Designs:

- One master and multiple slaves. The master is also a front tier VM.
- The master has a thread that scales out middle tier and front tier. It is based on the size of the request queue and the current active middle tier VMs.
- The master computes the average arrival rate of every 10 requests, and decides whether front tier and middle tier should scale in or not.
- Every front tier slave has a thread that controls the scaling in of front tier (by terminating itself). When the threshold is met, the slave will shut down itself to scale in.
- When the scaling in threshold is met, several middle tier VMs will shutdown themselves when they currently have no processing requests received from front tier.
- In the implementation of the cache, the get method will read from cache, or get the value from database if it's not in the cache. The set and transaction will update both the cache and database.

Important data structure and methods in Server.java:

- public static class VmType: A class that stores a VM's id and its type (master/front/middle).
- ConcurrentHashMap<Integer, VmType> frontHashMap: A hashmap that stores the front tier VM's ID and it's VmType.
- ConcurrentHashMap<Integer, VmType> middleHashMap: A hashmap that stores the middle tier VM's ID and it's VmType.
- ArrayList<Integer> frontUpList: An arraylist that stores the booted front tier VMs' IDs.
- private static ArrayList<Integer> middleUpList: An arraylist that stores the booted middle tier VMs' IDs.
- private static ConcurrentLinkedQueue<Cloud.FrontEndOps.Request> requestQueue: A queue that stores the requests sent from front tier.
- private static ConcurrentLinkedQueue<VMType> VMTypeQueue: A queue that saves the booted but not yet assigned VM.
- public Server.VmType getNextVM() throws RemoteException: Get the next type of VM in MTypeQueue
- public void startInitialWMs() throws RemoteException: Master starts initial front tier and middle tier VMs according to the hour of the day
- public boolean isOneMiddleVMRunning() throws RemoteException: Check if at least one of the middle tier VM is running
- public void addRequestToQueue(Cloud.FrontEndOps.Request request) throws RemoteException: Add an incoming request from the front tier to the master server queue
- public Cloud.FrontEndOps.Request getRequestFromQueue() throws RemoteException: Return a request or null to the middle tier from the master server queue

- public int getMiddleScaleOutNum() throws RemoteException: Checks if middle tier should scale out, return the number.
- public void scaleOutMiddle(int scale) throws RemoteException: Scale out middle tier
- public void scaleOutFront(int scale) throws RemoteException: Scale out front tier
- public boolean checkScaleInFront() throws RemoteException: Check if front tier should scale in
- public boolean checkScaleInMiddle() throws RemoteException: Check if front tier should scale in

Important data structure and methods in Cache.java:

- public String get(String key) throws RemoteException: Return the value from cache if it's available in the cache. Otherwise, get the value from database and save it in the cache.
- public boolean set(String key, String value, String auth) throws RemoteException: Call the database's set method, and update the cache.
- public boolean transaction(String item, float price, int qty) throws RemoteException: Call the database's transaction method, and update the cache.