Spring Core

Capgemini

Table of Content

What you will learn

- Understand the context of Spring applications
- Implement Dependency Injection & Setter Injection for creating applications
- Define Spring Containers
- Learn Application context and Aware Interfaces
- Implement Spring beans for creating a Spring application
- Learn Bean Scopes
- Master Collections Support
- Implement Inner Bean & Aliasing
- Learn Bean Inheritance
- Initializing Bean & Dispoable Bean
- Bean Post Processor
- Bean factory post processor
- Property Placeholder Configurer
- Implement Autowiring
- Implement Annotations
- Internationalization
- Implement Event Handling
- Define Bean Life Cycle

Software requirement

- Java 8
- Spring Tool Suite (STS)
- Apache Tomcat 9.0
- Maven

Pre-requisites

- Core Java Knowledge is mandatory for this course.

Day wise Schedule

Day 1

- Introduction to Spring
  - What is Spring?
  - Overview of the Spring Framework
  - Spring Philosophies
  - Spring Modules and architecture
- A First Look at Spring
  - A Simple Example
  - Wiring Beans
  - Configuring a Properties File
  - Schema-Based Configuration
- Beans and Containers
  - Spring Containers
  - Spring Configuration File
  - Spring Beans
  - Using the Container
  - The BeanFactory Interface
  - The ApplicationContext Interface
  - Singleton vs. Prototype
  - Bean Naming
  - Dependency Injection
  - Setter Injection
  - Constructor Injection

Day wise Schedule

Day 2

- **Autowiring**
  - Autowiring through configuration
  - Autowiring by type and by name
  - Autowiring using @Autowired, @Qualifier and @Resource

- **The Bean Life cycle**
  - Configuration of initialization methods
  - The InitializingBean and DisposibleBean interfaces
  - The @PostConstruct and @PreDestroy annotations
- **Internationalization**

Course Links

Basic Course

Pluralsight Video Course

https://app.pluralsight.com/library/courses/spring-fundamentals/table-of-contents

Spring Core tutorial

http://www.mkyong.com/tutorials/spring-tutorials/

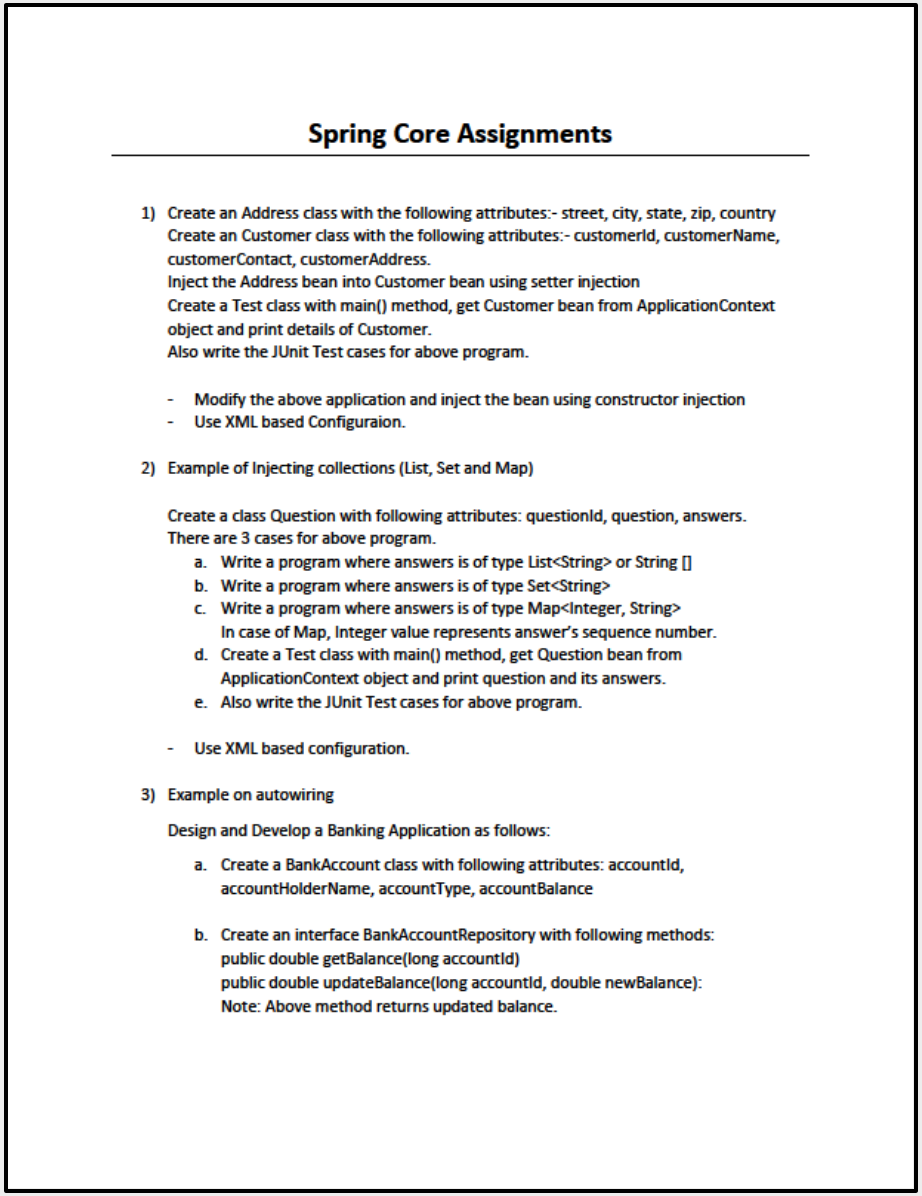
Advance Course

JavaBrains Spring Framework Video Course

https://www.youtube.com/playlist?list=PL719420CCED1DB833

Basic to advance Assignment

Please double click on image it will open the pdf



Spring Core Assignments

1) Create an Address class with the following attributes:- street, city, state, zip, country
   Create an Customer class with the following attributes:- customerId, customerName, customerContact, customerAddress.
   Inject the Address bean into Customer bean using setter injection
   Create a Test class with main() method, get Customer bean from ApplicationContext object and print details of Customer.
   Also write the JUnit Test cases for above program.

   - Modify the above application and inject the bean using constructor injection
   - Use XML based Configuraion.

2) Example of Injecting collections (List, Set and Map)

   Create a class Question with following attributes: questionId, question, answers.
   There are 3 cases for above program.
   a. Write a program where answers is of type List<String> or String []
   b. Write a program where answers is of type Set<String>
   c. Write a program where answers is of type Map<Integer, String>
      In case of Map, Integer value represents answer's sequence number.
   d. Create a Test class with main() method, get Question bean from ApplicationContext object and print question and its answers.
   e. Also write the JUnit Test cases for above program.

   - Use XML based configuration.

3) Example on autowiring

   Design and Develop a Banking Application as follows:

   a. Create a BankAccount class with following attributes: accountId, accountHolderName, accountType, accountBalance

   b. Create an interface BankAccountRepository with following methods:
      public double getBalance(long accountId)
      public double updateBalance(long accountId, double newBalance):
      Note: Above method returns updated balance.

Thank you for your Participation

kindly reach onkar.deshpande@Capgemini.com for queries.

Spring AOP

Capgemini

Table of Content

What you will learn

- recognize the concepts involved in Aspect-Oriented Programming
- recognize some of the benefits of Aspect-Oriented Programming
- become familiar with AOP terminology
- create a simple aspect in XML configuration in a Spring application
- create a simple aspect using annotations in a Spring application
- access information about the current JoinPoint (method execution) in advice body
- configure Spring AOP in a Spring application with a Java-based configuration
- understand Spring AOP's proxy-based approach to aspect-oriented programming
- recognize the different types of advice you can use when using Spring AOP
- recognize how pointcuts are declared
- declare and combine pointcuts
- implement aspects and advice in a Spring application, implement different types of pointcut designators supported by Spring AOP.
- You can use pointcut expressions to match method executions with specific return types, specific method names & specific method parameters.

Software requirement

- Java 8
- Spring Tool Suite (STS)
- Apache Tomcat 9.0
- Maven

Pre-requisites

- Core Java Knowledge is mandatory for this course.
- Spring Core knowledge is mandatory for this course.

Day wise Schedule

Day 1

- Spring Quick Start
- Why Aspect-oriented Programming (AOP)?
- Your First Aspect
- Advice Deep Dive
- Pointcut Deep Dive
- Expressing Architecture Using Pointcuts
- How Aspects are Added to Objects
- Spring AOP vs. Aspect
- Spring's Aspect Library
- Real Life Aspects

Course Links

Basic Course

Introduction to Spring AOP

https://www.baeldung.com/spring-aop

Spring AOP Example

https://www.journaldev.com/2583/spring-aop-example-tutorial-aspect-advice-pointcut-joinpoint-annotations

Aspect Oriented Programming and AOP in Spring Framework

https://www.geeksforgeeks.org/aspect-oriented-programming-and-aop-in-spring-framework/

Java Brain AOP Tutorial

https://www.youtube.com/watch?v=QdyLsX0nG30&list=PLE37064DE302862F8

Spring AOP Tutorial - with Aspectj Examples

https://www.youtube.com/watch?v=Og9Fyew8ltQ

Advance Course

Aspect Oriented Programming (AOP) using Spring AOP and AspectJ

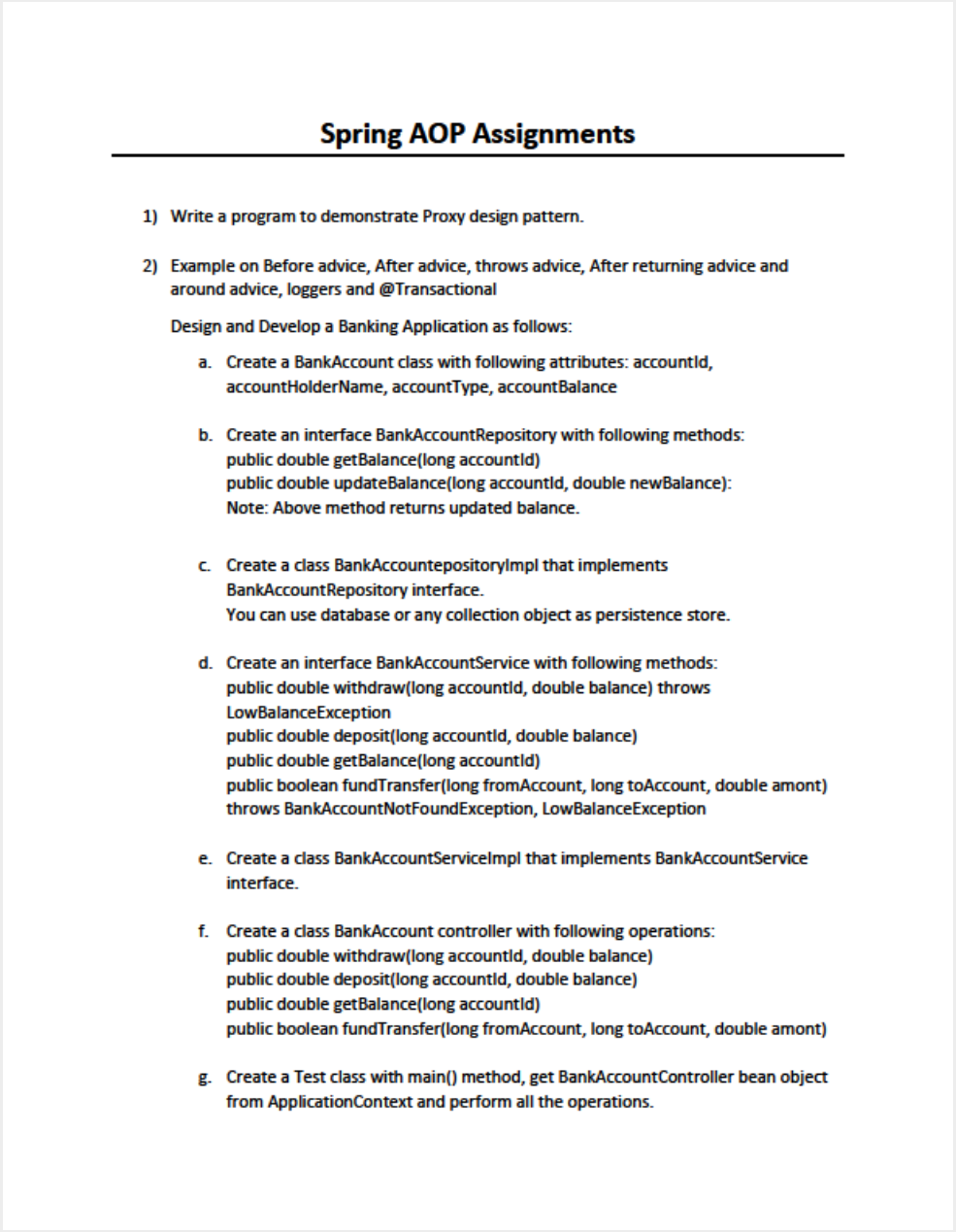
https://app.pluralsight.com/library/courses/aspect-oriented-programming-spring-aspectj/table-of-contents

Performance Monitoring with Spring AOP

https://www.javaworld.com/article/2073130/performance-monitoring-with-spring-aop.html

Assignment Basic to Advance

Please double click on image it will open the pdf



Spring AOP Assignments

1) Write a program to demonstrate Proxy design pattern.

2) Example on Before advice, After advice, throws advice, After returning advice and around advice, loggers and @Transactional

   Design and Develop a Banking Application as follows:

   a. Create a BankAccount class with following attributes: accountId, accountHolderName, accountType, accountBalance

   b. Create an interface BankAccountRepository with following methods:
   public double getBalance(long accountId)
   public double updateBalance(long accountId, double newBalance):
   Note: Above method returns updated balance.

   c. Create a class BankAccountepositoryImpl that implements BankAccountRepository interface.
   You can use database or any collection object as persistence store.

   d. Create an interface BankAccountService with following methods:
   public double withdraw(long accountId, double balance) throws LowBalanceException
   public double deposit(long accountId, double balance)
   public double getBalance(long accountId)
   public boolean fundTransfer(long fromAccount, long toAccount, double amont) throws BankAccountNotFoundException, LowBalanceException

   e. Create a class BankAccountServiceImpl that implements BankAccountService interface.

   f. Create a class BankAccount controller with following operations:
   public double withdraw(long accountId, double balance)
   public double deposit(long accountId, double balance)
   public double getBalance(long accountId)
   public boolean fundTransfer(long fromAccount, long toAccount, double amont)

   g. Create a Test class with main() method, get BankAccountController bean object from ApplicationContext and perform all the operations.

Thank you for your Participation

kindly reach onkar.deshpande@Capgemini.com for queries.

Spring JDBC

Capgemini

Table of Content

What you will learn

- This course covers integrating Spring JDBC into your application. You'll learn setup and configuration, PreparedStatements, RowMapper, NamedParameter, and their various approaches utilizing JdbcTemplate, SimpleJdbcInsert, and SimpleJdbcCall.
- Unit testing in Spring JDBC.
- To map Query Results to Java Object.

Software requirement

- Java 8
- Spring Tool Suite (STS)
- Apache Tomcat 9.0
- MySQL server OR Oracle 11g database
- Maven

Pre-requisites

- Core Java Knowledge is mandatory for this course.
- Spring Core knowledge is mandatory for this course.

Day wise Schedule

Day 1

- Introduction to Spring JDBC
- Setting up the JDBC Project
- Creating Records in the Database
- Reading Records from the Database
- Updating Records in the Database
- Deleting Records from the Database
- Exceptions and Transactions
- Unit testing in Spring JDBC

Course Links

Basic Course

The Spring Jdbc Template for database access
https://www.vogella.com/tutorials/SpringJDBC/article.html

Spring JDBC Example
https://www.journaldev.com/2593/spring-jdbc-example

Data access using JDBC
https://docs.spring.io/spring/docs/3.2.x/spring-framework-reference/html/jdbc.html

Spring JDBC Tutorial
https://www.youtube.com/watch?v=1WgsDP10iqw&t=14s

Building Applications Using Spring JDBC
https://app.pluralsight.com/library/courses/building-applications-spring-jdbc/table-of-contents

Advance Course

Test-driven Spring JDBC
https://htr3n.github.io/2018/11/test-driven-spring-jdbc/

Assignment Basic

Spring JDBC Assignments

Design model classes Employee and Department as follows

Employee class

Attributes	Type
employeeId	int
employeeName	String
employeeHireDate	LocalDate
employeeJobId	String
employeeSalary	double
employeeDepartment	Department

Department class

Attributes	Type
departmentId	int
departmentName	String
departmentManagerId	int

Note: Sample data is provided here. download

1) Write a program to display details of all the employees of a particular department. Department id is entered through keyboard.

2) Write a program to display employee details whose employee id is entered through keyboard.

3) Write a program to display name, job id and salary of employees whose jobs are either sales representative or stock clerk and whose salaries between 5000 and 13000 & sort the data in ascending order of employee names.

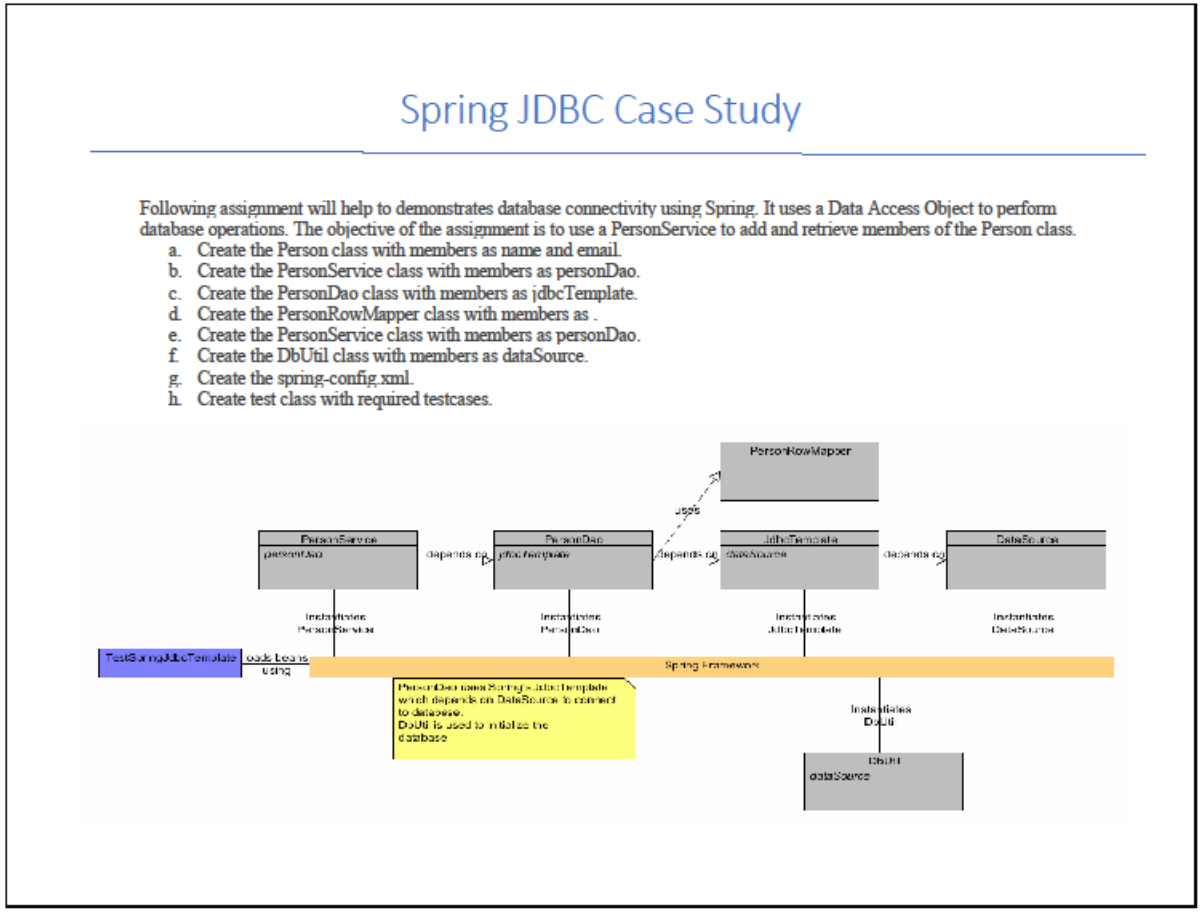
4) Write a program to display name and salary of an employee who has maximum salary in a particular department. Department id is entered through keyboard.

5) Write a program to display department ids and all employees who are working in those departments. Sort the result in ascending order of department id.

6) Write a program to display employee id and employee name along with his/her manager id and manager name.

7) Write a program to display id, name, salary and department name of all employees.

Case Study Advance



Spring JDBC Case Study

Following assignment will help to demonstrates database connectivity using Spring. It uses a Data Access Object to perform database operations. The objective of the assignment is to use a PersonService to add and retrieve members of the Person class.

a. Create the Person class with members as name and email.
b. Create the PersonService class with members as personDao.
c. Create the PersonDao class with members as jdbcTemplate.
d. Create the PersonRowMapper class with members as .
e. Create the PersonService class with members as personDao.
f. Create the DbUtil class with members as dataSource.
g. Create the spring-config.xml.
h. Create test class with required testcases.

Please double click on image it will open case study in pdf

Thank you for your Participation

kindly reach onkar.deshpande@Capgemini.com for queries.

Spring MVC

Capgemini

Table of Content

What you will learn

- This material provides you the details about Spring MVC architecture. You will understand how to create a web-mvc application by using controllers, models, views, handler mappings and view resolvers. You will understand how to use validation APIs to perform validations at server side.
- Implement Exception Handling using @ControllerAdvice and @ExceptionHandler.
- Implement Internationalization, Implement Interceptors.

Software requirement

- Java 8
- Spring Tool Suite (STS)
- Apache Tomcat 9.0
- Maven

Pre-requisites

- Core Java Knowledge is mandatory for this course.
- Spring Core knowledge is mandatory for this course.
- Basic knowledge of HTML, CSS & JavaScript will be useful.

Day wise Schedule

Day 1

- Spring MVC 4 Configuration
- Spring MVC 4 Architecture
- Spring MVC 4 Controllers
- Spring MVC 4 Views
- Spring MVC 4 Build Out
- Spring MVC 4 Validation
- Spring MVC 4 REST and Ajax
- What's new in Spring MVC 5

Course Links

Basic Course

Introduction to Spring MVC 4
https://app.pluralsight.com/library/courses/spring-mvc4-introduction/table-of-contents

Spring MVC Tutorial
https://www.baeldung.com/spring-mvc-tutorial

Spring MVC Tutorial in28Minutes
https://www.youtube.com/watch?v=BjNhGaZDr0Y

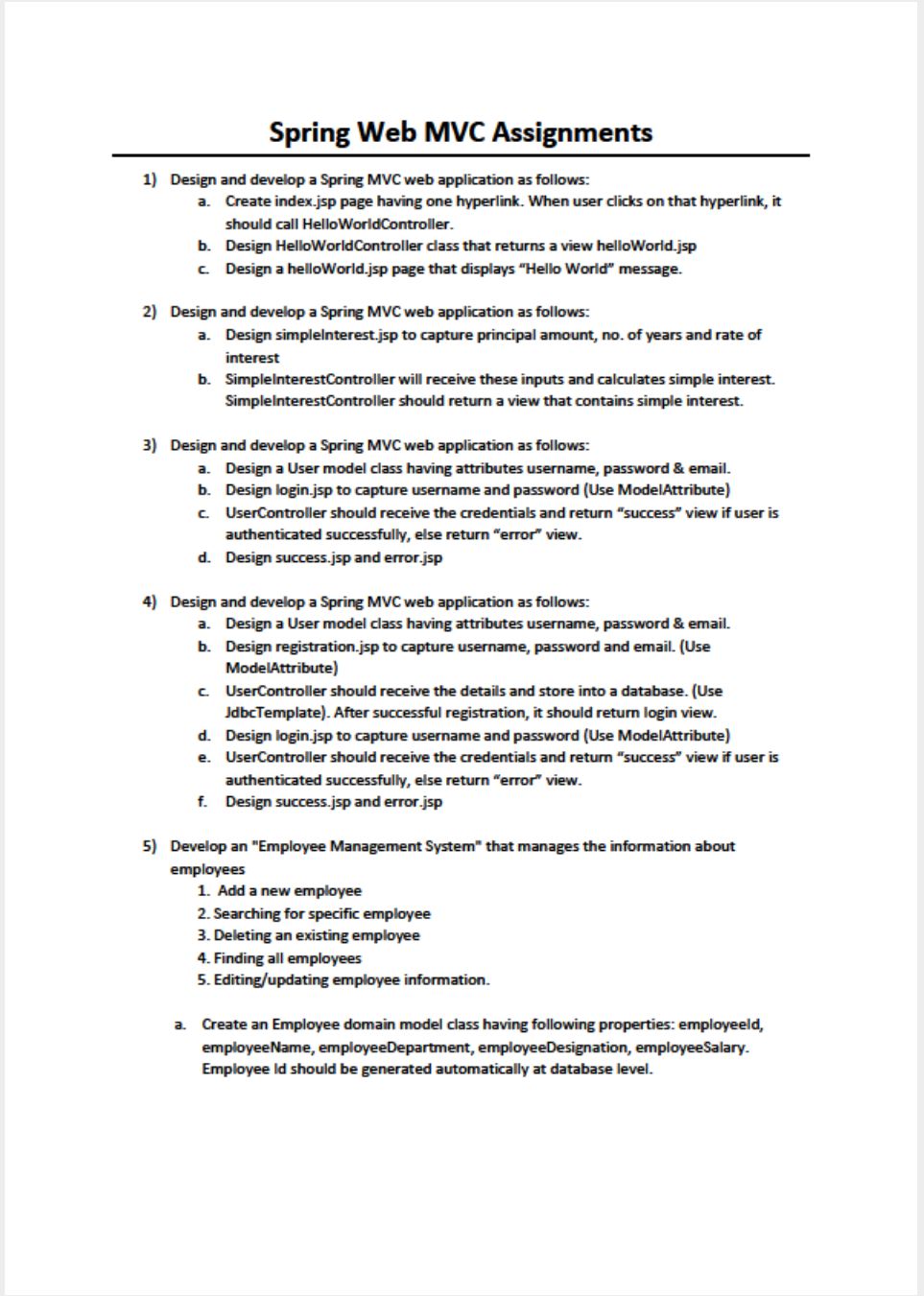
Advance Course

How Spring Web MVC Really Works
https://stackify.com/spring-mvc/

Mastering Spring framework 5
https://www.javaworld.com/article/2078034/spring-framework-mastering-spring-mvc.html

Advance to Basic Assignment

Please double click on image it will open the pdf



Spring Web MVC Assignments

1) Design and develop a Spring MVC web application as follows:
   a. Create index.jsp page having one hyperlink. When user clicks on that hyperlink, it should call HelloWorldController.
   b. Design HelloWorldController class that returns a view helloWorld.jsp
   c. Design a helloWorld.jsp page that displays "Hello World" message.

2) Design and develop a Spring MVC web application as follows:
   a. Design simpleInterest.jsp to capture principal amount, no. of years and rate of interest
   b. SimpleInterestController will receive these inputs and calculates simple interest. SimpleInterestController should return a view that contains simple interest.

3) Design and develop a Spring MVC web application as follows:
   a. Design a User model class having attributes username, password & email.
   b. Design login.jsp to capture username and password (Use ModelAttribute)
   c. UserController should receive the credentials and return "success" view if user is authenticated successfully, else return "error" view.
   d. Design success.jsp and error.jsp

4) Design and develop a Spring MVC web application as follows:
   a. Design a User model class having attributes username, password & email.
   b. Design registration.jsp to capture username, password and email. (Use ModelAttribute)
   c. UserController should receive the details and store into a database. (Use JdbcTemplate). After successful registration, it should return login view.
   d. Design login.jsp to capture username and password (Use ModelAttribute)
   e. UserController should receive the credentials and return "success" view if user is authenticated successfully, else return "error" view.
   f. Design success.jsp and error.jsp

5) Develop an "Employee Management System" that manages the information about employees
   1. Add a new employee
   2. Searching for specific employee
   3. Deleting an existing employee
   4. Finding all employees
   5. Editing/updating employee information.

   a. Create an Employee domain model class having following properties: employeeId, employeeName, employeeDepartment, employeeDesignation, employeeSalary. Employee Id should be generated automatically at database level.

Thank you for your Participation

kindly reach onkar.deshpande@Capgemini.com for queries.

Spring Security

Capgemini

Table of Content

What you will learn

- This material provides you the details about how to use spring security to secure your web application, how to use database authentication with custom login form and with different user roles to access specific URLs. You can implement remember me logic, with password encryption and limiting no. of login attempts.
- Configure other authentication features including remember-me, anonymous users, and logout.
- Fix authorization constraints over individual methods of service beans, in lieu of URL authorization or in tandem with it.
- Implement OAuth 2.0 authorization-server and resource-server roles.
- Implement application-specific authorization constraints as AccessDecisionVoters.
- Apply authorization constraints to URLs and URL patterns.
- Bind authorization roles to user accounts in relational databases.

Software requirement

- Java 8
- Spring Tool Suite (STS)
- Apache Tomcat 9.0
- Maven

Pre-requisites

- Core Java Knowledge is mandatory for this course.
- Spring Core knowledge is mandatory for this course.

Day wise Schedule

Day 1

- Getting the Scaffold Application
- Securing your Spring MVC Application
- User Storage in a Database
- Spring Security Client Integration
- Password Storage
- Customizing Spring Security
- Enabling Security with Expressions
- Authentication Using LDAP
- Forcing the use of HTTPS
- Introduction: Why Spring Security?
- Diving Under the Hood of Spring Security Authentication
- Dealing with Common Security Threats
- Securing User Credentials
- Adding Additional Layers for Authentication
- Persisting Access with Remember-Me
- Outsourcing Authentication with OpenID / OAuth2
- Layering Authorization with Spring Security

Course Links

Basic Course

Spring Security Fundamentals

https://app.pluralsight.com/library/courses/spring-security-fundamentals/table-of-contents

Security with Spring

https://www.baeldung.com/security-spring

Advance Course

Spring Security: Authentication / Authorization - Building Effective Layers of Defense

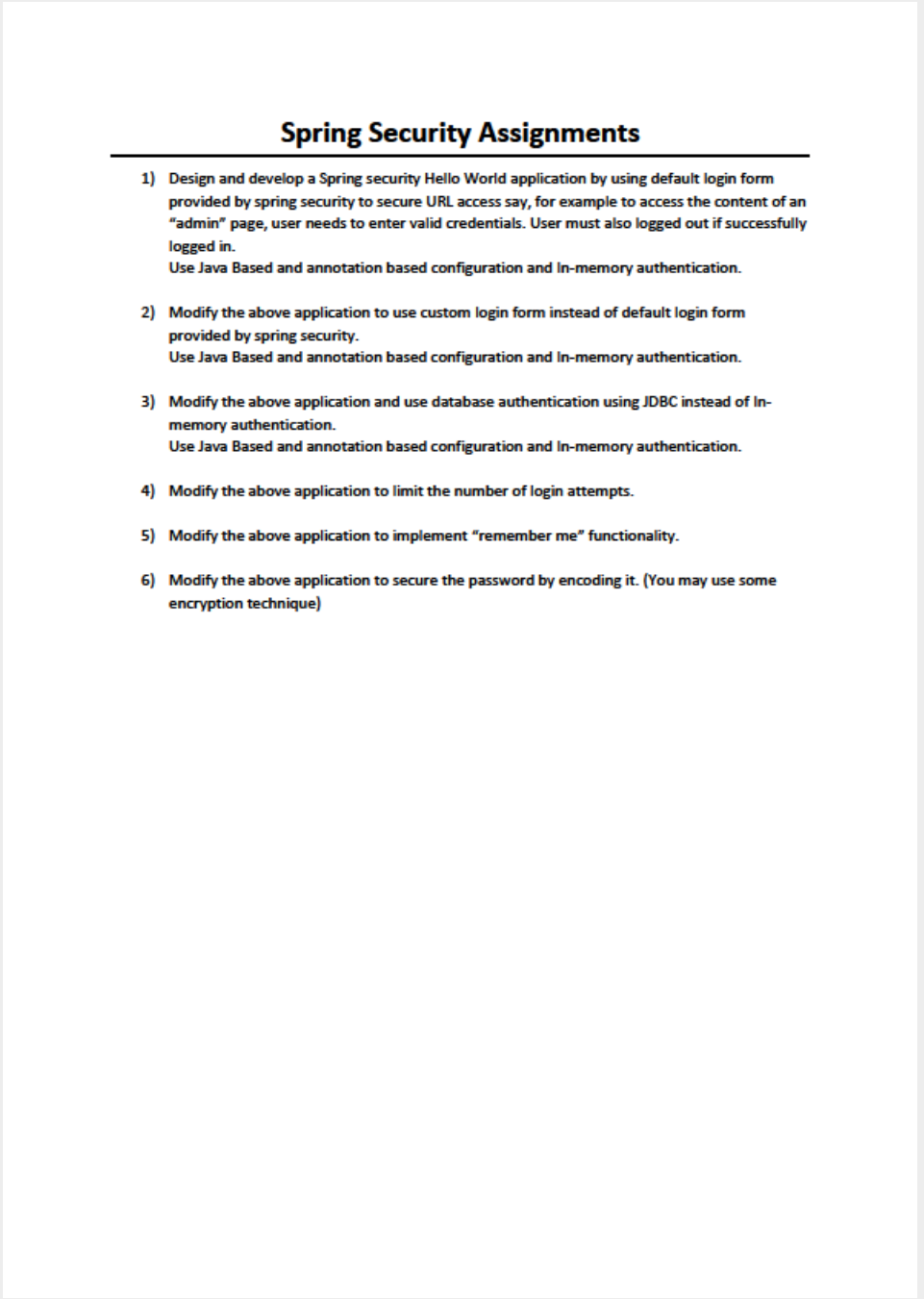
https://app.pluralsight.com/library/courses/spring-security-authentication-authorization-layers-of-defense/table-of-contents

Spring REST + Spring Security Example

https://www.mkyong.com/spring-boot/spring-rest-spring-security-example/

Basic Assignment

Please double click on image it will open the pdf



Spring Security Assignments

1) Design and develop a Spring security Hello World application by using default login form provided by spring security to secure URL access say, for example to access the content of an "admin" page, user needs to enter valid credentials. User must also logged out if successfully logged in.
   Use Java Based and annotation based configuration and In-memory authentication.

2) Modify the above application to use custom login form instead of default login form provided by spring security.
   Use Java Based and annotation based configuration and In-memory authentication.

3) Modify the above application and use database authentication using JDBC instead of In-memory authentication.
   Use Java Based and annotation based configuration and In-memory authentication.

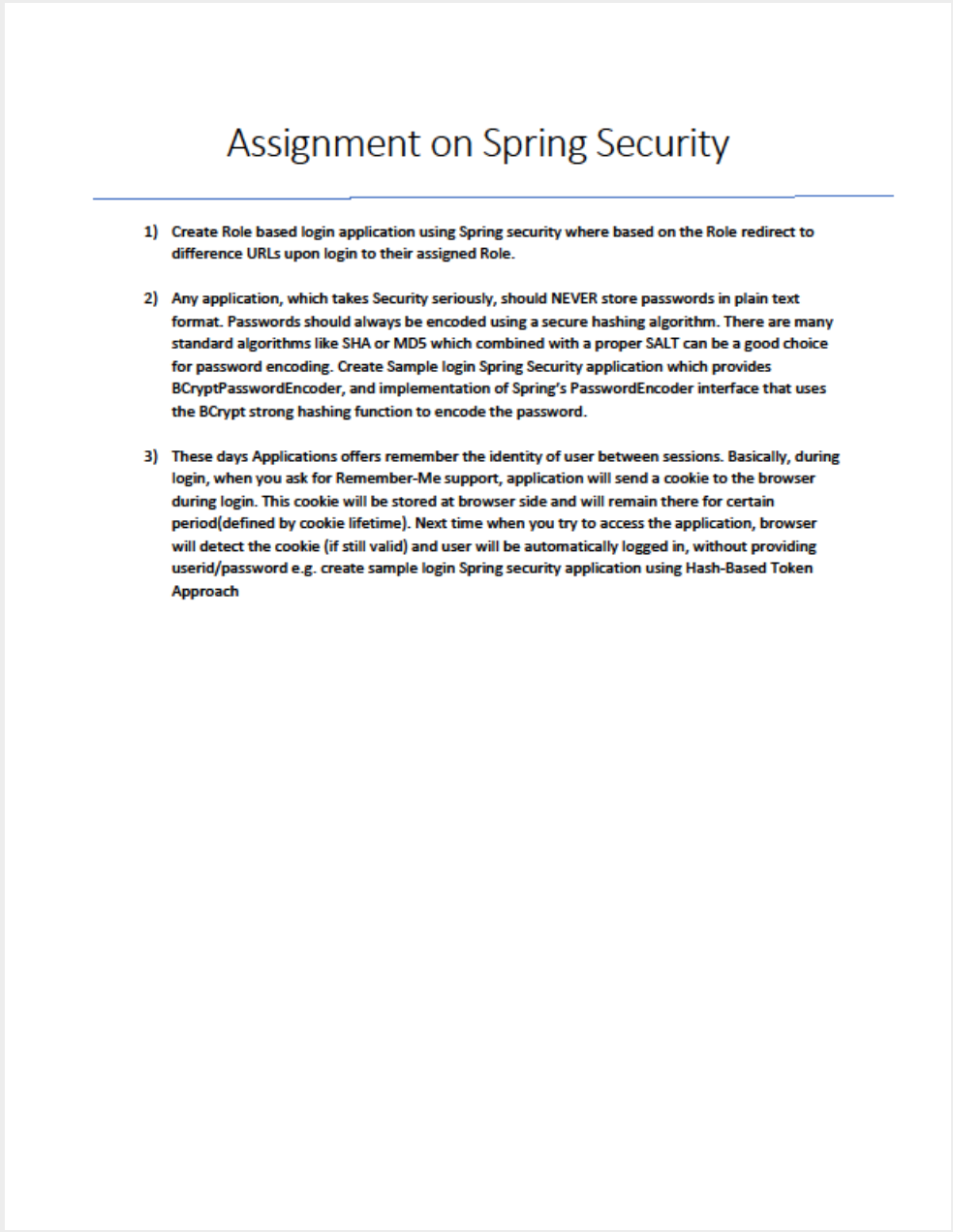
4) Modify the above application to limit the number of login attempts.

5) Modify the above application to implement "remember me" functionality.

6) Modify the above application to secure the password by encoding it. (You may use some encryption technique)

Advance Assignment

Please double click on image it will open the pdf



Assignment on Spring Security

1) Create Role based login application using Spring security where based on the Role redirect to difference URLs upon login to their assigned Role.

2) Any application, which takes Security seriously, should NEVER store passwords in plain text format. Passwords should always be encoded using a secure hashing algorithm. There are many standard algorithms like SHA or MD5 which combined with a proper SALT can be a good choice for password encoding. Create Sample login Spring Security application which provides BCryptPasswordEncoder, and implementation of Spring's PasswordEncoder interface that uses the BCrypt strong hashing function to encode the password.

3) These days Applications offers remember the identity of user between sessions. Basically, during login, when you ask for Remember-Me support, application will send a cookie to the browser during login. This cookie will be stored at browser side and will remain there for certain period(defined by cookie lifetime). Next time when you try to access the application, browser will detect the cookie (if still valid) and user will be automatically logged in, without providing userid/password e.g. create sample login Spring security application using Hash-Based Token Approach

Thank you for your Participation

kindly reach onkar.deshpande@Capgemini.com for queries.