**CS544**

**Enterprise Architecture Final**

## Exam 2 December 2016

Name\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Student ID \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**NOTE: This material is private and confidential. It is the property of MUM and is not to be disseminated.**

1. [15 points]Determine which of the following are TRUE/FALSE concerning REST :

T **F** RESTful web services are based on well-defined industry standards

EXPLAIN: REST is not about any standards. It is architectural style.

**T**  F An XML payload offers an advantage in a RESTful web service in that provides data validation

via XML schema definition.

EXPLAIN: refer to page #7 lesson 11(RESTfull )

T F RESTful operations map directly to HTTP Methods.

EXPLAIN:

T **F** Both a PUT and a POST REST operation modify data on the server. Therefore neither of them

are idempotent.

EXPLAIN:

**T** F REST security requires authenticating the user on every request.

EXPLAIN: Since REST is stateless it doesn’t hold any information.

1. [10 points] Consider the following AOP Aspect:

**AuditAspect.java**

@Pointcut("execution(\* edu.mum.service..\*(..))")

**public** **void** auditMethod() {}

@Pointcut("execution(\* edu.mum.service..list(Integer))")

**public** **void** auditMethodList() {}

@Pointcut("@annotation(edu.mum.validation.Audit)")

**public** **void** audit() {}

@Pointcut("args(Object)")

**public** **void** argsMethod(Object object) {}

@Before("auditMethod() && argsMethod(Object)")

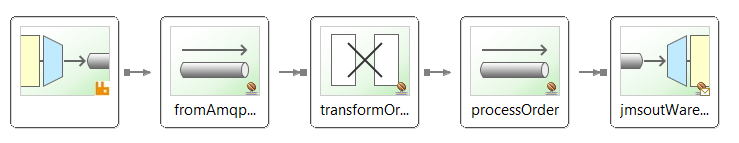
**public** **void** doAudit( Object object) **throws** Throwable {

1. **Implement example application code where the Advice method is applied.**

**Be sure to identify the package, the class & the join point.**

1. [15 points] Enterprise Integration Patterns [EIP] are a fundamental definition of how to do integration in a company of any significant size. Spring Integration implements those patterns.

Here is a diagram describing a basic ESB flow:



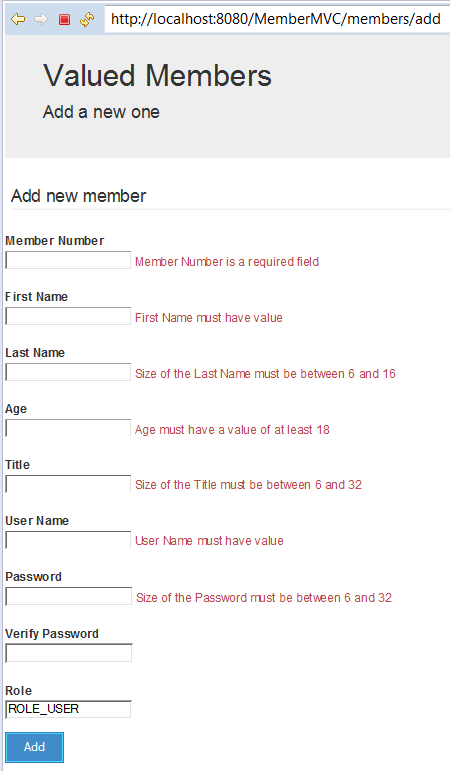
1. Itemize step by step what happens in this flow.

List 5 steps – one step per component

Identify the component and detail what it does.

If a step requires Java code to be implemented, give a code example

1. [15 Points] The following screen displays the member entry screen. Annotate The Member & User Credentials domain objects to reflect the Validation requirements. The provided Member Controller also needs to be annotated. Also the controller methods need to be completed as necessary [ including validation support].



**Here is the relevant part of the Member Domain Class:**

@Entity

**public** **class** Member {

@Id

@GeneratedValue(strategy=GenerationType.***AUTO***)

**private** **long** id;

@Column(length = 16)

**private** String firstName;

@Column(length = 16)

**private** String lastName;

**private** Integer age;

@Column(length = 32)

**private** String title;

**private** Integer memberNumber;

@OneToOne(fetch=FetchType.***EAGER***, cascade = CascadeType.***ALL***)

@JoinColumn(name="member\_id")

UserCredentials userCredentials;

**Here is the UserCredentials:**

@Entity(name = "Authentication")

**public** **class** UserCredentials {

@Id

@Column(name = "USER", nullable = **false**, unique = **true**, length = 127)

String userName;

@Column(name = "PASSWORD", nullable = **false**, length = 32)

String password;

@Column( nullable = **false**, length = 32)

String verifyPassword;

Boolean enabled;

@OneToOne(mappedBy="userCredentials", cascade = {CascadeType.***PERSIST***, CascadeType.***MERGE***})

**private** User user;

**ErrorMessage.properties**

**MemberController.java**

**public** **class** MemberController {

@Autowired

**private** MemberService memberService;

**public** String getAddNewMemberForm(@ModelAttribute("newMember") Member newMember) {

**return** "addMember";

}

**public** String processAddNewMemberForm(

memberService.save(memberToBeAdded);

**return** "redirect:/members";

}

1. [20 points]

Distributed Messaging is basic to scalable enterprise architectures. We covered two messaging technologies, JMS & AMQP. The following diagram represents, at a deployment level [network connected “machines”], possible messaging scenarios.

Each “box” is a different computer, representing messaging components.

****

Implement the following use case:

At close of business, every day, a report is sent asynchronously, from the Accounting department to everyone in the Sales Department; Managers, Supervisors, and Representatives.

1. Using JMS terminology, “modify” the above diagram to identify the components necessary to implement the appropriate messaging. Cross out [delete] components that are NOT necessary.
2. Give Step-by-Step pseudo-code definition of the configuration/setup [Both Sender[s] & Receivers].
3. Give details [code or pseudo code] concerning the implementation necessary to perform the operation. Specifically how the sender[s] send & how the receiver[s] receive.
4. [20 points] RBAC [Role Based Access Control] is widely used in Corporate Enterprises.

It has its limitations, in that it is ONLY Role based.

1. Explain the alternative Access Control Methodology, ABAC.
2. Be specific.
3. **Give an example [use case/scenario].**
4. Diagrams are excellent but be sure to explain them.
5. What are the Authorization features of Spring that accommodate an ABAC implementation?