（a）

Cloud computing is generally understood through three primary service models: Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS). IaaS provides users with virtualized computing resources such as servers, storage, and networking, allowing them to build and manage their own environments with full control over operating systems and applications; a typical example would be using Amazon EC2 to set up a development server.

PaaS, by contrast, offers a managed environment where developers can deploy applications without concerning themselves with system-level administration, enabling them to focus on coding and application logic; services such as Google App Engine and Heroku illustrate this approach.

SaaS represents the most accessible model, in which complete applications are delivered over the internet and consumed directly by end users without any requirement for installation or maintenance, as seen with tools like GitHub, Google Docs, or Slack. Together, these three models illustrate different layers of abstraction in cloud computing, each addressing distinct needs within modern software development.

**(b)**  
Docker and containerization have transformed software development by enabling applications to run consistently across different environments. A container can be thought of as a lightweight package that includes not only the application itself but also its dependencies, runtime, and configuration, ensuring that it behaves the same way regardless of the underlying system. This eliminates the common “works on my machine” problem, since a containerized application tested on a developer’s laptop can be deployed to a testing server or production system with confidence in its reliability. Beyond consistency, containerization also simplifies scalability, as additional instances of a container can be launched quickly to handle increased demand. In software development workflows, this means teams can standardize environments, reduce configuration errors, and accelerate deployment cycles, ultimately achieving greater efficiency and stability across the entire lifecycle of an application.

**(c)**

**1.Pull n8n Docker image**

**docker pull n8nio/n8n**

**docker volume create n8n\_data**

**2.** **Run n8n container**

**docker run -d `**

**>> --name n8n `**

**>> -p 5678:5678 `**

**>> -v n8n\_data:/home/node/.n8n `**

**>> -e N8N\_BASIC\_AUTH\_ACTIVE=true `**

**>> -e N8N\_BASIC\_AUTH\_USER=admin `**

**>> -e N8N\_BASIC\_AUTH\_PASSWORD=606987243 `**

**>> -e N8N\_HOST=0.0.0.0 `**

**>> docker.n8n.io/n8nio/n8n:latest**

