USN | | | | | | | | | | |

Internal Assessment Test 3 – June 2023

Sub:	Cloud Computing and Applications					Sub Code:	18CS643	Branch :	CSE
Date:	4.06.2023	Duration:	90 mins	Max Marks:	50	Sem / Sec:	6 A,B,C		OBE

Answer any FIVE FULL Questions

		MARKS	CO	RBT
1 (a)	How does Aneka threads differ from local thread? Explain with diagram.	9 M	CO2	L2

- Local threads belong to the **hosting process.**
- To create a thread, only necessary to provide a **pointer** to a method
- Aneka threads live in the **context** of a **distributed application**
- **multiple distributed applications** can be managed within a single process
- **thread creation** also requires the specification of the reference to the application to which the thread belongs

.Net Threading API	Aneka Threading API
System.Threading	Aneka.Threading
Thread	AnekaThread
Thread.ManagedThreadId (int)	AnekaThread.Id (string)
Thread.Name	AnekaThread.Name
Thread.ThreadState (ThreadState)	AnekaThread.State
Thread.IsAlive	AnekaThread.IsAlive
Thread.IsRunning	AnekaThread.IsRunning
Thread.IsBackground	AnekaThread.IsBackground[false]
Thread.Priority	AnekaThread.Priority[ThreadPriority.Normal]
Thread.IsThreadPoolThread	AnekaThread.IsThreadPoolThread [false]
Thread.Start	AnekaThread.Start
Thread.Abort	AnekaThread.Abort
Thread.Sleep	[Not provided]
Thread.Interrupt	[Not provided]
Thread.Suspend	[Not provided]
Thread.Resume	[Not provided]
Thread.Join	AnekaThread.Join

Aneka threads live and execute in a **distributed environment**, so their lifecycle is different from life cycle of threads.

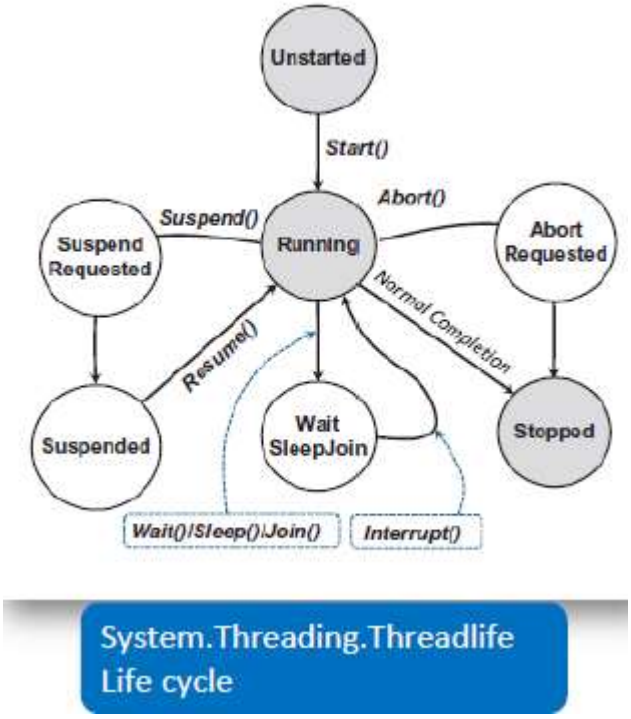
- Not possible to **map state value** of local threads to Aneka threads.
- **Local threads** : most of the state transitions are triggered by the developer

- In Aneka : triggered by the **Aneka Middleware**

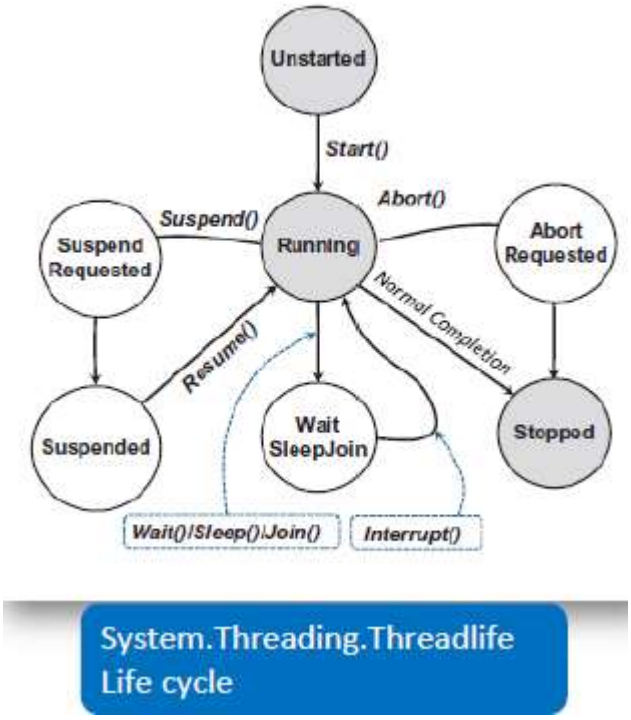
- **Aneka threads** : has more states

- they support **file staging**, scheduled by the middleware, which can queue them for a period of time

- They support **reservation of nodes** : state indicating execution failure due to missing reservation credentials.



System.Threading.Threadlife
Life cycle



System.Threading.Threadlife
Life cycle

Unstarted

- On invoking Start() method moves to ***Started*** state
- Moves to ***StagingIn*** state (if there are files to upload for execution) or ***Queued*** state
- If there is an error, goes to ***Failed*** state.
- ***Rejected*** State is reached for invalid reservation token
- From the ***Queued*** state, if there is a free node, it moves to the ***Running*** state
- If thread generates an exception, moves to ***Failed*** state
- On successful completion, moves to the ***Completed*** state
- If output files are yet to be retrieved, it moves to ***StagingOut*** state
- If developer directly calls Abort() method, goes into the ***Abort*** state.

Unstarted-[Started]-[Queued]-Running-Completed/Aborted/Failed

Thread Synchronization

.NET base class: provides **monitors, semaphores, reader-writer locks, and basic synchronization constructs** at language level

- **Aneka** provides **minimal support for thread synchronization**: limited to **join** operation.
- .NET provides more **stringent access** to shared data.
- This is not as strict in distributed environments, where there is **no shared memory** among threads.
- Providing locking facility in distributed environment leads to **distributed deadlocks** that are hard to detect.

Thread Priorities

System.Threading.Thread

- ThreadPriority enumeration: **Highest, AboveNormal, Normal, BelowNormal, or Lowest**.
- Aneka **does not support thread priorities**
- But for interface compatibility purposes, it has
- Aneka.Threading.Thread
- Priority property - **ThreadPriority** is always set to **normal**

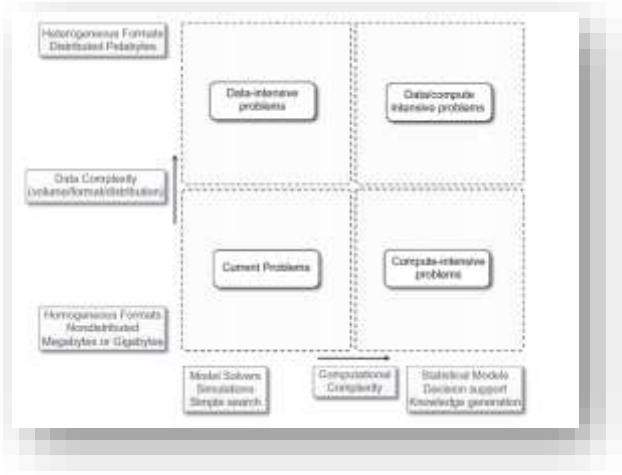
Type Serialization

	Aneka threads execute in a distributed environment in which the **object code in the form of libraries** and live instances information are moved over the network ►**Local threads** execute all within the same address space and share memory – they do not need objects to be copied or transferred into a different address space. ►In Aneka threads : **state of enclosing instance** needs to be transferred and reconstructed in the remote machine : at class level this is called ***type serialization***			

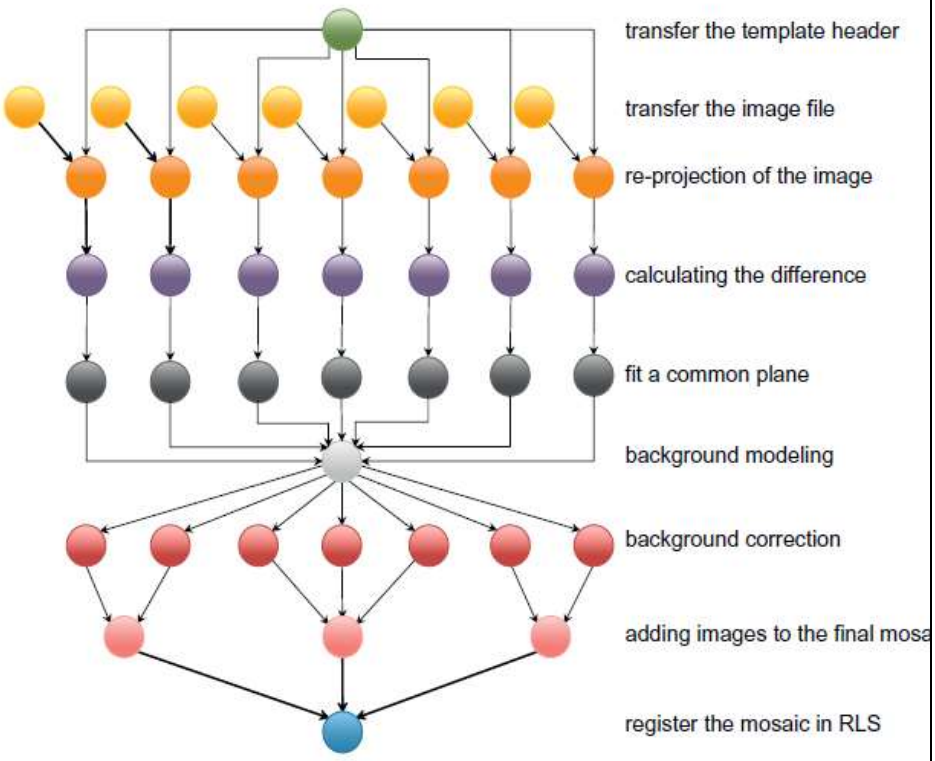
(b)	In the following code snippet, what is the use of [Serializable]?
	[Serializable]
	public class Cosine(double x) {
	double x;
	public double Result;
	
	public Cosine(double x) {
	this.x = x;
	}
	public void CosineCompute() {
	result = System.math.cos(x * System.Math.PI/180);
	}
	}
	
	means it is possible to convert an instance of the type into a binary array.
		1M	CO2	L2

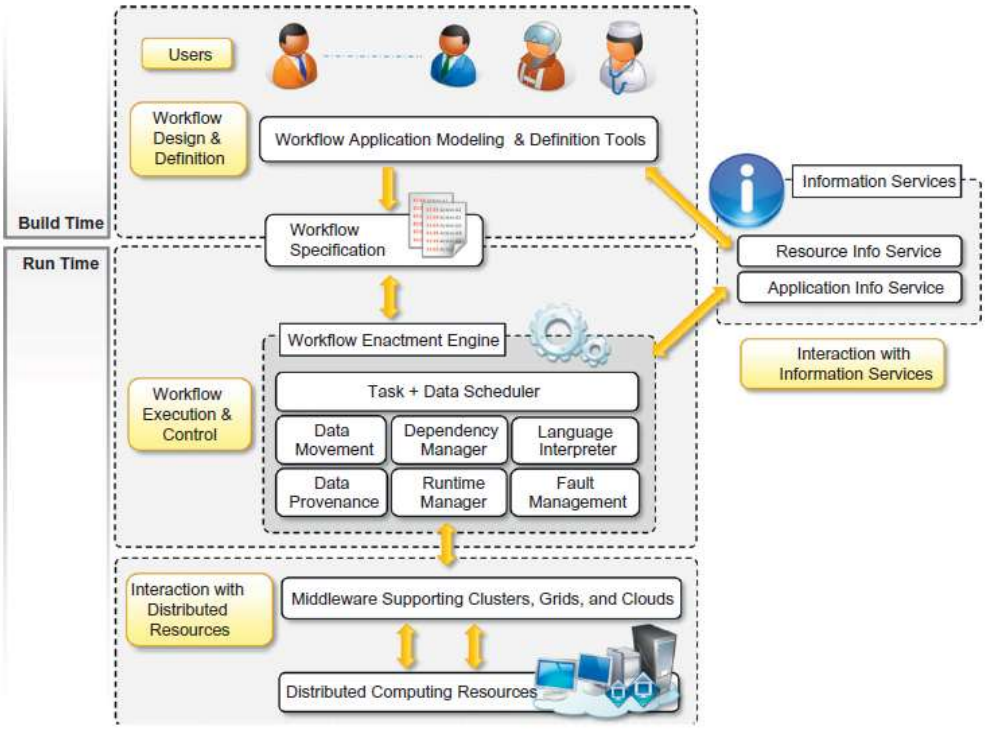
2 (a)	Explain Google File System with a neat diagram GoogleFileSystem(GFS). GFS [54]is the storage infrastructure that supports the execution of distributed applications in Google's computing cloud. The system has been designed to be a fault- tolerant, highly available, distributed file system built on commodity hardware and standard Linux operating systems. Rather than a generic implementation of a distributed file system, GFS specifically addresses Google's needs in terms of distributed storage for applications, and it has been designed with the following assumptions: • The system is built on top of commodity hardware that often fails. • The system stores a modest number of large files; multi-GB files are common and should be treated efficiently, and small files must be supported, but there is no need to optimize for that. • The workloads primarily consist of two kinds of reads: large streaming reads and small random reads. • The workloads also have many large, sequential writes that append data to files. • High-sustained bandwidth is more important than low latency. The architecture of the file system is organized into a single master, which contains the metadata of the entire file system, and a collection of chunk servers, which provide storage space. From a logical	5M	CO2	L2

	Point of view the system is composed of a collection of software daemons, which implement either the master server or the chunk server. A file is a collection of chunks for which the size can be configured at file system level. Chunks are replicated on multiple nodes in order to tolerate failures. Clients lookup the master server and identify the specific chunk of a file they want to access. Once the chunk is identified, the interaction happens between the client and the chunk server. Applications interact through the filesystem with a specific interface supporting the usual operations for file creation, deletion, read, and write. The interface also supports snapshots and record append operations that are frequently performed by applications. GFS has been conceived by considering that failures in a large distributed infrastructure are common rather than a rarity; therefore, specific attention has been given to implementing a highly available, lightweight, and fault-tolerant infrastructure. The potential single point of failure of the single-master architecture has been addressed bygivingthepossibilityofreplicatingthemasternodeonanyothernode belongingtotheinfrastructure.Moreover,astatelessdaemonandextensiveloggingcapabilities facilitatethesystem'srecoveryfromfailures.			
(b)	Explain Amazon S3 [OR] Apache Cassandra. ► is a distributed object store for managing large amounts of **structured data** spread across many **commodity servers** ► The system is designed to **avoid a single point of failure** ► offer a highly **reliable service** ► initially developed by **Facebook** ► It is now part of the Apache incubator initiative ► Used by **Facebook, Digg, and Twitter** ► 2nd gen distributed database that builds on concept of ► **AmazonDynamo** : fully distributed ► **Google BigTable** : inherits the column family concept ► The **data model** exposed by Cassandra is based on the concept of a table that is implemented as a distributed **multidimensional map** indexed by a **key** ► value corresponding to a key : **highly structured object** and constitutes the **row of a table** ► Cassandra organizes the **row of a table into columns** ► sets of columns can be grouped **into column families** ► APIs : insertion, retrieval, and deletion ► **Insertion : at row level** ► **Retrieval and delete :** at column level. Amazon S3 ► Amazon S3 is the online storage service provided by **Amazon** ► the system is claimed to support high availability, reliability, scalability, infinite storage, and low latency at commodity cost	5M	CO2	L2

	► offers a flat storage space organized into buckets, which are attached to an **Amazon Web Services (AWS)** account ► Each **bucket** can store multiple objects, each identified by a unique key ► Objects are identified by **unique URLs and exposed through HTTP** ► Allows very simple **get-put** semantics ► Because of use of **HTTP**, there is no need for any specific library for accessing the storage system ► a **POSIX-like client library** has been developed to **mount S3 buckets** as part of the local file system. ► Since the buckets are linked to AWS accounts, the owner of a bucket can decide to make it visible to other accounts or the public			
3 (a)	What is data intensive computing? List the challenges in data-intensive computing and explain each one in detail. **Concerned with the production, manipulation and analysis of large-scale data in the range of hundreds of megabytes to petabytes of data.**  ► Data-intensive applications not only deal with **huge volumes** of data but, very often, also exhibit **compute-intensive** properties ► Datasets are commonly persisted in **several formats** and **distributed** across different locations. ► Processing requirements **scale linearly** with the data size and they can be easily processed parallel. ► **Scalable algorithms** that can **search and process** massive datasets ► New **metadata management** technologies that can scale to handle **complex**, **heterogeneous,** and **distributed** data sources ► Advances in high-performance computing platforms aimed at providing a better support for **accessing in-memory multiterabyte data structures** ► **High-performance**, **highly reliable**, **petascale distributed file systems** ► **Data signature-generation** techniques for data reduction and rapid processing ► New approaches to **software mobility** for delivering algorithms that are able to move the computation to where the data are located	9M	CO2	L2

	► Specialized **hybrid interconnection architectures** that provide better support for **filtering multigigabyte datastreams** coming from high- speed networks and scientific instruments ► Flexible and high-performance software integration techniques.			
(b)	According to the Backblaze stats for Q1 2023, "The annualized failure rate (AFR) for hard drives has increased over the last three years. For example, Backblaze recorded an AFR of 0.93% in 2020 and 1.10% in 2021. The AFR for 2022 was up to 1.37%. The main reason for the increase is due to the age of the drives." How is this challenge of disk failure addressed in data-intensive computing? Data replication, checksum	1M	CO2	L3

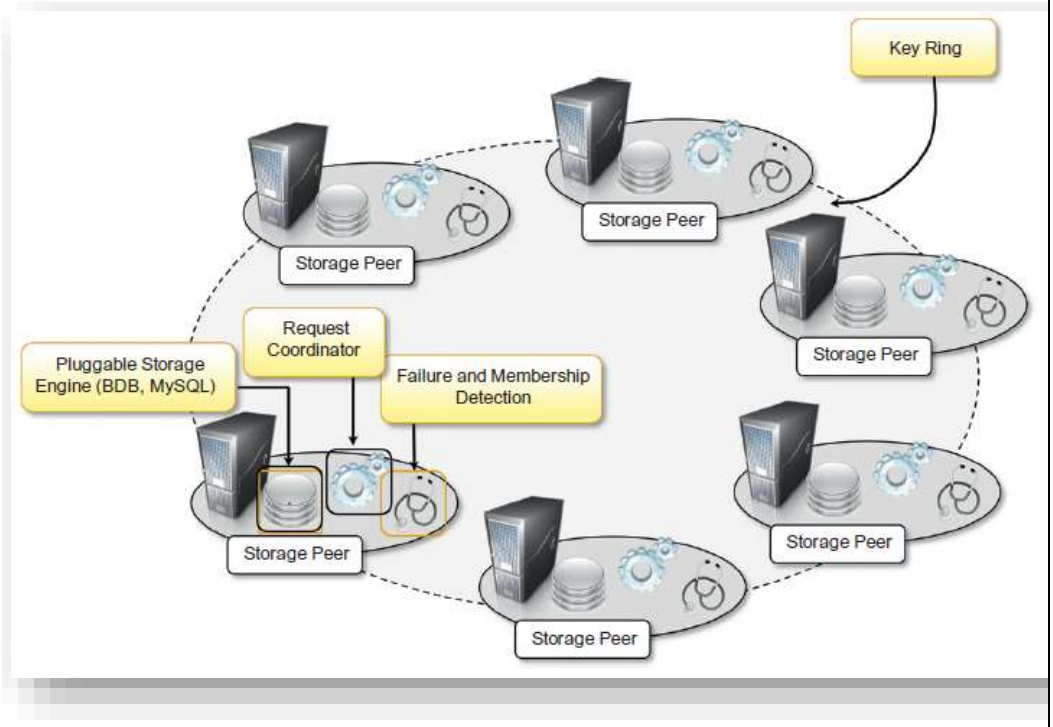
4 (a)	What is a workflow? ► ***An automation of a business process, in whole or in part, during which, documents, information, or tasks are passed from one participant (a resource, human or machine) to another for action, according to a set of procedural rules.*** ► DAG(Directed Acyclic Graph)  ► ►	5M	CO2	L2
(b)	Explain workflow technologies for designing and extracting workflow based technologies.	5M	CO2	L2



- ► design tools allow users to visually compose a workflow application
- ► normally stored in the form of an XML document
- ► controls the execution of the workflow by leveraging a distributed infrastructure
- ► Some frameworks can natively support the execution of workflow applications

- ► Kepler
- ► DAGMan
- ► Cloudbus WfMS
- ► Offspring

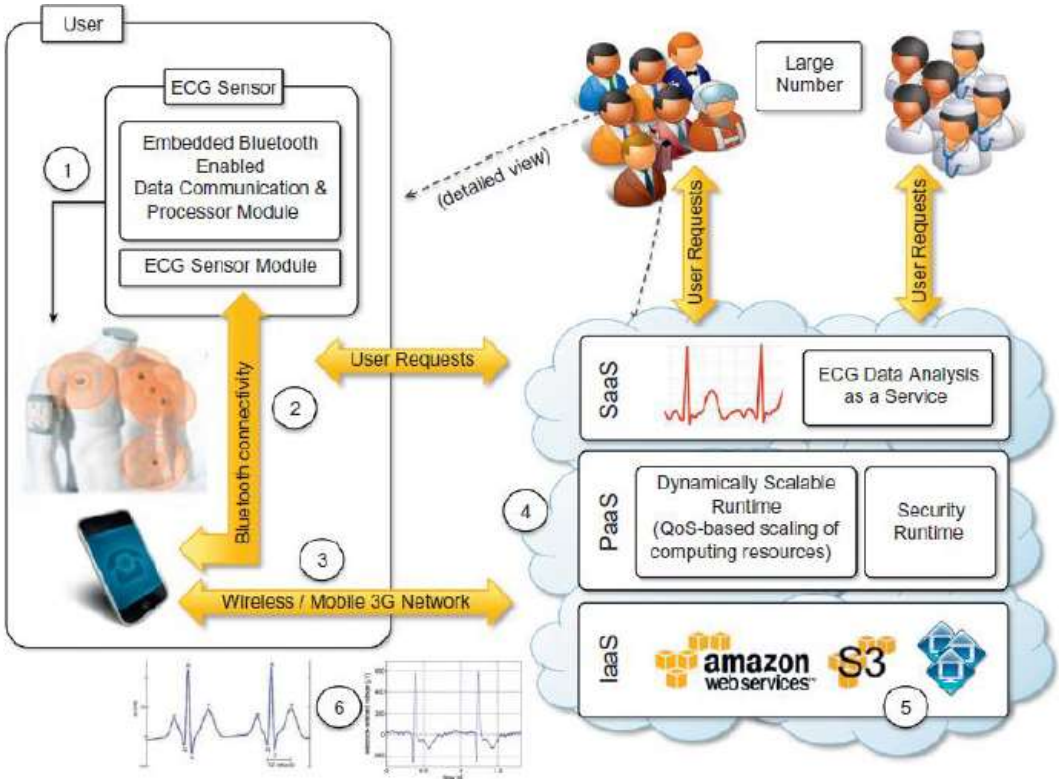
5 (a)	Explain Amazon Dynamo architecture with a neat diagram.	[09]	CO2	L2



- is composed of a collection of storage peers organized in a ring that share same key space.
- **Key space :** partitioned among the storage peers
- Each peer is configured with access to a local storage facility where original objects and replicas are stored
- each node provides facilities for distributing the updates among the rings and to detect failures and unreachable nodes
- Dynamo implements the capability of being an ***always-writable*** store, where consistency of data is resolved in the background
- which requires applications to build their own data models on top of simple building blocks provided by the store.
- there are no **referential integrity constraints**, relationships are not embedded in the storage model, and therefore **join operations** are not supported
- is the distributed key-value store that supports the manage- ment of information of several of the business services offered by Amazon Inc.
- **Goal :** provide an incrementally scalable and highly available storage system
  - To reach reliability at a massive scale.
  - Serve 10 million requests per day.
- **Interface :** get/put semantics
- **ACID properties** are sacrificed for a more reliable and efficient infrastructure.
- Creates an **eventually consistent model** : in the long term, all users will see the same data.

(b)	In ACID properties, which property is relaxed to guarantee high availability in Amazon Dynomo and generally in all distributed databases ?	[1]	CO2	L3

6	Explain how cloud computing can support ECG monitoring with a neat diagram.	10M	CO3	L2



Wearable computing devices equipped with ECG sensors constantly monitor the patient's heartbeat.

►transmitted to the patient's **mobile device**, eventually forwarded to **cloud-hosted Web service** for analysis

►Web service forms the front-end that is entirely hosted in the cloud

►Leverages SaaS, PaaS and IaaS.

►Web service constitute the SaaS application : **store ECG data in the Amazon S3 service**
►iissue a **processing request** to the scalable cloud platform
►runtime platform is composed of a dynamically sizable number of instances running the workflow engine and Aneka.

Tasks : **set of operations involving the extraction of the waveform from the heartbeat data**

►**comparison of the waveform with a reference waveform** to detect anomalies.

►If anomalies are found, **doctors and first-aid personnel** can be notified to act on a specific patient.

CO-PO and CO-PSO Mapping																				
Course Outcomes		**Blooms Level**	**Modules covered**	**PO1**	**PO2**	**PO3**	**PO4**	**PO5**	**PO6**	**PO7**	**PO8**	**PO9**	**PO10**	**PO11**	**PO12**	**PSO1**	**PSO2**	**PSO3**	**PSO4**
CO 1	Explain cloud computing, virtualization and classification of services of cloud computing	**L2**	**1,2**	3	2	2	2	0	3	3	0	0	0	0	0	3	0	2	2
CO 2	Illustrate architecture and programming examples in cloud	**L3**	**2,3, 4**	3	2	0	2	2	3	3	0	0	0	0	0	3	2	0	2
CO 3	Describe the platforms for development of cloud applications with examples	**L2**	**4,5**	3	3	3	3	2	3	3	0	0	0	0	0	3	2	0	2

CO PO Mapping

COGNITIVE LEVEL	REVISED BLOOMS TAXONOMY KEYWORDS
L1	List, define, tell, describe, identify, show, label, collect, examine, tabulate, quote, name, who, when, where, etc.
L2	summarize, describe, interpret, contrast, predict, associate, distinguish, estimate, differentiate, discuss, extend
L3	Apply, demonstrate, calculate, complete, illustrate, show, solve, examine, modify, relate, change, classify, experiment, discover.
L4	Analyze, separate, order, explain, connect, classify, arrange, divide, compare, select, explain, infer.
L5	Assess, decide, rank, grade, test, measure, recommend, convince, select, judge, explain, discriminate, support, conclude, compare, summarize.

PROGRAM OUTCOMES (PO), PROGRAM SPECIFIC OUTCOMES (PSO)				CORRELATION LEVELS	
PO1	Engineering knowledge	PO7	Environment and sustainability	0	No Correlation
PO2	Problem analysis	PO8	Ethics	1	Slight/Low
PO3	Design/development of solutions	PO9	Individual and team work	2	Moderate/ Medium
PO4	Conduct investigations of complex problems	PO10	Communication	3	Substantial/ High
PO5	Modern tool usage	PO11	Project management and finance		
PO6	The Engineer and society	PO12	Life-long learning		
PSO1	Develop applications using different stacks of web and programming technologies				

PSO2	Design and develop secure, parallel, distributed, networked, and digital systems
PSO3	Apply software engineering methods to design, develop, test and manage software systems.
PSO4	Develop intelligent applications for business and industry