BIRLA INSTITUTE OF TECHNOLOGY & SCIENCE, PILANI

WORK INTEGRATED LEARNING PROGRAMMES

COURSE HANDOUT

Part A: Content Design

Course Title	**Scalable Services**
Course No(s)	**SE ZG583**
Credit Units	5
Course Author	Akanksha Bharadwaj
Version No	1.0
Date	Jan 2021

Course Objectives

No	Course Objective
CO1	Build competence to design, develop, implement and manage scalable information systems
CO2	Gain understanding of different techniques & tools for building and managing scalable services
CO3	Gain understanding of challenges and best practices in creating and managing scalable services

Text Book(s)

T1	Microservices patterns by Chris Richardson, Manning Publications 2018
T2	Microservices in action by Morgan Bruce & Paulo Pereira, Manning Publications 2018
T3	Building Microservices by Sam Newman, O'Reilly Media 2015

Reference Book(s) & other resources

R1	https://kubernetes.io/docs/concepts/

R2	The Art of Scalability: Scalable Web Architecture, Processes, and Organizations for the Modern Enterprise, Second Edition by Michael T. Fisher; Martin L. Abbott Published by Addison-Wesley Professional, 2015
R3	Scalability patterns by Microsoft Azure: https://docs.microsoft.com/en-us/azure/architecture/patterns/category/performance-scalability
R4	Scalability Patterns: Best Practices for Designing High Volume Websites by Chander Dhall

Modular Content Structure

1. **Getting to know Scalability:**
   - 1.1. Introduction to Performance, Consistency and availability
   - 1.2. What is scalability?
   - 1.3. Need for scalable architectures
   - 1.4. Principles of Scalability
   - 1.5. Guidelines for Building Highly Scalable Systems
   - 1.6. Architecturally scalable requirements
   - 1.7. Challenges for Scalability
   - 1.8. YouTube case Study

2. **Popular scaling approaches**
   - 2.1. Managing & processing high volumes of data
     - 2.1.1. Partitioning and sharding
     - 2.1.2. Distributed data (CAP theorem, NoSQL, HDFS)
     - 2.1.3. Distributed Processing (Map reduce, Spark)
   - 2.2. Managing high velocity data streams (Kafka)
     - 2.2.1. Video streaming: Netflix, YouTube, use of CDN
     - 2.2.2. Real time analytics: Credit card fraud detection
     - 2.2.3. Web conferencing: WebEx, Zoom
     - 2.2.4. Edge computing: IoT systems
   - 2.3. Managing high volume transactions
     - 2.3.1. Service Replicas & load balancing
     - 2.3.2. Minimizing event processing: Command Query Responsibility Segregation (CQRS)
     - 2.3.3. Asynchronous communication
     - 2.3.4. Caching techniques: Distributed cache, global cache
   - 2.4. Scalability features in the Cloud (AWS, Azure, Google)
     - 2.4.1. Auto-scaling
     - 2.4.2. Horizontal and vertical scaling
     - 2.4.3. Use of Load balancers
     - 2.4.4. Virtualization
     - 2.4.5. Serverless computing
   - 2.5. Best Practices for Achieving Scalability

3. **Microservices -** Introduction
   - 3.1. Challenges with Monolith applications
   - 3.2. Microservices architecture
   - 3.3. Advantages and disadvantages of Microservices
   - 3.4. Process & organization for microservices

4. **Decomposition strategies**
    4.1. Decomposition by business capability
    4.2. Decomposition by business domain
    4.3. Decomposition guidelines
    4.4. Obstacles to decomposing an application into services
    4.5. Defining service APIs

5. **Communication between Microservices**
    5.1. Inter-service communication
        5.1.1. Synchronous communication (REST, gRPC)
        5.1.2. Asynchronous communication
    5.2. Application boundary
        5.2.1. API gateway
        5.2.2. API design
        5.2.3. Creating and versioning APIs
        5.2.4. API security
        5.2.5. Backends for frontends
6. **Transaction management**
    6.1. Distributed transactions
    6.2. Implementation
    6.3. Challenges
    6.4. Solutions
    6.5. Sagas

7. **Building with pipelines**
    7.1. Continuous integration
    7.2. Tooling
    7.3. Repository patterns – Multi-repo, mono-repo

8. **Designing reliable microservices**
    8.1. Sources of failure, cascading failures
    8.2. Designing reliable communication: Retires, async. Comm., circuit breakers
    8.3. Maximizing service reliability: Load balancing, Rate limiting (Queues, Throttling)
    8.4. Service mesh

9. **Securing and Testing scalable services**
    9.1. Securing code and repositories
    9.2. Using Authentication
    9.3. Unit testing
    9.4. Integration testing
    9.5. Load testing

10. **Deploying Microservices**
    10.1. Service startup
    10.2. Running multiple instances
    10.3. Adding load balancer
    10.4. Service to host models
        10.4.1. Single Service Instance to Host

10.4.2. Multiple static Service Per Host
10.4.3. Multiple scheduled services per host
10.5. Deploying services without downtime: Canaries, Blue-Green, & rolling deploys
10.6. Deploying microservices using Serverless deployment

11. **Deployment with Containers**
    11.1. Introduction to containers
    11.2. Containerizing a service
    11.3. Deploying to a cluster

12. **Monitoring**
    12.1. Golden signals
    12.2. Types of metrics
    12.3. Recommended practices
    12.4. Monitoring with Prometheus & Grafana
    12.5. Collecting metrics
    12.6. Instrumenting
    12.7. Raising sensible & actionable alerts
    12.8. Using logs & traces
    12.9. Useful info in log entries
    12.10. Tools for logging: ELK, Fluentd
    12.11. Logging the right information
    12.12. Tracing interaction between services
    12.13. Visualizing traces

13. **Kubernetes**
    13.1. Dockers for CaaS
    13.2. What is Kubernetes
    13.3. Deployment of Microservices using Kubernetes
    13.4. Scalability in Kubernetes
    13.5. Security in Kubernetes
    13.6. CI/CD using Kubernetes
    13.7. Kubernetes Dashboard

Learning Outcomes:

No	Learning Outcomes
LO1	Understanding of different scenarios where scaling is needed
LO2	Understanding of different approaches to scaling
LO3	Understanding of microservices technology
LO4	Ability to design, develop and deploy microservices based applications
LO5	Understanding of ways to monitor and manage Microservices
LO6	Confident in using tools used in building scalable services

Part B: Contact Session Plan

Academic Term	First Semester 2021-2022
Course Title	Scalable Services
Course No	SE ZG583
Lead Instructor	Akanksha Bharadwaj

Course Contents

Contact Session	List of Topic Title (from content structure in Part A)	Topic # (from content structure in Part A)	Text/Ref Book/external resource
1	**Getting to know Scalability** ● Introduction to Performance, Consistency and availability ● What is scalability? ● Need for scalable architectures ● Principles of Scalability ● Guidelines for Building Highly Scalable Systems ● Architecturally scalable requirements ● Challenges for Scalability ● YouTube case Study	1	R2 and R4
2	**Popular scaling approaches** ● Managing & processing high volumes of data ● Partitioning and sharding ● Distributed data (CAP theorem, NoSQL, HDFS) ● Distributed Processing (Map reduce, Spark) ● Managing high velocity data streams (Kafka) ● Video streaming: Netflix, YouTube, use of CDN ● Real time analytics: Credit card fraud detection ● Web conferencing: WebEx, Zoom ● Edge computing: IoT systems	2.1 and 2.2	R2

3	**Popular scaling approaches** ● Managing high volume transactions ● Service Replicas & load balancing ● Minimizing event processing: Command Query Responsibility Segregation (CQRS) ● Asynchronous communication ● Caching techniques: Distributed cache, global cache ● Scalability features in the Cloud (AWS, Azure, Google) ● Auto-scaling ● Horizontal and vertical scaling ● Use of Load balancers ● Virtualization ● Serverless computing ● Best Practices for Achieving Scalability	2.3, 2.4 and 2.5	R2
4	**Microservices -** Introduction ● Challenges with Monolith applications ● Microservices architecture ● Netflix case study	3.1 and 3.2	T1
5	**Microservices -** Introduction ● Advantages and disadvantages of Microservices ● Process & organization for Microservices **Decomposition strategies** ● Decomposition by business capability ● Decomposition by business domain	3.3, 3.4, 4.1 and 4.2	T1
6	**Decomposition strategies** ● Decomposition guidelines ● Obstacles to decomposing an application into services ● Defining service APIs **Communication between Microservices** ● Inter-service communication	4.3, 4.4, 4.5 and 5.1	T1

	○ Synchronous communication (REST, gRPC) ○ Asynchronous communication		
7	**Communication between Microservices** ● Application boundary ○ API gateway ○ API design ○ Creating and versioning APIs ○ API security ○ Backends for frontends **Transaction management** ● Distributed transactions ● Implementation ● Challenges ● Solutions ● Sagas	5.2 and 6	T1
8	Review and discussion		
9	**Building with pipelines** ● Continuous integration ● Tooling ● Repository patterns – Multi-repo, mono-repo	7	T3
10	**Designing reliable Microservices** ● Sources of failure, cascading failures ● Designing reliable communication: Retires, async. Comm., circuit breakers ● Maximizing service reliability: Load balancing, Rate limiting (Queues, Throttling) ● Service mesh	8	T2
11	**Securing and Testing scalable services** ● Securing code and repositories ● Using Authetication ● Unit testing ● Integration testing	9	T1 and T3

	● Load testing		
12	**Deploying microservices** ○ Service startup ○ Running multiple instances ○ Adding load balancer ○ Service to host models ○ Single Service Instance to Host ○ Multiple static Service Per Host ○ Multiple scheduled services per host	10.1, 10.2, 10.3 and 10.4	T2
13	**Deploying microservices** ○ Deploying services without downtime: Canaries, Blue-Green, & rolling deploys ○ Deploying microservices using Serverless deployment **Deployment with Containers** ● Introduction to containers ● Containerizing a service ● Deploying to a cluster	10.5, 10.6 and 11	T2
14	**Monitoring** ● Golden signals ● Types of metrics ● Recommended practices ● Monitoring with Prometheus & Grafana ● Collecting metrics ● Instrumenting ● Raising sensible & actionable alerts ● Using logs & traces ● Useful info in log entries ● Tools for logging: ELK, Fluentd ● Logging the right information ● Tracing interaction between services ● Visualizing traces	12	T2
15	**Kubernetes** ● Dockers for CaaS ● What is Kubernetes	13	R1 and T3

	• Deployment of Microservices using Kubernetes • Scalability in Kubernetes • Security in Kubernetes • CI/CD using Kubernetes • Kubernetes Dashboard		
16	Review and demonstration of Microservices based application		

Case studies:

1. YouTube
2. Netflix
3. Amazon Prime
4. Google
5. Facebook
6. Twitter
7. Scaling an IoT based Systems

Sources for above case studies to be added

Labs:

1. Design and develop a Microservices based application
2. Add a Microservice to existing application and perform incremental deployment
3. Using MongoDB for large scale data
4. Configuring Minikube and running a local cluster
5. Creating a Kubernetes Cluster on AWS Cloud
6. All services running on single container Vs services running on separate containers
7. Deploying Microservices based application using Kubernetes
8. Exploring Auto-scaling, auto-recovery and other features of Kubernetes
9. Security in Kubernetes
10. CI/CD in Kubernetes
11. Understanding the Kubernetes Dashboard

Use of API Gateway & exploring its features
Using gRPC, message queues, and ReST for communication between microservices and comparing their performance & ease of development

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%		August 16-30, 2021

	Quiz 2		5%		September 16-30, 2021
	Lab (exploring different tools)		10%		To be announced
	Assignment (end to end app development)		10%		To be announced
EC – 2	Mid-term Exam	Open book	30%	2 hours	Saturday 25/09/2021 (FN) 10 AM – 12 Noon
EC – 3	End Semester Exam	Open book	40%	2hours	Saturday 13/11/2021 (FN) 10 AM – 12 Noon

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

Important Information

Syllabus for Mid-Semester Test (Open Book): SL No. 1 to 8
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 centers 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>