M149: Database Systems **Winter 2018**

Application Design and Development: October 30

Lecturer: Panagiotis Liakos

1 Applications Programs and User Interfaces

- very few people use a query language to interact with a database system directly
- usually this is done through an application program that provides a user interface at the front end, and interfaces with a database at the back end
- a typical application program includes:
  - front-end component: deals with the user interface
  - back-end component: communicates with a database
  - middle layer: contains "business logic", i.e., code that executes specific requests for information or updates, enforcing rules of business such as what actions should be carried out to execute a given task, or who can carry out what task
- evolution of applications:
  - in the "early-days" applications ran on a large "mainframe" computer and users interacted with the application through terminals
  - when personal computers emerged applications were installed on each user's machine and accessed a central database
  - in the last two decades Web browsers have become the universal front-end to database applications, connecting to the back-end through the Internet.

2 Web fundamentals

- URL (Uniform Resource Locator): http://www.google.gr/search?q=databases
  - http: indicates that the document is to be accessed by the HyperText Transfer Protocol
  - www.google.gr: gives the name of a machine that has a Web server
  - search: the program to be executed
  - q=databases: the arguments for the program to be executed
- HTML:
  - Tables
  - Forms
  - etc.

- Web Server: program running on the server machine, accepts requests from a Web browser and sends back results in the form of HTML documents. Can act as an intermediary to provide access to a variety of information services
  - Connectionless: no continuous connection between the client and the Web server (performancre reasons on the server side)
    * Session Based Authentication (cookies): google.com may set a cookie with the name prefs, which encodes preferences set by the user such as the preferred language and the number of answers displayed per page. google.com can retrieve the cookie named prefs from the user's browser, and display results according to the specified preferences. A domain (Web site) is permitted to retrieve only cookies that it has set, not cookies set by other domains, and cookie names can be reused across domains.
    * Token Based Authentication: Many web applications use JSON Web Token (JWT) instead of sessions for authentication (`https://jwt.io/`). In the token based application, the server creates JWT with a secret and sends the JWT to the client. The client stores the JWT (usually in local storage) and includes JWT in the header with every request. The server would then validate the JWT with every request from the client and sends response. The biggest difference here is that the users state is not stored on the server, as the state is stored inside the token on the client side instead. Most of the modern web applications use JWT for authentication for reasons including scalability and mobile device authentication. Token Based Authentication using JWT is the more recommended method in modern web apps. One drawback with JWT is that the size of JWT is much bigger comparing with the session id stored in cookie because JWT contains more user information. Care must be taken to ensure only the necessary information is included in JWT and sensitive information should be omitted to prevent XSS security attacks.

3 Web Applications / Services

Spring-boot: A framework that offers the advantages of Spring without the need for almost any configuration (convention over configuration)

- Sample application is available here: `https://goo.gl/S4KNb6`
- @SpringBootApplication is a convenience annotation that adds all of the following:
  - @EnableAutoConfiguration This annotation tells Spring Boot to "guess" how you want to configure Spring, based on the jar dependencies that you have added. Since spring-boot-starter-web added Tomcat and Spring MVC, the auto-configuration assumes that you are developing a web application and sets up Spring accordingly.
  - @ComponentScan tells Spring to look for other components, configurations, and services in the gr.di.uoa.pliakos.databasesystems package, allowing it to find the controllers.
- The main() method uses Spring Boots SpringApplication.run() method to launch an application. SpringApplication bootstraps our application, starting Spring, which, in turn, starts the auto-configured Tomcat web server. We need to pass SpringBootPostgresExample as an argument to the run method to tell SpringApplication which is the primary Spring component.

Model-View-Controller (MVC):

- Model: The Model component corresponds to all the data-related logic that the user works with. This can represent either the data that is being transferred between the View and Controller components or any other business logic-related data. For example, a Student object will retrieve the student information from the database, manipulate it and update it data back to the database or use it to render data.
- View: The View component is used for all the UI logic of the application. For example, the Student view will include all the UI components such as text boxes, dropdowns, etc. that the final user interacts with.
- Controller: Controllers act as an interface between Model and View components to process all the business logic and incoming requests, manipulate data using the Model component and interact with the Views to render the final output. For example, the Student controller will handle all the interactions and inputs from the Student View and update the database using the Student Model. The same controller will be used to view the Student data.

Hibernate:

- A framework that maps the objects of a programming language to database relations (ORM). Once you map the objects, you get advantages of OOP concepts like inheritance, and encapsulation.
- Hibernate is database independent as it has its own query language.
- It provides a caching mechanism (1st level & 2nd level cache): you don't need to hit the database for similar queries, you can cache the result and use it from buffered memory to improve performance.
- It supports Lazy loading (Load only what you need).

Web services:

- a wide variety of data is available on the Web that is intended to be processed by a program, rather than displayed directly to the user; Such data are typically accessed using what is in effect a Web application programming interface; that is, a function call request is sent using the HTTP protocol, executed at an application server, and the results sent back to the calling program. A system that supports such an interface is called a Web service.
- RESTful web services:
  - Use HTTP methods to map CRUD (create, retrieve, update, delete) operations to HTTP requests.
  - Resources expose easily understood directory structure URIs.
  - Representations transfer JSON or XML to represent data objects and attributes.
  - Stateless interactions store no client context on the server between requests. State dependencies limit and restrict scalability. The client holds session state.
- In many applications of RESTful Web services, the requestor is JavaScript code running in a Web browser; the code updates the browser screen using the result of the function call. For example, when you scroll the display on a map interface on the Web, the part of the map that needs to be newly displayed may be fetched by JavaScript code using a REST ful interface, and then displayed on the screen.

Client side scripting:

- JavaScript is by far the most widely used (jquery, React, Angular, Vue.js).
- Error checks (validation) on user input, such as a date string being properly formatted, or a value entered (such as age) being in an appropriate range. These checks are carried out on the browser as data is entered, even before the data are sent to the Web server. JavaScript code can modify the tree structure to carry out certain operations. For example, suppose a user needs to enter a number of rows of data, for example multiple items in a single bill. A table containing text boxes and other form input methods can be used to gather user input. The table may have a default size, but if more rows are needed, the user may click on a button labeled (for example) "Add Item."

4 Application Performance

An application may be accessed by millions of people from across the globe, at rates of thousands of requests per second, or even more, for the most popular sites.

- Connection pooling: The application maintains a pool of connections with the database server and uses them to serve incoming requests, thus eliminating the need to establish a new connection for every request.
- Caching: Certain requests may result in exactly the same query being resubmitted to the database. The cost of communication with the database can be greatly reduced by caching the results of earlier queries and reusing them, so long as the query result has not changed at the database. Costs can be further reduced by caching the final Web page that is sent in response to a request.
- Parallel Processing: use a large number of application servers running in parallel, each handling a fraction of the requests. A Web server or a network router can be used to route each client request to one of the application servers. All requests from a particular client session must go to the same application server, since the server maintains state for a client session. This property can be ensured, for example, by routing all requests from a particular IP address to the same application server.

5 Application Security

- applications must authenticate users, and ensure that users are only allowed to carry out authorized tasks. Several libraries are available to handle this task and can be used to allow for secure and rapid web application development, e.g., spring-security and devise.
- SQL injection: `select * from student where name like ‘’; <some SQL statement>;’`
  it is best to use *prepared statements* to execute SQL queries. When setting a parameter of a prepared query, JDBC automatically adds escape characters so that the user-supplied quote would no longer be able to terminate the string.
- cross-site scripting (XSS) attack:
  `<img src="http://mybank.com/transfermoney?amount=1000&toaccount=14523">`
  - disallow any HTML tags whatsoever in text input by users
  - check that the referer is valid, for example, that the referer URL is a page on the same Web site, XSS attacks that originated on a different Web page accessed by the user can be prevented the session could also be restricted to the IP address from which it was originally authenticated
  - never use a GET method to perform any updates

- never store passwords in clear text in the application code