Assignment 3

Task 1

Sure, here are the steps to containerize an application using Docker:

1. Write a Dockerfile: A Dockerfile is a script containing a list of instructions to build a Docker image. It should contain the dependencies and configurations needed to run the application.

2. Build the Docker image: Run the `docker build` command in the directory where the Dockerfile is located. This will create a Docker image based on the instructions in the file.

3. Run the Docker container: Run the `docker run` command to start a Docker container from the image. This will start the application in a container environment.

4. Configure the container: Use environment variables or command line options to configure the container to match the environment in which it will be deployed.

5. Publish the Docker image: Once the container is configured, you can use the `docker push` command to publish the image to a Docker registry, so that others can use it.

Here is an example Dockerfile for a Node.js application:

```
FROM node:16-alpine

WORKDIR /app

COPY package*.json ./

RUN npm install

COPY . .

EXPOSE 8080

CMD ["npm", "start"]
```

To build the Docker image, run:

```
docker build -t my-app-image .
```

To run the Docker container:

```
```

```
docker run -p 8080:8080 my-app-image
```

This will run the container on port 8080 and map it to port 8080 on the localhost.

Task 2

Sure! Here are the steps to upload a Docker image to IBM Container Registry:

1. Create a namespace: If you haven't already, create a namespace in the IBM Container Registry. Namespaces are used to organize your images and prevent name collisions. You can create a namespace through the IBM Cloud console or by using the IBM Cloud CLI command `ibmcloud cr namespace-add`.

2. Log in to IBM Cloud Container Registry: Log in to IBM Cloud using the command line interface by running `include login` and authenticate in the correct account and region.

3. Tag the Docker image: Use the `docker tag` command to tag your Docker image with the name of the registry and namespace.

   ```
   docker tag my-app-image <region>.icr.io/<namespace>/my-app-image:latest
   ```
   Replace `<region>` with the region ID of your registry and `<namespace>` with the namespace you created.

4. Push the Docker image: Use the `docker push` command to push the image to the registry.

   ```
   docker push <region>.icr.io/<namespace>/my-app-image:latest
   ```
   This command will upload your Docker image to the IBM Container Registry in the specified region and namespace.

Once the upload is complete, the image will be available in the IBM Container Registry for deployment to IBM Cloud Kubernetes Service or other container platforms.

Task 3

To orchestrate Docker containers using Kubernetes, follow these steps:

1. Set up a Kubernetes cluster: You will need to set up a cluster of nodes that will run your containers. This can be done through a cloud provider (such as Google Cloud Platform or Amazon Web Services) or on your own local hardware using a tool like Minikube.

2. Define your container in a YAML file: In this file, you will specify the Docker image you want to use, as well as the ports and environment variables you want to expose.

3. Create a Kubernetes deployment: This is where you will specify the number of replicas you want to run in the cluster.

4. Create a Kubernetes service: This will create a stable IP address and DNS name for your deployment, allowing external traffic to access your containers.

5. Deploy your container using kubectl: Use the kubectl CLI tool to create and manage your Kubernetes resources.

6. Scale up or down your deployment as needed: You can easily add or remove replicas of your container by updating the deployment specification.

By following these steps, you can easily manage and scale your Docker containers using Kubernetes.