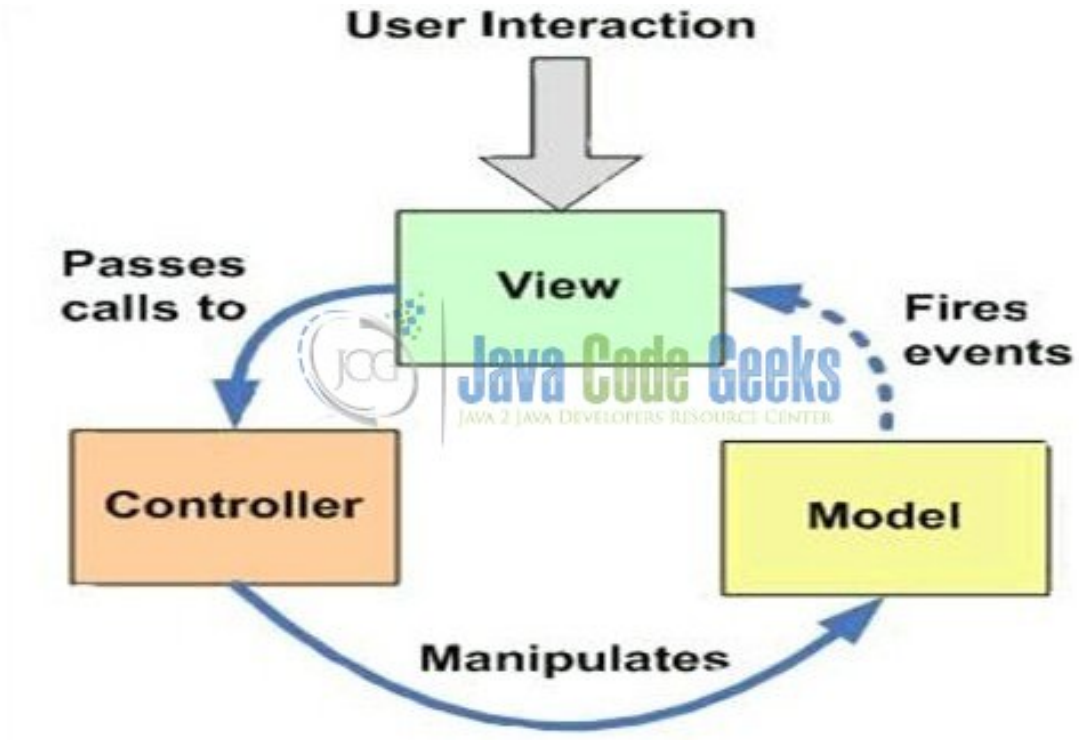

Figure 8: Spring MVC Diagram

2. What view technology are we using for our Spring MVC projects?

   Postman is a way to view the raw data being sent and to send data without having to create an entire front end.

3. What is the Front Controller Pattern?

   The Front Controller Pattern provides a single entry point for all requests into an application. For Spring MVC, the front controller is the DispatcherServlet, which handles all incoming requests and then routes them to proper Controller component

in the application.

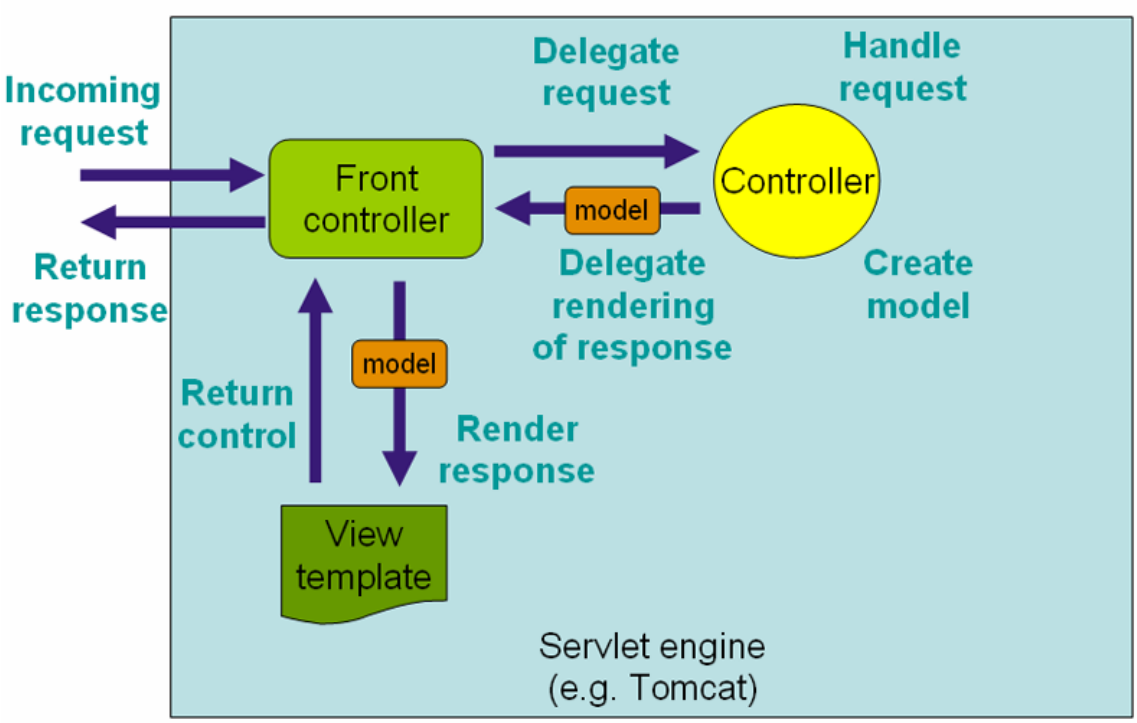

Figure 9: Front Controller Pattern

4. What role does the Dispatcher Servlet play in the Spring MVC web app?

   The Dispatcher Servlet acts as the Front Controller.

5. What role does the Controller play in the Spring MVC web app?

   The Controller is the final destination point that a web request can reach. After being invoked, the controller method starts to process the web request by interacting with the service layer to complete the work that needs to be done.

6. What role does the View Resolver play in the Spring MVC web app?

   View Resolvers enable you to render models in a browser without tying you to a specific view technology like JSP, Velocity, XML... etc.

7. How are URL patterns mapped to controller code in Spring MVC?

   Through annotation.

```
    @RequestMapping("/home", 'GET')
    @ResponseBody
    public String getHome()
    {
        return "/home";
    }
    // or
    @GetMapping("/home")
    public String getHome()
    {
        return "/home";
    }

```

8. Spring Boot Annotations

Annotation	Purpose
@SpringBootApplication	used on App, denotes program uses Spring Boot framework
SpringApplication.run(App.class, args);	placed in main method, runs the spring boot framework
@RestController	denotes controller in a REST API
@RequestMapping(" ")	denotes url path to api, placed before controller class declaration
@Autowired	used to let Spring Boot know we want it to set up these variables (like ServiceLayer or Dao) for us
@GetMapping(" ")	denotes method to run in controller in response to GET url
@PostMapping(" ")	denotes method to run in controller in response to POST url
@ResponseStatus(HttpStatus.)	denotes response status to return
@RequestBody	found in parameter list, tells method to get parameters from body sent with POST
@ModelAttribute	creates Model object you can use to pass variables to the templates by adding attributes to it
@PathVariable	found in parameter list, tells method to get parameters from URL
@ControllerAdvice	found on ControllerExceptionHandler, tells spring boot this component will be helping the controller (also must include @RestController after)
@Repository	placed on classes that have to do with memory (ex. GameDaoDB), not on interfaces
@Service	denotes the service layer implementation

@Component	denotes class for spring boot to look for and compile as part of program
@Configuration	tells spring boot to look within class for its own configuration
@ComponentScan	requests spring boot to search for components along specifications
@EnableAutoConfiguration	allows spring boot to configure automatically
@Profile	allows developer to group classes in a profile to separate which are used when. For example, you could have a memory profile and database profile. One uses classes to store and access info in a database, the other from memory.
@RunWith(SpringRunner.class)	tells it to run junit using the spring runner (changes how we set-up test if not using)
@SpringBootTest	denotes a test class
@Before	denotes method to run before each test
@After	denotes method to run after each test
@Test	denotes method of test
@MockBean	used by system to create mock componenet during testing
@Valid	added in parameter list of mapping function to validate input as whole object
@Entity	denotes an object represented in the database (for the JPA to manage in table with that name)
@Entity(name = " ")	allows us to hardcode the table name so the class name doesn't have to match
@GeneratedValue (strategy=GenerationType.IDENTITY)	tells JPA field is auto-incremented, and the strategy parts says that MySQL is going to manage that part for us
@Id	tells JPA this is the primary key for all queries that require one (make sure import is javax.persistence)
@Column	any other field with standard data in it (aka not another entity) to pull from column with that name
@Column(name = " ")	can hardcode the column name if we want
@Column(nullable = false)	tells JPA this field has a NOT NULL characteristic in the database (can also denote length, unique, scale, precision, etc.)
@ManyToOne	indicates many to one relationship
@JoinColumn(name = " ")	indicates what field in the "many" table is the foreign key to the "one" table
@ManyToMany	indicates many to many relationship
@JoinTable(name = " ", joinColumns = {@JoinColumn(name = " ")}, inverseJoinColumns = {@JoinColumn(name = " ")})	the name defines the bridge table to the JPA, the field for the object we're in is the joinColumns, and the field for the other object is the inverse.

@ManyToMany(mappedBy = "inverse entity")	since relationship already defined, let's us easily "reverse" it and manage it from the other side without having to redefine it
@Query	tells JPA exact query you want to run for a method
@Configuration	denotes config file
@EnableWebSecurity	tells it to use this class to set its self up
@NotNull	validates property value is not null
@AssertTrue	validates property value is true
@Min(#)	validates property has value ¿ #
@Max(#)	validates property has value ¡ #
@NotEmpty	validates property value is not null nor empty, only good for String, Collection, Map, or Arrays
@Positive	validates numeric value is ¿ 0
@PositiveOrZero	validates numeric value is ¿= 0
@Negative	validates numeric value is ¡ 0
@NegativeOrZero	validates numeric value is ¡= 0
@Past	validates date is in past
@PastOrPresent	validates date is in past or present
@Future	validates date is in future
@FutureOrPresent	validates date is in future or present
@NotBlank(message = " ")	requires variable to not be blank, shows message if it is
@Size(max = #, min = #)	confines length of variable (size)
@Digits(integer=X, fraction=X)	checks if number has required number of integral digits and fractional digits (both attributes are required when using)
@Email	verifies field is a well-formed email
@Pattern(regexp=X)	verifies field follows the regular expression pattern X

Table 2: Spring Boot Annotations

5. Databases

1. What is a relational database?

   Relational databases organize data into one or more tables or relations. Relation is the formal, academic term. In day-to-day professional work, we usually say table, which is the term preferred in this course. A table is a logical group of data.
   Each row represents an object and the column are its attributes.

2. What are some advantages of a relational database?

   Highly structured and designed to minimize how much storage space the data re-

quires and reducing data anomalies.
The application can retrieve the data relatively quickly because of the organizational structure.
All use SQL to retrieve and manage the data.

3. What is the purpose of a database management system?

   A DBMS is a software system that manages our databases. The DBMS executes commands, provides security, enables network access, and provides admin tools for Database Administrators (DBAs) to work with database files.

4. What is the difference between a MySQL server and a MySQL client?

   The MySQL server is a server which you can interact with using a MySQL client. You can use the MySQL client to send commands to any MySQL server; on a remote computer or your own. The MySQL server is used to persist the data and provide a query interface for it (SQL). The MySQL clients purpose is to allow you to use that query interface.

5. What is MySQL?

   MySQL is a database management system.

6. What is MySQL Workbench?

   MySQL Workbench is a unified visual tool for database architects, developers, and DBAs. MySQL Workbench provides data modeling, SQL development, and comprehensive administration tools for server configuration, user administration, backup, and much more.

7. What is PHPMyAdmin?

   phpMyAdmin is a free software tool written in PHP, intended to handle the administration of MySQL over the Web.

8. What is normalization?

   The process of breaking down these relationships into simpler structures. A properly-normalized design improves performance and reduces the complexity of relationships by minimizing data duplication (redundancy).
   First Normal Form (1NF):

- No top-to-bottom nor left-to-right ordering of rows (can be accessed in any order)
- Every row can be uniquely identified
- Every entry consists of one entry, not multiple (no commas, etc.)

Second Normal Form (2NF):

- Is in 1NF
- All the columns except the primary key should strictly relate to each other
- Not repeating data across rows.

Third Normal Form (3NF):

- Is in 2NF
- No non-primary-key column is functionally dependent on any non-key set of fields

9. What is a primary key?

   The unique identifier for a row of data. Must be unique and must exist.

10. What is a foreign key?

    Reference to the primary key of another table. The value must match an existing primary key in the table it is binded to.

11. What is a one-to-one relationship? (With example)

    Each row in one table corresponds to exactly one row in the other table and vice versa. For example:

SSN	Name	DOB
123-45-6789	Billy Jean	01-01-2000
987-65-4321	Jadie Tae	04-04-1999

SSN	Driver's License
123-45-6789	123456789
987-65-4321	987654321

Table 3: One to One Relationship

12. What is a one-to-many relationship? (With example)

    Each record in the first table corresponds to multiple records in the second table, but each record in the second table corresponds to only one record in the first table. For example:

SSN	Name	DOB
123-45-6789	Billy Jean	01-01-2000
987-65-4321	Jadie Tae	04-04-1999

SSN	Children SSN
123-45-6789	123-45-9876
123-45-6789	123-45-8765

Table 4: One to Many Relationship

13. What is a many-to-many relationship? (With example)

    Each record in the first table can have multiple corresponding records in the second table, and vice versa. For example:

SSN	Name	DOB
123-45-6789	Billy Jean	01-01-2000
987-65-4321	Jadie Tae	04-04-1999

SSN	Former Addresses
123-45-6789	123 Main Street
123-45-6789	456 White Street
987-65-4321	123 Main Street
987-65-4321	456 White Street

Table 5: Many to Many Relationship

    But, this repeats a lot of information. So we create a bridge table:

SSN	Name	DOB
123-45-6789	Billy Jean	01-01-2000
987-65-4321	Jadie Tae	04-04-1999

ID	Former Addresses
1	123 Main Street
2	456 White Street

SSN	AddressID
123-45-6789	1
123-45-6789	2
987-65-4321	1
987-65-4321	2

Table 6: Many to Many Relationship with Bridge Table

14. What is referential integrity?

    This is the constraint placed on foreign keys, meaning that the thing they reference must be a valid reference (must exist).

15. What is a transaction?

    A sequence of operations performed (using one or more SQL statements) on a database as a single logical unit of work. The effects of all the SQL statements in a transaction can be either all committed (applied to the database) or all rolled back (undone from

the database).

16. What does commit mean?

    Means it is written to record, whether that be adding to the database, updating it, or deleting something. It gets done.

17. What does rollback mean?

    This means we're going to undo everything we just did in the transaction.

18. What is a dangling reference?

    A reference to an object that no longer exists...can happen when you delete an object that is being referenced by a foreign key in another table.

19. What is a bridge table?

    A table consisting of the ids of the two tables it is bridging. Represents the many-to-many relationship betwixt the two.

20. What type of relationships require a bridge table?

    Many-to-Many relationships

21. What is a validation table?

    Also known as a **lookup table**, stores data that you specifically use to implement data integrity. You won't often insert, update, or delete any records within the table once you populate the table with the data you require. Validation tables usually (but not always) comprise two fields: The first acts as the primary key and is what you'll use to help you enforce data integrity, and the second is simply a non-key field that stores a set of values required by some other field in the database. For example:

ID	State
AL	Alabama
AK	Alaska
AR	Arkansas
AZ	Arizona
⋮	⋮

Table 7: U.S. States Validation Table

22. What does cascade mean?

    DELETE CASCADE: When we create a foreign key using this option, it deletes the referencing rows in the child table when the referenced row is deleted in the parent table which has a primary key.
    UPDATE CASCADE: When we create a foreign key using UPDATE CASCADE the referencing rows are updated in the child table when the referenced row is updated in the parent table which has a primary key.

23. What is a SQL select statement?

    Gets requested data from table.

24. What is a SQL delete statement?

    Deletes rows in table as specified.

25. What is a SQL update statement?

    Updates rows in table (cannot change the primary key) as denoted.

26. What is a SQL join statement?

    Cross-references tables to find intersections as requested.

27. What are the types of SQL join statements?

    An **INNER JOIN** returns a result only when rows from both tables match on their relationship condition.

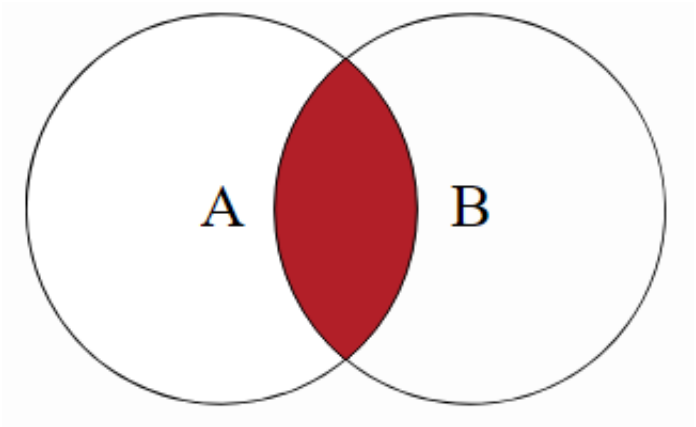

Figure 10: Inner Join

```sql
SELECT
    TaskId,
    Title,
    `Name`,
    TaskStatusId -- This will cause problems.
FROM TaskStatus
INNER JOIN Task ON TaskStatus.TaskStatusId = Task.TaskStatusId
WHERE TaskStatus.IsResolved = 1;
```

Figure 11: Inner Join Example

When the [Join Type] is **LEFT**, the results include "everything from table A and whatever matches from table B."

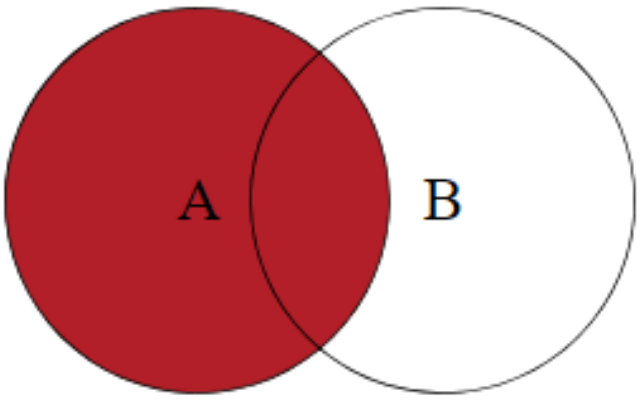

Figure 12: Left Outer Join

```sql
SELECT *
FROM Task
LEFT OUTER JOIN TaskStatus
    ON Task.TaskStatusId = TaskStatus.TaskStatusId;
```

Figure 13: Inner Join Example

RIGHT results include "everything from table B and whatever matches from table A."

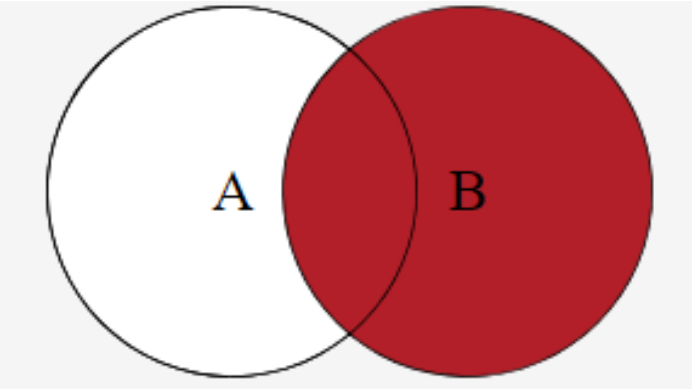

Figure 14: Right Outer Join

FULL OUTER JOIN results are "everything from both tables regardless of match."

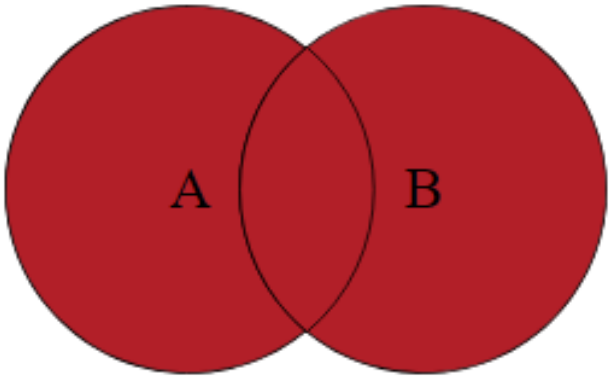

Figure 15: Full Outer Join

28. What is a Cartesian Product (SQL cross join)?

CROSS JOIN does not use an ON clause because it does not match on a condition. Instead, CROSS JOIN creates a Cartesian product, with every possible combination of rows between the joined tables included in the results. The results don't show

actual relationships, they just show every possible combination.

```
SELECT
   CONCAT(w.FirstName, ' ', w.LastName) WorkerName,
   p.Name ProjectName
FROM Worker w
CROSS JOIN Project p
WHERE w.WorkerId = 1
AND p.ProjectId NOT LIKE 'game-%';
```

Figure 16: Cross Join Example

29. What is JDBC?

    Java Database Connectivity is a Java API for connecting to databases. It was first released in 1997 and has been continually refined since. JDBC is a collection of interfaces, classes, and abstract classes that provide a consistent way of connecting to relational databases, regardless of who is providing the database.
    Before JDBC, each database had its own API – network protocols, metaphors, and conventions – to store and retrieve data from their database. If you ever needed to change the database system used in an application, it was a long process to rewrite all the code necessary to correctly communicate with the new system. JDBC is the solution to that problem.
    Database vendors now provide a driver for their database that implements the JDBC interface. As a developer, that means we don't necessarily need to worry about the minutiae of how to connect and communicate with the database. Also, if we need to switch to a new database, it can be as simple as changing the driver we are using to connect.

30. What are the advantages of using Spring JDBC Templates?

    SpringBoot will set up our database connection for us using our application properties file.
    We pass our queries directly in and don't have to worry about using Statement and PreparedStatement. An overloaded RowMapper() iterates through the table for us.
    More efficient.

31. What is a prepared statement?

    A single SQL query that can use dynamic parameters. The parameters are added into the query in such a way that they are sanitized, which prevents SQL injection attacks. Prepared Statements can be for any valid SQL query.

32. When should prepared statements be used?

    Whenever the statement is being prepared using input from a user. It's what protects us from SQL injection by sanitizing the input.

33. Why should transactions be used?

    Transactions will rollback the changes if an error occurs. That way, you don't end up with partially edited data due to errors or power failure that can really mess you up in the future.

34. What can happen if transactions are not used when needed?

    We can end up with partially edited data, not knowing without perusing the database what was successfully done and what wasn't. If uncaught, can desecrate data integrity.

35. Do all SQL statements require a transaction?

    No, only those with multiple modifications.

36. What is an ORM?

    Object Relational Mapper - basically what JPA is. It lets us call queries in Java without having to actually write them. Instead, we just call them as functions on the object. It does all the work in its little black box.