1. Explain the working mechanism of virtual machines.

   Virtual machines are created with the help of programs called hypervisors. Hypervisors are either directly installed on top of the hardware(bare metal) or on top of the host operating system(hosted).

   Then the virtual machines are created in the hypervisor software. Examples of hypervisor software are KVM, Vmware, Oracle virtual box, etc.

   Virtual machines provide much more control to the underlying hardware and software to the running applications.

2. Explain the working mechanism of containers.

   From bottom to top, these are the following components in containerization:

   Infrastructure: It is the lowest level and includes the actual hardware on which the programs run. These include arm chips, intel chips.

   Host operating system:
   The hos operating system runs the container engine or container daemon. It is responsible for handling the networking and other operations for the containers. Ex: linux distos, windows , macos.

   Container runtime:
   Container runtime runs in the host operating system and actually runs the images as containers. Ex: docker, containerd, CRI-O. These runtimes run in the host operating system as a daemon. These daemons manage the lifecycles of containers in the host.

   Images:
   Images are the bundled code which can be run into the container runtime. These images use minimal os kernels and programs which are just needed to run our code. We can build our own container images on top base of these images. All the major operating systems and programs are available as container images.

   Container registry:

The container images are usually stored in the registries. For example: docker hub, github packages, Aws ECR, etc. When our code is ready to be deployed, our images are pushed into the registry from developers and then pulled into the deployment server and run in the server os.

3. What problem does virtualization solve and what is its drawback in context to modern application deployments?

    - Virtualization optimizes the hardware to the max, so it solves the problem of wastage of expensive hardware where a single application runs on a huge server.
    - On a traditional server only one kind of application could be run , but with virtualization different virtual machines of different characteristics can be installed in a single host machine.
    - For example on a single host, we could install a windows VM, a cent os Vm, an Ubuntu VM all running different kinds of applications and used for different tasks. With traditional deployments, we could not run .net program(before it went open source) on a Linux host machine but due to virtualization is possible to do so.

4. What are the problems container solves in regard to app deployment and how does it solve it?

The biggest problem containers solve in app deployment is the difference in the development, testing and deployment environment.

- Due to the use of containers the runtime environment is the same whether it is development, testing, or deployment. In legacy deployment methods, the app would probably run in developers' devices but would have problems running in the servers because of issues with version conflicts, OS version, etc.

- Since it has the same runtime env, the app can be developed, tested, and deployed in any os.
- It is a lot easier to run multiple containers in a host than creating multiple virtual machines on the same host since containers are isolated and self-contained and containers are easier to spin up and destroy than traditional virtual machines.

- Containers are very easy to scale horizontally. It is as easy as running the same image container image multiple times and load balancing between them.