

**BIRLA INSTITUTE OF TECHNOLOGY & SCIENCE, PILANI**

**WORK INTEGRATED LEARNING PROGRAMMES**

**COURSE HANDOUT**

**Part A: Content Design**

|  |  |
| --- | --- |
| **Course Title** | Service Oriented Computing |
| **Course No(s)** | SE ZG533 / CSI ZG533 |
| **Credit Units** | 4 |
| **Course Author** | Akshaya Ganesan |
| **Version No** | 2.0 |
| **Date** | Jan 2021 |

**Course Objectives:**

|  |  |
| --- | --- |
| **No** | **Course Objective** |
| **CO1** | Understand the need for Service oriented Architectures and its evolution |
| **CO2** | Understand the architecture of applications using service-oriented principles and design service oriented applications |
| **CO3** | Apply specific standards, protocols, and technologies to design and develop web services in a Service oriented application |
| **CO4** | Comprehend the ways to design secure , fault tolerant web services and implement them using the relevant technologies |

**Text Books:**

|  |  |
| --- | --- |
| T1 | SOA with REST: Principles, Patterns & Constraints for Building Enterprise Solutions with REST,by Thomas Erl, Benjamin Carlyle, Cesare Pautasso Raj Balasubramanian Prentice Hall (10 August 2012) |
| T2 | Restful Web services, Leonard Richardson and Sam Ruby, 1 st edition published by O'Reilly Media, May 2007 |

**References:**

|  |  |
| --- | --- |
| R1 | Service Oriented Computing: Semantics, Processes, Agents Munindar Singh & Michael Huhns, Wiley; 1st edition (26 November 2004) |
| R2 | RESTful Web APIs: Services for a Changing World Book by Leonard Richardson, 1st edition, O'Reilly Media, Sept 2013 |
| R3 | Building Microservices: Designing Fine-Grained Systems Book by Sam Newman, 1st edition, published by O'Reilly Media, Feb 2015 |
| R4 | Web Services Essentials by Ethan Cerami, Publisher: O′Reilly; 1st edition (28 February 2002) |
| R5 | Hands-On RESTful API Design Patterns and Best Practices by Harihara Subramanian, Pethuru Raj Publisher: Packt Publishing, January 2019 |
| R6 | The Design of Web APIs by Arnaud Lauret Published by Manning Publications; 1st edition (November 2019) |

**Modular Content Structure**

|  |  |
| --- | --- |
| **TOPICS** | **REFERENCES** |
| **Module 1: Introduction:**   * Evolution and Need for SOA * Monolithic architecture, * Distributed architecture and its Limitations * Service Oriented Architecture (SOA) * Goals and Benefits * Service and Service Capability * Service Provider and Consumer * Service contract * Characteristics of SOA * SOA Manifesto | T1 chapter 3  T2 Chapter 1  R1 Chapter 1, 5  <https://patterns.arcitura.com/soa-patterns/basics/soamanifesto/annotated>  <https://patterns.arcitura.com/soa-patterns/basics/serviceorientation/the_need_for_service_orientation> |
| **Module 2: Understanding the SOA Terminology**   * Service Orientation   + - Service-orientation design paradigm     - Service Autonomy * Service-Related Granularity   + Capability based granularity * Data Based granularity * Service Models and Layers * Service Inventory   + Service Registry   + Service Discovery * Service Description   + Interface definition Language   + Service Profiling * Service Composition * Composition Members and Controllers * Web services and SOA * Types of Web Service- SOAP, REST, gRPc | T1 Chapter 4  <https://patterns.arcitura.com/soa-patterns/basics/soamethodology/service_layers> |
| **Module 3: SOAP based Web services**   * Simple Object Access Protocol   + - Structure of SOAP message     - SOAP over HTTP   + Service Description with Web Services Description Language (WSDL)     - Anatomy of a WSDL document * Implementing Code–First and contract First Web Services.   + Service Registry with Universal Description, Discovery and Integration registry (UDDI) * Features of SOAP (WS\* specifications) * Web Services Protocol Stack | R1 Chapter 3, 4  R4 Chapter 3, 4, 5  Web Services Architecture  W3C Working Group Note 11 February 2004  [https://www.w3.org/TR/2004/NOTE-ws-arch-20040211/#relwwwrest](https://www.w3.org/TR/2004/NOTE-ws-arch-20040211/%23relwwwrest)  Cesare Pautasso, Olaf Zimmermann, Frank Leymann - RESTful Web Services vs. “Big” Web Services: Making the Right Architectural Decision |
| **Module 4: REST based services**   * Introduction to REST (Representational State Transfer) * REST architectural style * REST constraints   + Client-server   + Statelessness   + Cacheable   + Uniform interface   + Code on demand * Goals of the REST architectural style * Resources and Resource Representations   + Identifying Resources   + Designing a Resource representation * Uniform contract elements   + Uniform Resource Identifier   + HTTP methods   + Media Types   + Designing URIs * REST services Description Languages * Hypermedia and Application State * Serialization and Deserialization   + Handling representation formats(JSON, XML) * Service Contracts | T1 Chapter 5,6  T2 Chapter 4, 8  Richardson Maturity Model  <https://martinfowler.com/articles/richardsonMaturityModel.html>  Fielding, Roy Thomas (2000). "Chapter 5: Representational State Transfer (REST)". Architectural Styles and the Design of Network-based Software Architectures (Ph.D.). University of California, Irvine  <https://www.ics.uci.edu/~fielding/pubs/dissertation/rest_arch_style.htm> |
| **Module 5: Service Oriented Design with REST (Design of Services-REST)**   * Identifying services by analyzing the domain * Case Study * Design Principles - applying service-orientation principles to REST services * REST service contract design * Service Design with REST   + Interaction Design with HTTP(response codes, request methods)   + Metadata Design(Media Types, content negotiation)   + Representation Design(Message body format, Hypermedia Representation)   + Hypermedia and URI Templates | T1 Chapter 7 , 10  T2 Chapter 5, 6  R2 Chapter 3,4  CASE STUDY:   1. KIOSKETC CO 2. MIDWEST UNIVERSITY ASSOCIATION (MUA)   Case Study Reference: Case Study: T1 chapter 2 and Appendix A |
| **Module 6: Design of REST services and Management**   * API Design for REST Based Services   + Key requirements for the API   + API first Approach   + Design guidelines and Best Practices * Naming and Versioning of API   + Versioning using custom headers * API documentation   + Use of documentation tools- Swagger * API publishing Tools * Best Practices for effective API Management * API Management Tools | R2 Chapter 3  T2 Chapter 7  R5 Design Strategy, Guidelines, and Best Practices  R5 API Design Principles  R5 API Versioning  R6 Chapter 6 API documentation  A Practical Approach to API Design (2014) D. Keith Casey Jr. and James |
| **Module 7: Invocations and Communication between services**   * Service invocations   + Ajax Applications as REST Clients * Frameworks for REST Services- Django, Spring, Ruby on Rails * Synchronous and Asynchronous communication * Message Queues * Publish/subscribe * Event Driven Communication * Communication through API gateways   + Configuring APIs * Routing requests   + API gateway solutions | T2 chapter 2, 11, 12  R5 API Gateway  https://www.w3.org/TR/2004/NOTE-ws-arch-20040211/#relwwwrest  R5 API Gateway |
| **Module 8: Developing Secure Services**   * Controlling access to web services and methods * Implementing security using API gateway * Controlling API access using API gateway * Authentication   + HTTP Basic Authentication   + Using API keys   + Establishing secure message transmission with SSL/TLS   + OAuth * Authorization   + Role based Access control   + Attribute based Access control | R3 Chapter 9  R5 API Security |
| **Module 9: Service Composition**   * Choreography and Orchestration * Service composition with REST * REST service composition design considerations   + Synchronous and Asynchronous REST Service Composition   + Binding Between Composition Participants   + Dealing with Idempotent Service Activities * Pros and cons of composition * Service composition with API Gateway | T1 chapter 11 , 13  R3 Chapter 4 |
| **Module 10: Transaction management and Session Management**   * Limitations with REST for implementing transactions * Handling transactions among REST services * Session Management   + Client based session management   + Server based session management   + Using Tokens   + Use of Distributed Cache-Redis,Memcache | R1 Chapter 11  <https://sites.google.com/site/wagingguerillasoftware/rest-series/transactions-in-restful-services> |
| **Module 11: Fault tolerance and Monitoring**   * Creating multiple instances of service * Ensuring fault tolerance of services * Use API gateways to Manage Failovers * Throttling * Monitoring   + Performance metrics,   + API Metrics   + Service Level Objectives * Logging Reporting and analytics * API Monitoring Tools | R3 Chapter 8 |
| **Module 12: Service Deployment**   * On-premises deployment   + Release Plan Packaging Services * Cloud deployments * Manage API lifecycle Autoscaling * Fault tolerant deployments * SLA based TiersCloud Provider Services – Usage of services (security, scaling, monitoring, API gateway) | R3 Chapter 6 |
| **Module 13: Microservices Architecture**   * Introduction * Comparing Architecture Characteristics SOA vs Microservices * Services and Micro services * Pros and cons of Micro-services * Technologies used in Micro-services: Containers, Kubernetes, etc. | R3 Chapter 1,2 |

**Learning Outcomes:**

|  |  |
| --- | --- |
| **No** | **Learning Outcomes** |
| LO1 | Articulate benefits of service orientation and identify scenarios where SOA is applicable. |
| LO2 | Design and architect services to meet specific interface and QoS requirements. |
| LO3 | Apply specific standards, protocols, and technologies to build services and Deploy services in various Platforms |
| LO4 | Apply security features to the web services |
| LO5 | Articulate the difference between the SOA and microservices architecture |

**Part B: Contact Session Plan**

|  |  |
| --- | --- |
| **Academic Term** | Second Semester 2022-2023 |
| **Course Title** | Service Oriented Computing |
| **Course No** | CSI ZG533/SE ZG533 |
| **Lead Instructor** | Sanjay Joshi |

**Course Contents**

|  |  |  |
| --- | --- | --- |
| **Contact Session** | **Topics** | **References** |
| 1 | **Module 1: Introduction:**   * Evolution and Need for SOA * Monolithic architecture, * Distributed architecture and its Limitations * Service Oriented Architecture (SOA) * Goals and Benefits * Service and Service Capability * Service Provider and Consumer * Service contract * Characteristics of SOA * SOA Manifesto | T1 chapter 3  T2 Chapter 1  R1 Chapter 1, 5  <https://patterns.arcitura.com/soa-patterns/basics/soamanifesto/annotated>  <https://patterns.arcitura.com/soa-patterns/basics/serviceorientation/the_need_for_service_orientation> |
| 2 | **Module 2: Understanding the SOA Terminology**   * Service Orientation   + - Service-orientation design paradigm     - Service Autonomy * Service-Related Granularity   + Capability based granularity * Data Based granularity * Service Models and Layers * Service Inventory   + Service Registry   + Service Discovery * Service Description   + Interface definition Language   + Service Profiling * Service Composition * Composition Members and Controllers * Web services and SOA * Types of Web Service- SOAP, REST, gRPc | T1 Chapter 4  <https://patterns.arcitura.com/soa-patterns/basics/soamethodology/service_layers> |
| 3 | **Module 3: SOAP based Web services**   * Simple Object Access Protocol   + - Structure of SOAP message     - SOAP over HTTP   + Service Description with Web Services Description Language (WSDL)     - Anatomy of a WSDL document * Implementing Code–First and contract First Web Services.   + Service Registry with Universal Description, Discovery and Integration registry (UDDI) * Features of SOAP (WS\* specifications)   Web Services Protocol Stack | R1 Chapter 3, 4  R4 Chapter 3, 4, 5  Web Services Architecture  W3C Working Group Note 11 February 2004  [https://www.w3.org/TR/2004/NOTE-ws-arch-20040211/#relwwwrest](https://www.w3.org/TR/2004/NOTE-ws-arch-20040211/%23relwwwrest)  Cesare Pautasso, Olaf Zimmermann, Frank Leymann - RESTful Web Services vs. “Big” Web Services: Making the Right Architectural Decision |
| 4 | **Module 4: REST based services**   * Introduction to REST (Representational State Transfer) * REST architectural style * REST constraints   + Client-server   + Statelessness   + Cacheable   + Uniform interface   + Code on demand * Goals of the REST architectural style * Resources and Resource Representations   + Identifying Resources   + Designing a Resource representation | T1 Chapter 5,6  T2 Chapter 4, 8  Richardson Maturity Model  <https://martinfowler.com/articles/richardsonMaturityModel.html>  Fielding, Roy Thomas (2000). "Chapter 5: Representational State Transfer (REST)". Architectural Styles and the Design of Network-based Software Architectures (Ph.D.). University of California, Irvine  <https://www.ics.uci.edu/~fielding/pubs/dissertation/rest_arch_style.htm> |
| 5 | * Uniform contract elements   + Uniform Resource Identifier   + HTTP methods   + Media Types   + Designing URIs * REST services Description Languages * Hypermedia and Application State * Serialization and Deserialization   + Handling representation formats(JSON, XML) * Service Contracts | T1 Chapter 5,6  T2 Chapter 4, 8 |
| 6 | **Module 5: Service Oriented Design with REST (Design of Services-REST)**   * Identifying services by analyzing the domain * Case Study * Design Principles - applying service-orientation principles to REST services * Rest service contract design | T1 Chapter 7 , 10  T2 Chapter 5, 6  R2 Chapter 3,4  CASE STUDY:  1)KIOSKETC CO  2)MIDWEST UNIVERSITY ASSOCIATION (MUA)  Case Study Reference: T1 chapter 2 and Appendix A |
| 7 | * Service Design with REST   + Interaction Design with HTTP(response codes, request methods)   + Metadata Design(Media Types, content negotiation)   + Representation Design(Message body format, Hypermedia Representation)   + Hypermedia and URI Templates | T2 Chapter 5, 6  R2 Chapter 3,4 |
| 8 | **Module 6: Design of REST services and Management**   * API Design for REST Based Services   + Key requirements for the API   + API first Approach   + Design guidelines and Best Practices * Naming and Versioning of API   + Versioning using custom headers * API documentation   + Use of documentation tools- Swagger * API publishing Tools * Best Practices for effective API Management * API Management Tools | R2 Chapter 3  T2 Chapter 7  R5 Design Strategy, Guidelines, and Best Practices  R5 API Design Principles  R5 API Versioning  R6 Chapter 6 API documentation  A Practical Approach to API Design (2014) D. Keith Casey Jr. and James |
| 9 | **Module 7: Invocations and Communication between services**   * Service invocations   + Ajax Applications as REST Clients * Frameworks for REST Services- Django, Spring, Ruby on Rails * Synchronous and Asynchronous communication * Message Queues * Publish/subscribe * Event Driven Communication * Communication through API gateways   + Configuring APIs   + Routing requests   + API gateway solutions | T2 chapter 2, 11, 12  R5 API Gateway  https://www.w3.org/TR/2004/NOTE-ws-arch-20040211/#relwwwrest |
| 10 | **Module 8: Developing Secure Services**   * Controlling access to web services and methods * Implementing security using API gateway * Controlling API access using API gateway * Authentication   + HTTP Basic Authentication   + Using API keys   + Establishing secure message transmission with SSL/TLS   + OAuth * Authorization   + Role based Access control   Attribute based Access control | R3 Chapter 9  R5 API Security |
| 11 | **Module 9: Service Composition**   * Choreography and Orchestration * Service composition with REST * REST service composition design considerations   + Synchronous and Asynchronous REST Service Composition   + Binding Between Composition Participants   + Dealing with Idempotent Service Activities      * Pros and cons of composition * Service composition with API Gateway | T1 chapter 11 , 13  R3 Chapter 4 |
| 12 | **Module 10: Transaction management and Session Management**   * Limitations with REST for implementing transactions * Handling transactions among REST services * Session Management   + Client based session management   + Server based session management   + Using Tokens   + Use of Distributed Cache-Redis,Memcache | R1 Chapter 11  <https://sites.google.com/site/wagingguerillasoftware/rest-series/transactions-in-restful-services> |
| 13 | **Module 11: Fault tolerance and Monitoring**   * Creating multiple instances of service * Ensuring fault tolerance of services * Use API gateways to Manage Failovers * Throttling * Monitoring   + Performance metrics,   + API Metrics   + Service Level Objectives * Logging Reporting and analytics * API Monitoring Tools | R3 Chapter 8 |
| 14 | **Module 12: Service Deployment**   * On-premises deployment   + Release Plan Packaging Services * Cloud deployments * Manage API lifecycle Autoscaling * Fault tolerant deployments * SLA based TiersCloud Provider Services – Usage of services (security, scaling, monitoring, API gateway) | R3 Chapter 6 |
| 15 | **Module 13: Microservices Architecture**   * Introduction * Comparing Architecture Characteristics SOA vs Microservices * Services and Micro services * Pros and cons of Micro-services * Technologies used in Micro-services: Containers, Kubernetes, etc. | R3 Chapter 1,2 |
| 16 | Review session |  |

**Experiential Learning:**

**Labs Exercises**

1): Create a new SOAP web service to perform operations, interact with database

2): Create a new REST service to perform CRUD operations with the database, using JSON data format.

3): (Building a Web-Service Client) Create a REST service that monitors stock data real-time and consume that API from the web application (or windows client) and display the stock price on the page.

4) Deployment of the REST service on webserver (or on cloud). Explore option of enabling certificates, authentication and authorization of users.

5) Using API gateway to manage API lifecycle, publish, document, route requests, monitor & log services.

Technology Stack: For REST:  Python based Django REST Framework

* Web Client: Plain JavaScript- AJAX
* Web Server: NGINIX, APACHE
* API Gateway: NGINX API Gateway
* Cloud Providers: AWS

**Assignment:**

1. Design of an full-fledged application with Service Oriented Architecture. [A requirements document with  problem requirements explaining the functionality  to be given, based on which the design of the Application is to be done]

Examples:

* Moving an existing ecommerce customer to a service based environment
* Design an order processing system for e-commerce site like Amazon and Flipkart.
* Migrate a monolithic Simple Banking System to a Service based banking system. Come up with a component diagram and sequence diagram of how Withdrawl, Deposit and Transfer of Money from one account to another account can be carried out.

**Evaluation Scheme**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Evaluation Component** | **Name**  (Quiz, Lab, Project, Midterm exam, End semester exam, etc) | **Type**  (Open book, Closed book, Online, etc.) | **Weight** | **Duration** | **Day, Date, Session, Time** |
| **EC - 1** | Quiz 1 |  | 5% |  | February 13-23, 2023 |
|  | Lab 1 |  | 15% |  | March 20-30, 2023 |
|  | Assignment 1 |  | 10% |  | April 20-30, 2023 |
| **EC - 2** | Mid-term Exam | Open book | 30% | 2 hours | Friday, 10/03/2023 (AN) |
| **EC - 3** | End Semester Exam | Open book | 40% | 2 ½ hours | Friday, 19/05/2023 (AN) |

***Note*** *- Evaluation components can be tailored depending on the proposed model.*

**Important Information**

Syllabus for Mid-Semester Test (Open Book): Topics in Weeks 1-8 (1-18 Hours)

Syllabus for Comprehensive Exam (Open Book): All topics given in plan of study

Evaluation Guidelines:

1. EC-1 consists of either two Assignments or three Quizzes. Announcements regarding the same will be made in a timely manner.
2. For Closed Book tests: No books or reference material of any kind will be permitted. Laptops/Mobiles of any kind are not allowed. Exchange of any material is not allowed.
3. For Open Book exams: Use of prescribed and reference text books, in original (not photocopies) is permitted. Class notes/slides as reference material in filed or bound form is permitted. However, loose sheets of paper will not be allowed. Use of calculators is permitted in all exams. Laptops/Mobiles of any kind are not allowed. Exchange of any material is not allowed.
4. If a student is unable to appear for the Regular Test/Exam due to genuine exigencies, the student should follow the procedure to apply for the Make-Up Test/Exam. The genuineness of the reason for absence in the Regular Exam shall be assessed prior to giving permission to appear for the Make-up Exam. Make-Up Test/Exam will be conducted only at selected exam centres on the dates to be announced later.

It shall be the responsibility of the individual student to be regular in maintaining the self-study schedule as given in the course handout, attend the lectures, and take all the prescribed evaluation components such as Assignment/Quiz, Mid-Semester Test and Comprehensive Exam according to the evaluation scheme provided in the handout.

</DIV>