Terms used in Spark Execution Framework:

Application: User application built on Spark.

Job: A parallel computation consisting of multiple tasks that gets spawned in response to a Spark action (e.g. save, collect).

Stage: Each job gets divided into smaller sets of tasks called *stages* that depend on each other (similar to the map and reduce stages in MapReduce); you'll see this term used in the driver's logs.

Task: A unit of work that will be sent to one executor.

Application Jar: A jar containing the user's Spark application codes.

Driver Program: The process running the main() function of the application and creating the SparkContext.

Cluster Manager: An external service for acquiring resources on the cluster (e.g. standalone manager, Mesos, YARN). In YARN it is Resource Manager + Application Master.

Deploy Mode: Distinguishes where the driver process runs. In "cluster" mode, the framework launches the driver inside of the cluster. In "client" mode, the submitter launches the driver outside of the cluster.

Worker Node: Any node that can run application code in the cluster.

Executor: A process launched for an application on a worker node, that runs tasks and keeps data in memory or disk storage across them. Each application has its own executors.

Cache: Cache is nothing but the job resources which we are trying to pass as part of execution. One of the job resource can be our code itself. Typically the code will be compiled in the jar file and jar file will be passed as cache and it will be cached into the executors. Tasks will be using this cache at the run time so that it can apply the logic onto the data while data is being processed. We can also persist or cache the RDDs.