JEE Components

Java EE Clients

Web Components

Business Components

ORM

JEE APPLICATION ARCHITECTURE

Java EE Components

- Java EE applications are made up of components.
  - **A Java EE component is a self-contained functional software unit** that is assembled into a Java EE application with its related classes and files and that communicates with other components.

- The Java EE specification defines the following Java EE components:
  - **Application clients** and **Applets** are components that run on the **client**.
  - **Web Components**.
    - Java Servlet,
    - JavaServer Faces (JSF), and
    - JavaServer Pages (JSP) technology components are web components that run on the **server.**
    - **EJB** components (enterprise beans) are business components that run on the server.

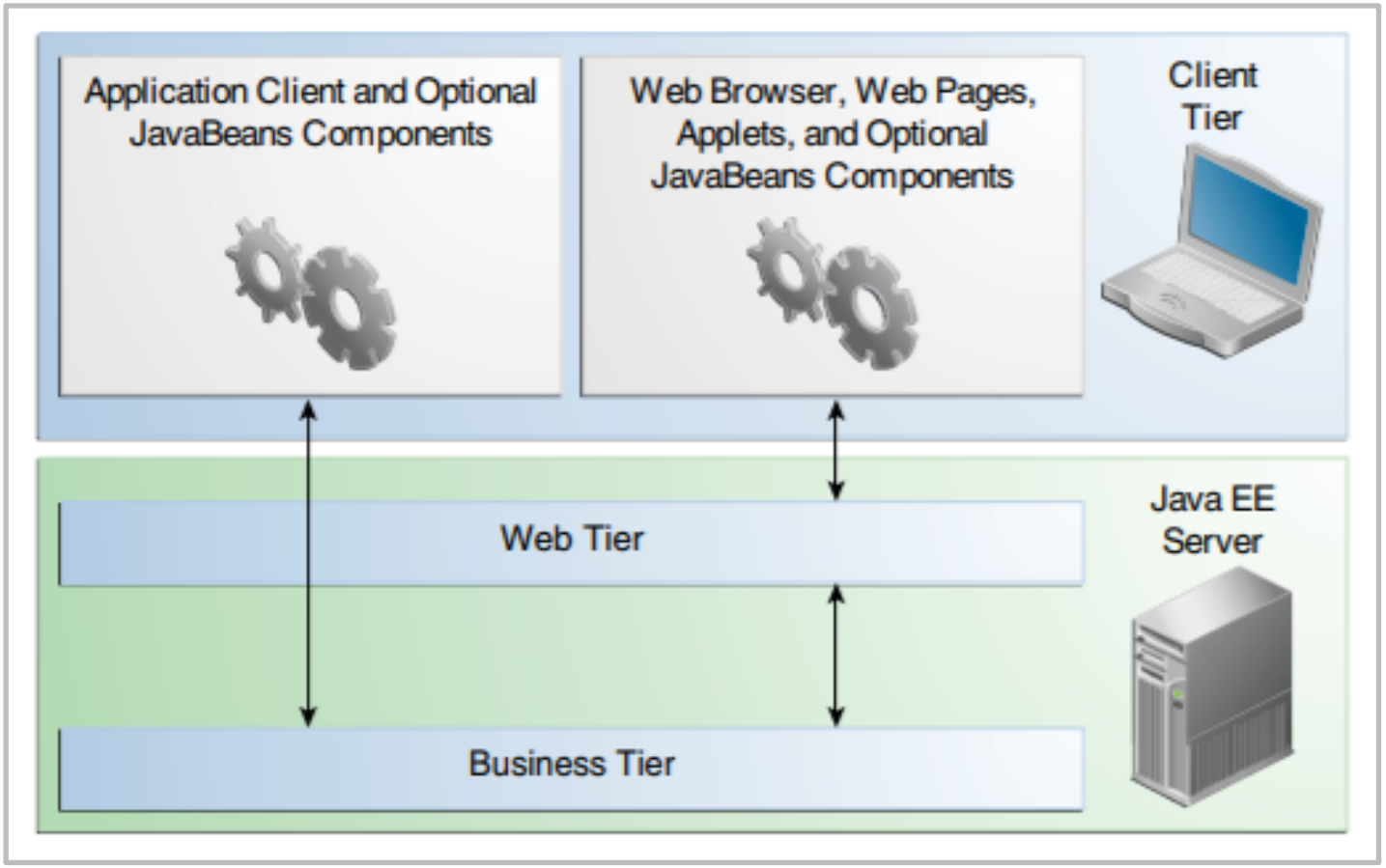
Java EE Clients

- A Java EE client is usually either a web client or an application client:

1. **Web Clients** (Thin Client). A web client consists of two parts:
    1. Dynamic web pages containing various types of markup language (HTML, XML,and so on), which are generated by web components running in the web tier.
    2. A web browser, which renders the pages received from the server
2. **Application Clients**. Runs on client machine (standalone):
    - GUI (Swing and AWT) APIs
    - FX API
3. **Applets**. A web page received from the web tier can include an embedded applet.
    - an applet is a small client application that executes in the Java virtual machine installed in the web browser

Server Communication: CLIENT

- Clients: Web browser, Web pages, Applets

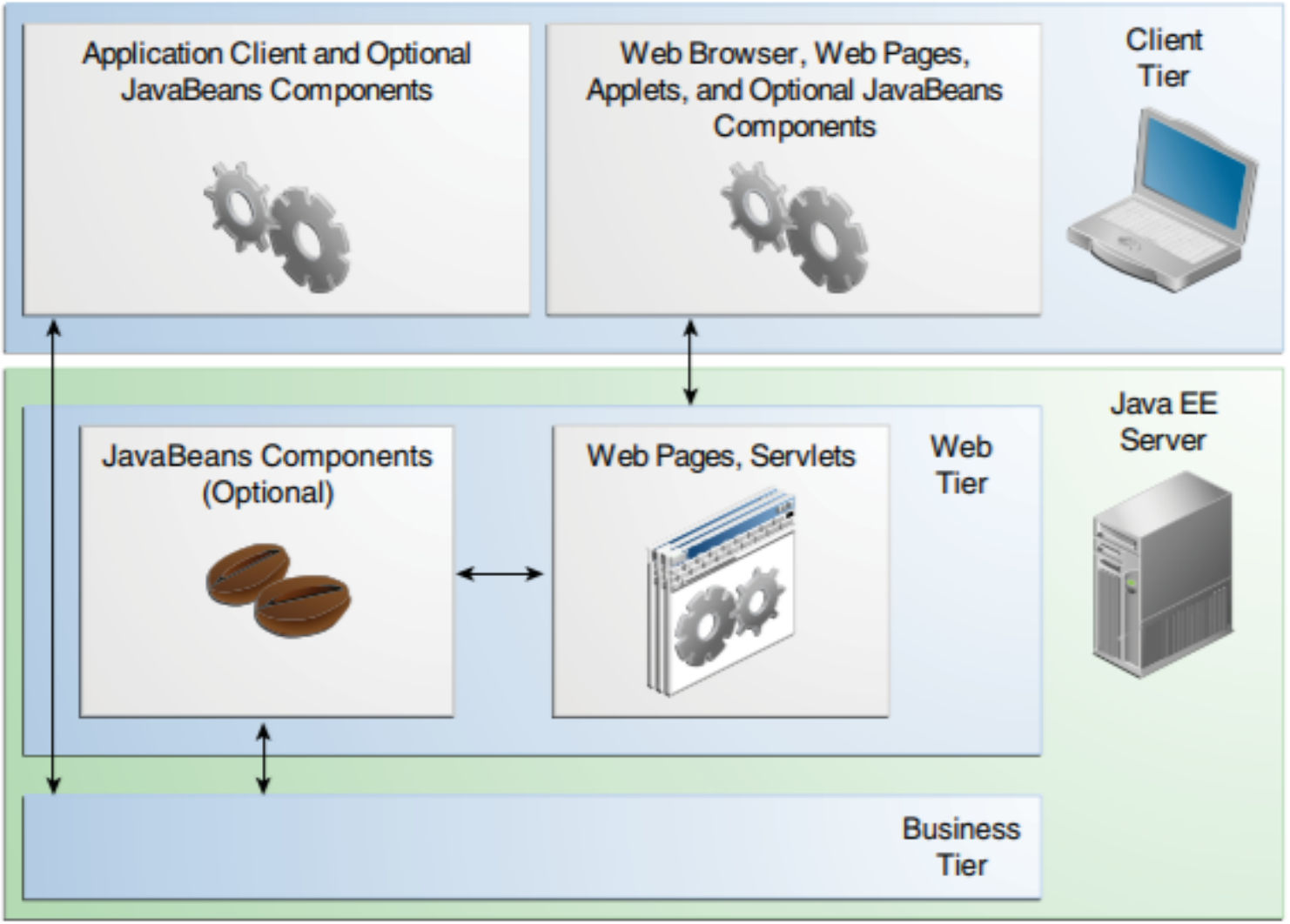


Web Components

- Java EE web components are either **servlets** or web pages created using **JavaServer Faces** technology and/or **JSP** technology (JSP pages).

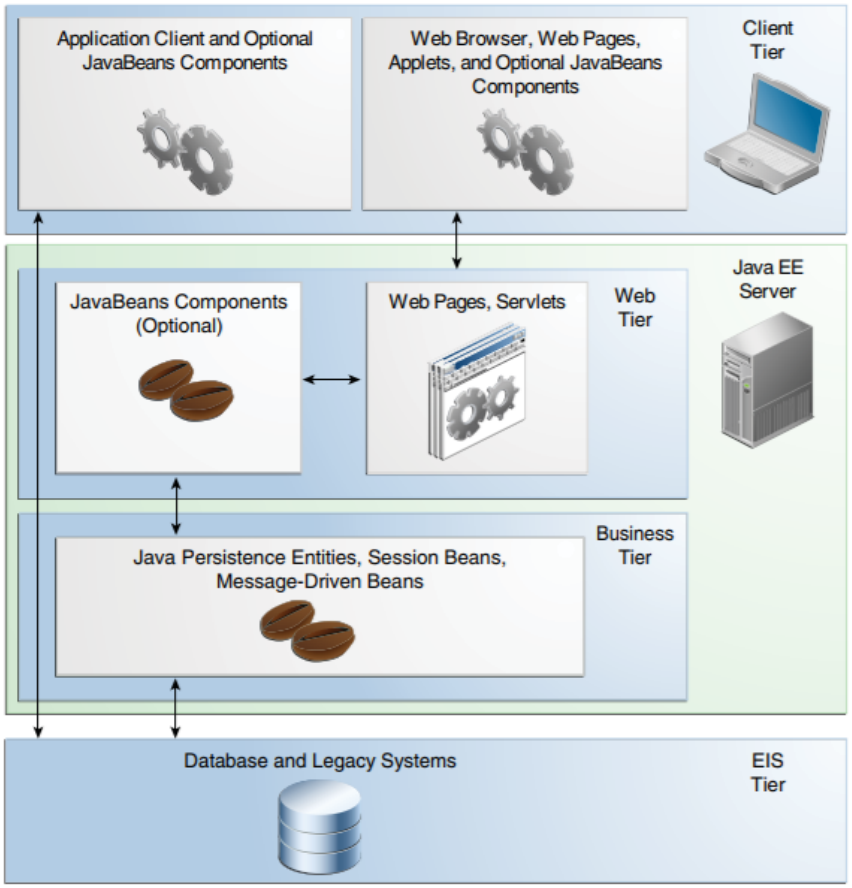
- Servlets are Java programming language classes that dynamically process requests and construct responses.
- JSP pages are text-based documents that execute as servlets but allow a more natural approach to creating static content.
- JavaServer Faces (JSF) technology builds on servlets and JSP technology and provides a user interface component framework for web applications

Web Tier and Java EE Applications



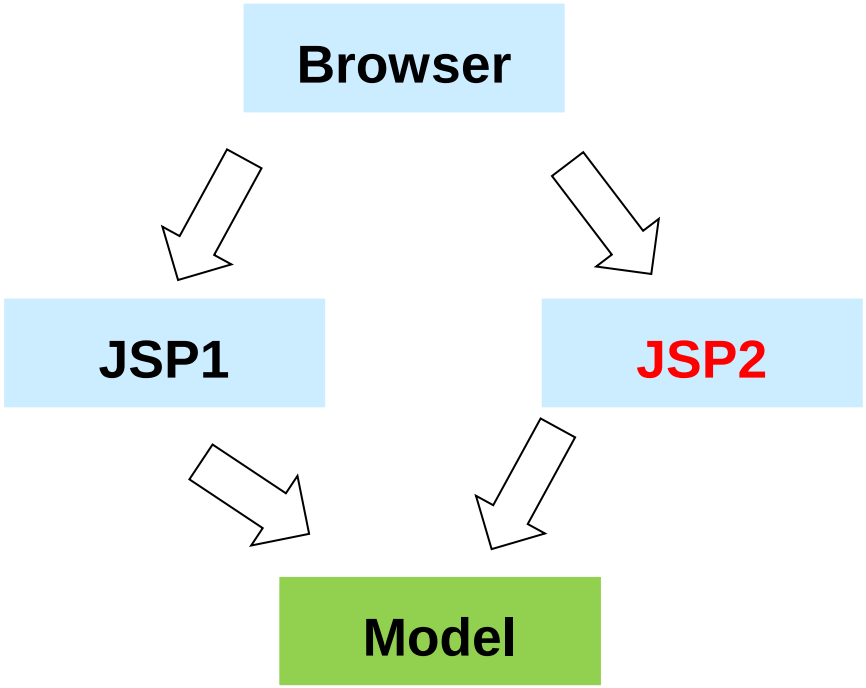
Business Components: EJB

- Business Components. Business code, which is logic that solves or meets the needs of a particular business domain such as banking, retail, or finance, is **handled by enterprise beans** running in either the business tier or the web tier.



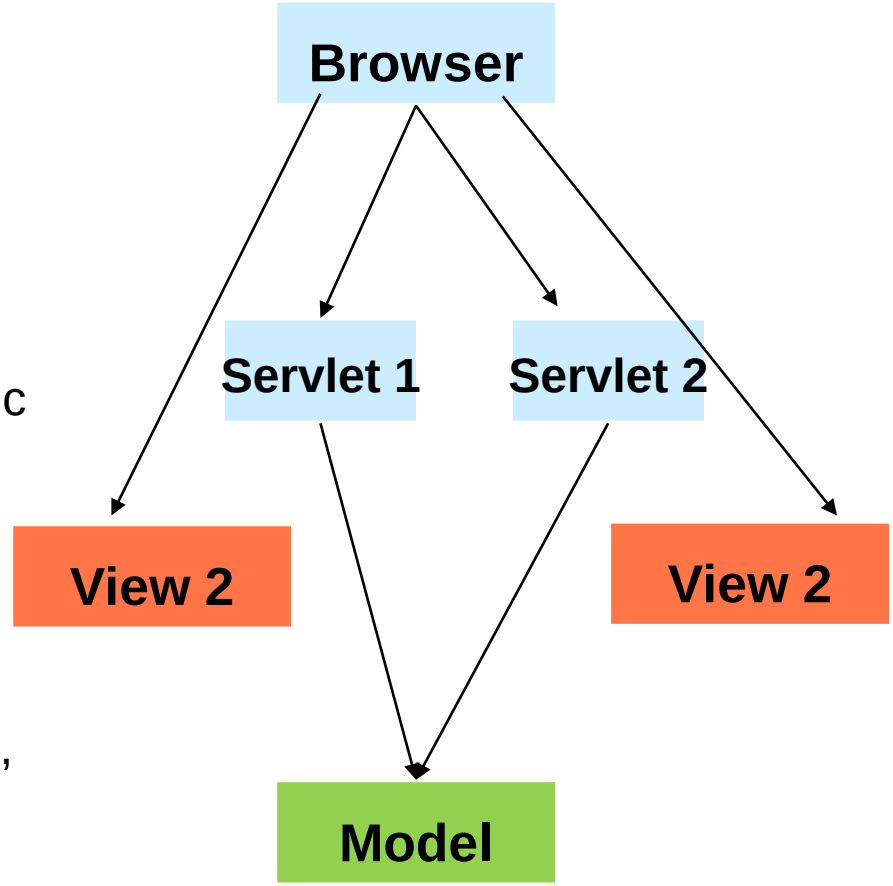
JEE Web Applications – Model 1

- Model 1 – First Generation JEE
  - No Servlets
  - JSP compiles into a Servlet
  - Request from browser to JSP
  - JSP calls Model Layer



JEE Web Applications – Model 2

- Servlet Driven
- MVC
  - Front Controller Pattern
  - Application Controller – business logic implementation
  - Data Layer – database or calls to other systems using:
    - JMS (queue messages) or
    - Web Services using HTTP (JAXB, JAX-RS, JAX-WS)



Business Logic Layer

- Provides abstractions of entities
  - e.g. students, instructors, courses, etc
- Enforces **business rules** for carrying out actions
  - E.g. student can enroll in a class only if she has completed prerequsites, and has paid her tuition fees
- Supports **workflows** which define how a task involving multiple participants is to be carried out
  - E.g. how to process application by a student applying to a university
  - Sequence of steps to carry out task
  - Error handling
    - e.g. what to do if recommendation letters not received on time
  - Workflows discussed in Section 26.2

Object-Relational Mapping

- Allows application code to be written on top of object-oriented data model, while storing data in a traditional relational database
  - alternative: implement object-oriented or object-relational database to store object model
- Schema designer has to provide a mapping between object data and relational schema
  - e.g. Java class *Student* mapped to relation *student,* with corresponding mapping of attributes
  - An object can map to multiple tuples in multiple relations
- Application opens a session, which connects to the database
- Objects can be created and saved to the database using
  session.save(object)
  - mapping used to create appropriate tuples in the database
- Query can be run to retrieve objects satisfying specified predicates

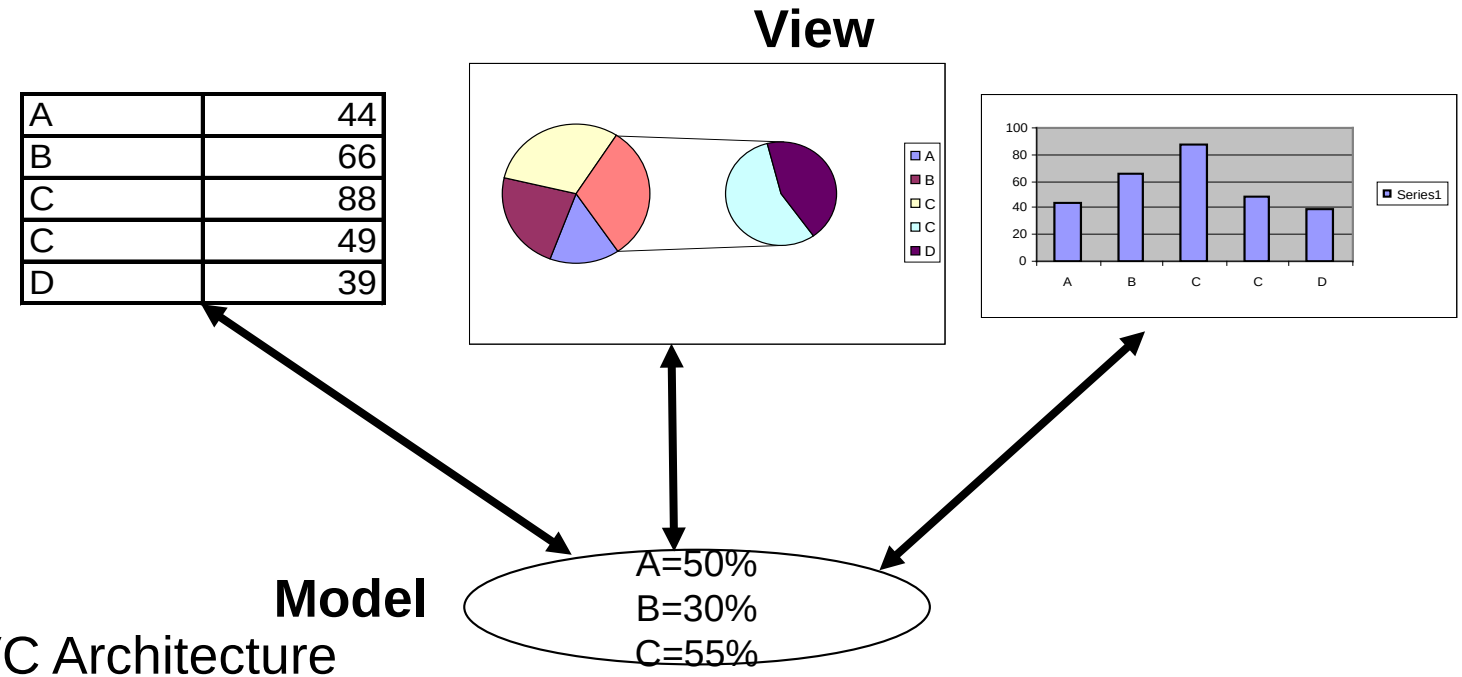
Model-View-Controller (MVC) frameworks also use Observer pattern where Model is the Subject and Views are observers that can register to get notified of any change to the model

MVC

Model-View-Controller

- **MVC Pattern**
  - Decouples views and models by establishing subscribe / notify protocol between them
  - Views represent the state of the model
  - Model contains data values
  - Controller mediates between views and model
  - See MVC Next Slide

MVC Example of Design Patterns



MVC Architecture

The **model** object holds the information

Views objects draw the visible parts of the view (or do something else)

- Display number in tables
- Display a bar chart

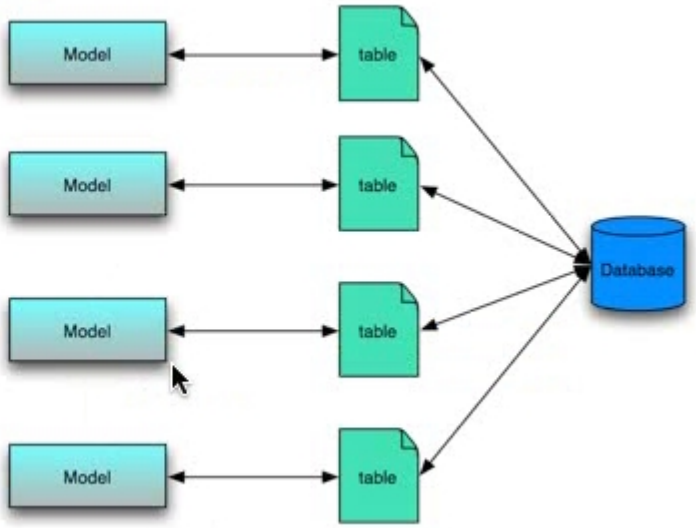
Each view has a **controller** which is an object that processes user interaction

- Mouse or keyboard events

The Model Object

- represent the state, what the current data is the database, does it need to be updated. also the model code represent the relationships between tables (m-m)
- Code that interacts directly with the db and maintains the state of the db
- a model corresponds directly to a table in a db. Elements in the table corresponds to attributes in the model class
- 4 tables for example corresponds to 4 models
- Object Relation Mapping OR/M

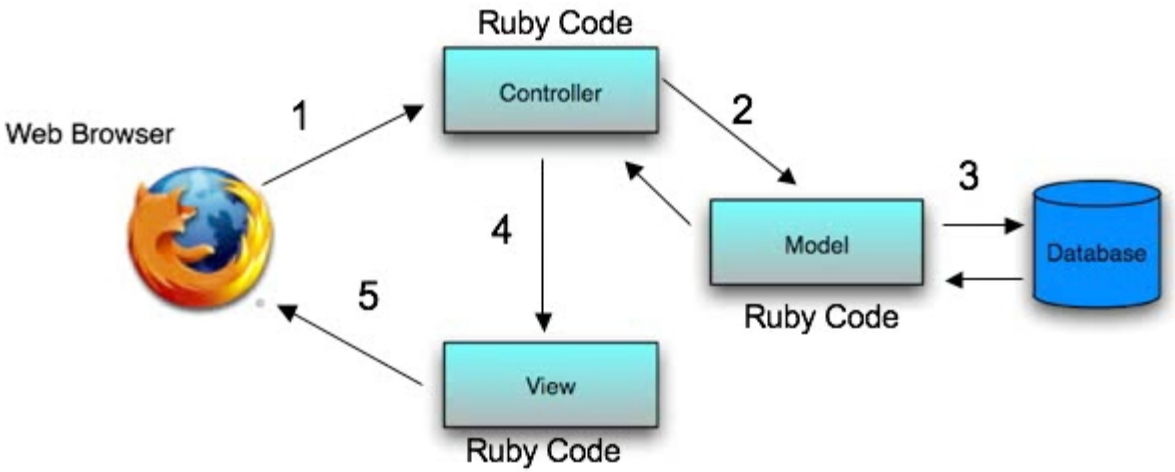
O/RM
Object-Relational Mapping



The View and Controller Objects

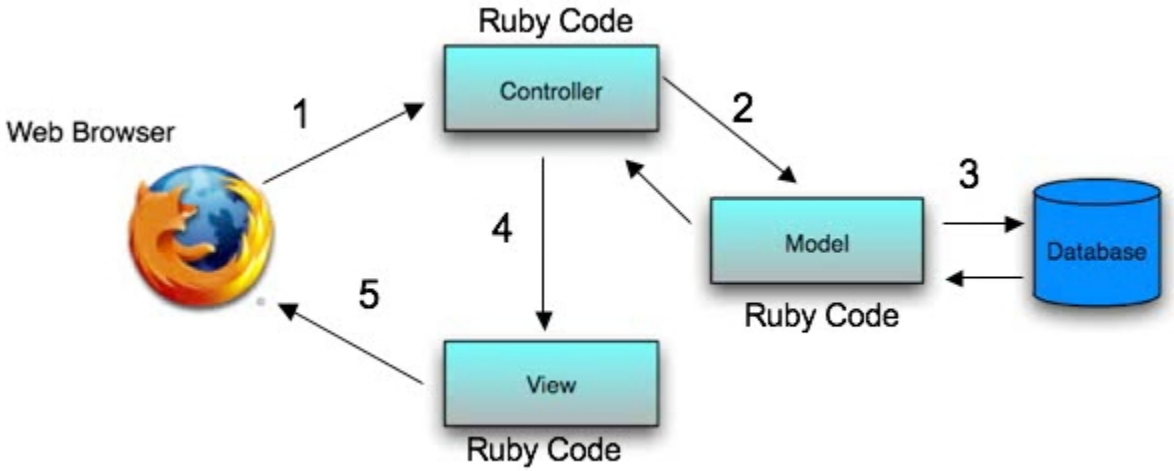
- View Object
  - generates the code that the web browser renders
  - html, javascript and other presentation technologies

- Controller object
  - The code that coordinates between the Model and the views
  - controls which model to invoke and which view to show
  - Called by the web browser



MVC

1. web browser calls the Controller
2. Controller interacts with the model
3. the model interacts with the database
4. database respond back to the controller
5. views are generated and a respond sent back to the web browser



MVC – Oracle ADF

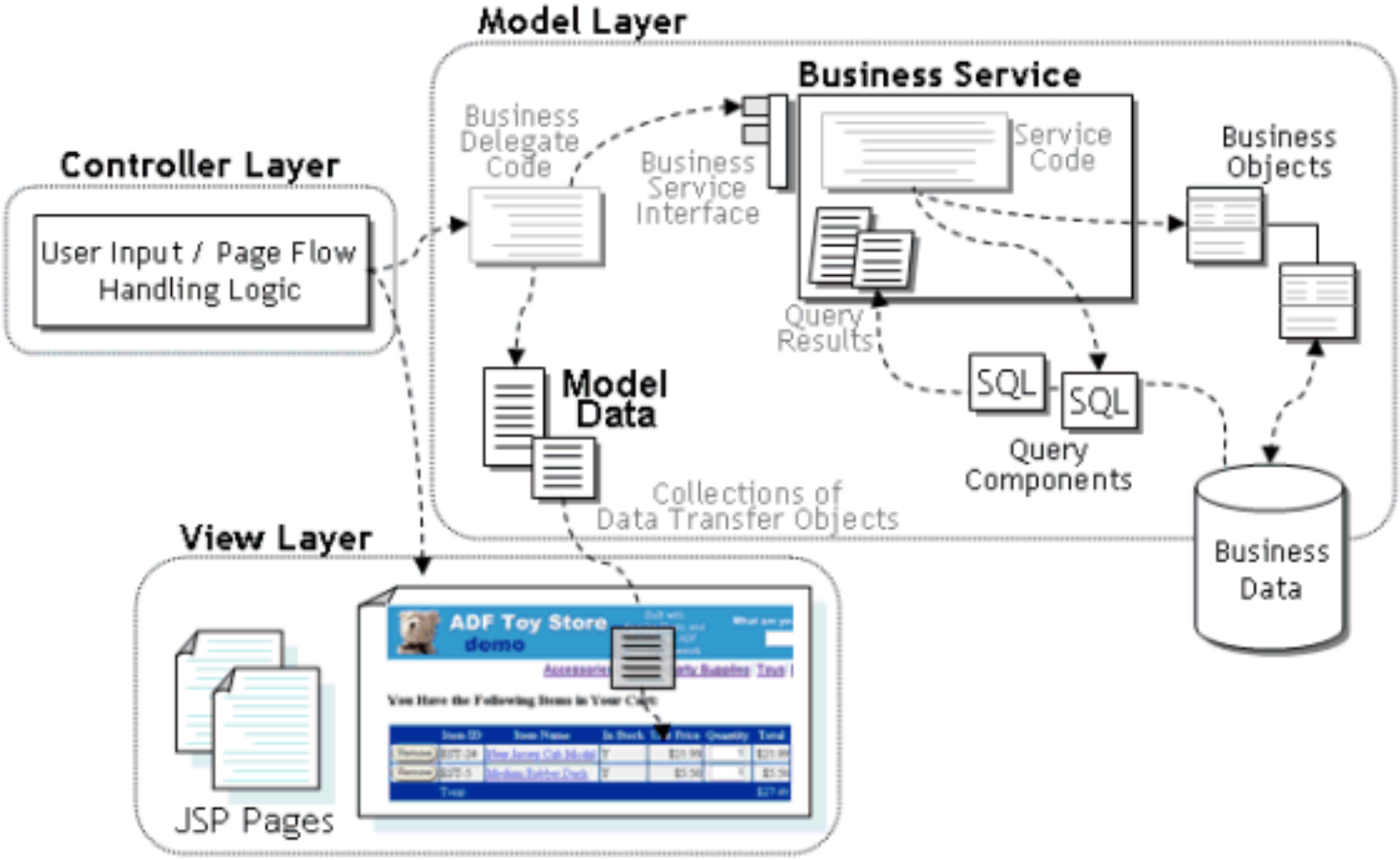
- Oracle Application Development Framework
  - The **model layer** represents the business information needed by the application,
  - The **controller layer** handles user input, interfaces with the model layer, and picks the presentation
  - The **view layer** presents the model data to the end-user.

- The model layer consists of one or more business services that expose application functionality and access to model data through a business service interface that is easy to test.

- These business services, in turn, rely on **query components** to retrieve that data and on **business objects** to validate and persist any new or modified data.

- Code implementing the business **delegate design pattern** abstracts the details of locating and using the business services.

MVC – Oracle ADF



Application Architectures

- Application layers
  - Presentation or user interface
    - **model-view-controller (MVC)** architecture
      - **model**: business logic
      - **view**: presentation of data, depends on display device
      - **controller**: receives events, executes actions, and returns a view to the user
  - **business-logic** layer
    - provides high level view of data and actions on data
      - often using an object data model
    - hides details of data storage schema
  - **data access** layer
    - interfaces between business logic layer and the underlying database
    - provides mapping from object model of business layer to relational model of database

Application Architecture