Velammal College of Engineering and Technology, Madurai
Department of Computer Science and Engineering
Innovative Teaching Methodology

Name of the course: Data Structures
Topic: Singly Linked List and its operations
Course Incharge: Mrs. K. Arunasakthi
Innovative Method: Role Play

A singly linked list can be imagined as a lively village where each citizen is a **node**, carrying some data and a **pointer** that tells who comes next. At the entrance of this village stands **Head**, who only knows the first citizen.

For example, Head points to Alice, Alice points to Bob, Bob points to Carol, and Carol finally points to **null**, marking the end of the village. When a new citizen arrives and wants to join **at the beginning**, Head simply shifts and points to this new citizen, who in turn points to the previous first citizen. If someone wants to join **at the end**, the last node that currently points to null changes its pointer to the newcomer. When inserting **in the middle**, such as placing a new node after Bob, Bob redirects his pointer to the new citizen, who then points to Bob's old successor. Sometimes citizens leave the village too; if the first node leaves, Head simply points to the next one. If the last node leaves, the second-last citizen changes their pointer back to null. Removing someone from the middle involves the previous node skipping over the departing citizen and pointing directly to the next one. To visit the entire village, we start at Head and follow each pointer until we reach null—this is called **traversal**. If we are searching for a specific citizen, we visit each node one by one until we find the one whose data matches what we are looking for. Through this animated village scenario, the operations of insertion, deletion, traversal, and searching in a singly linked list become easy to visualize and understand.

Name of the course: Object Oriented Software Engineering
Topic: SRS Document
Course Incharge: Mrs. Niranjana
Innovative Method: BrainStorming

Experiential Learning for SRS preparation involves engaging students directly in real or simulated project environments where they learn by doing rather than only reading about requirements. Instead of passively studying SRS templates, learners take on roles such as clients, analysts, developers, and testers to experience the full cycle of gathering, analyzing, and documenting requirements. They interact with mock stakeholders, conduct interviews, observe workflows, identify user needs, and translate them into functional and non-functional requirements.

Through hands-on activities like creating use-case models, drafting requirement statements, managing requirement changes, and validating them with stakeholders, students gain a deeper understanding of how an effective SRS is produced. This practical involvement helps them understand challenges such as ambiguity, missing information, conflicting needs, and communication gaps—issues that are difficult to grasp through theory alone. Experiential Learning makes SRS preparation realistic, collaborative, and reflective, helping students build confidence and competence in creating high-quality requirement documents.

Name of the course: Distributed Systems
Topic: Activity based Learning
Course Incharge: S.Kavitha
Innovative Method: Role Play

Activity-Based Learning in distributed systems can be visualized as a role-play where students become different components of a large network—one acts as a **server**, another as a **client**, others as **replicas**, **message routers**, or **faulty nodes**—all working together to simulate how a distributed system functions in real life.

As they pass messages, handle requests, detect failures, or coordinate tasks, students experience firsthand how concepts like **communication protocols**, **synchronization**, **consistency**, **load balancing**, and **fault tolerance** actually play out in a live system. The teacher becomes the system coordinator, giving tasks such as simulating a node crash, initiating a leader election, or distributing work among participants. Through this interactive role-play, students learn how distributed components must cooperate despite delays, failures, or mismatched data. This activity-based approach turns abstract distributed system theories into an engaging, hands-on experience where learners understand complex behaviors by acting them out, making the concepts clearer, memorable, and fun.

Name of the course: Android App Development
Topic: Content Providers
Course Incharge: Mr.S.Murali
Innovative Method: Problem based Experiential Learning

Problem-Based Experiential Learning methodology encourages content providers to design learning materials around real-world challenges that require learners to actively explore, analyze, and solve problems. Instead of presenting information first, content providers introduce an authentic problem scenario and guide learners to investigate, research, collaborate, and experiment to find solutions.

Through this approach, learners experience the subject matter as it unfolds in actual practice—collecting data, evaluating alternatives, making decisions, and reflecting on outcomes—while content providers curate resources, scenarios, and hands-on activities that support discovery and critical thinking. This methodology transforms content into interactive experiences where learners construct knowledge through action, rather than memorization. It also empowers content providers to integrate case studies, simulations, field tasks, role-plays, and iterative feedback cycles that mirror industry challenges. As a result, learning becomes deeply engaging, applicable, and meaningful, preparing learners with both conceptual understanding and practical problem-solving skills.

Name of the course: Exploratory Data Analysis
Topic: Boxplot, Histogram, Barplot, Scatterplot
Course Incharge: Mrs.S.Sangeetha
Innovative Method: Flipped Classroom

In a filled-in classroom session on data visualization, students are introduced to four important plot types: **Boxplot, Histogram, Barplot, and Scatterplot**. The class begins with the **Boxplot**, which helps students understand how data is spread using the median, quartiles, and outliers, making it useful for comparing distributions.

Next, they explore the **Histogram**, where data is grouped into bins to show the frequency of values, helping learners visualize the shape of the distribution such as whether it is skewed or symmetric. The **Barplot** is then introduced as an effective way to compare categorical data, where each bar represents a category and its height reflects its value or frequency. Finally, the **Scatterplot** is used to show the relationship between two numerical variables, helping students observe patterns, trends, or correlations. Throughout the classroom activity, students fill in worksheets, analyze sample data, draw each type of plot, and discuss what insights each visualization provides, making the learning interactive and hands-on.