WEB APPLICATION DEPLOYMENT

SFDWD501

Deploy a web application

Competence

REQF Level: 5

Credits: 4

Learning hours

40

Sector: ICT

Sub-sector: Software Development

Issue date: December 2018

Purpose statement

This module describes the skills, knowledge and attitudes to be acquired by the learner to Analyze web application deployment requirements and therefore deploy a web application.

At the end of this module the learner will be able to describe deployment environment requirements, identify deployment tools and equipment, and deploy any web application.

COURSE OUTLINE

Learning Unit 1: Introduce software deployment concepts

10 Hours

Learning Outcomes:

1. 1. Introduce web application Deployment

2. Describe the deployment environment

3. Identify deployment tools, equipment and materials in accordance with application requirements.

Learning Unit 2: Analyze web application deployment requirements

10 Hours

Learning Outcomes:

1. Explore web application deployment server in line with application requirements.

2. Interpret hardware and software deployment requirements.

3. Identify technologies used and network compatibility.

Learning Unit 3: Distribute the application

20 Hours

Learning Outcomes:

1. Inspect web application deployment model.

2. Allocate the web application package.

3. Experiment and launch application in line with customer needs.

Learning Unit 1: Introduce software deployment concepts

Learning Outcomes:

1. **1. Introduce web application Deployment**
   - **Description of Deployment**
     - ✓ **Definition**

Deploying an application is the process of copying, configuring and enabling a specific **application** to a specific base URL on Zend Server or on a cluster. Once the **deployment** process has finished, the **application** becomes publicly accessible on the base URL.

- ✓ **Reasons**

The main reason for deploying a web application is to make it available to user for being used

- ✓ **Benefits**

- Many users at time
- The application can be accessed any where
- The maintenance is easy
- The application is known rapidly as it is online
- Feedbacks from users to enhance system update

- **Exploration of deployment process / activities**
  - ✓ **Release**

The release activity follows from the completed development process, and is sometimes classified as part of the development process rather than deployment process. It includes all the operations to prepare a system for assembly and transfer to the computer system(s) on which it will be run in production.

- ✓ **Installation**

installation involves establishing some form of command, shortcut, script or service for executing the software (manually or automatically). For complex systems it may involve configuration of the system – possibly by asking the end-user questions about its intended use, or directly asking them how they would like it to be configured – and/or making all the required subsystems ready to use.

In larger software deployments on servers, the main copy of the software to be used by users - "production" - might be installed on a production server in a production environment.

- ✓ **Activation**

Activation is the activity of starting up the executable component of software for the first time (not to be confused with the common use of the term activation concerning a software license, which is a function of Digital Rights Management systems.)

- ✓ **Deactivation**

Deactivation is the inverse of activation, and refers to shutting down any already-executing components of a system. Deactivation is often required to perform other deployment activities, e.g., a software system may need to be deactivated before an update can be performed.

- ✓ **Uninstall**

It is the removal of a system that is no longer required. It may also involve some reconfiguration of other software systems in order to remove the uninstalled system's dependencies.

- ✓ **Update**

The update process replaces an earlier version of all or part of a software system with a newer release. It commonly consists of deactivation followed by installation. On some systems, such as on Linux when using the system's package manager, the old version of a software application is typically also uninstalled as an automatic part of the process.

- ✓ **Built-in update**

Mechanisms for installing updates are built into some software systems (or, in the case of some operating systems such as Linux, Android and iOS, into the operating system itself). Automation of these update processes ranges from fully automatic to user initiated and controlled.

- ✓ **Version tracking**

Version tracking systems help the user find and install updates to software systems. For example: Software Catalog stores version and other information for each software package installed on a local system. One click of a button launches a browser window to the upgrade web page for the application, including auto filling of the user name and password for sites that require a login.

- ✓ **Adaptation**

The application is developed with certain requirements, to be deployed it should adapt to all those requirements under which it have been developed.

2. **Describe the deployment environment**

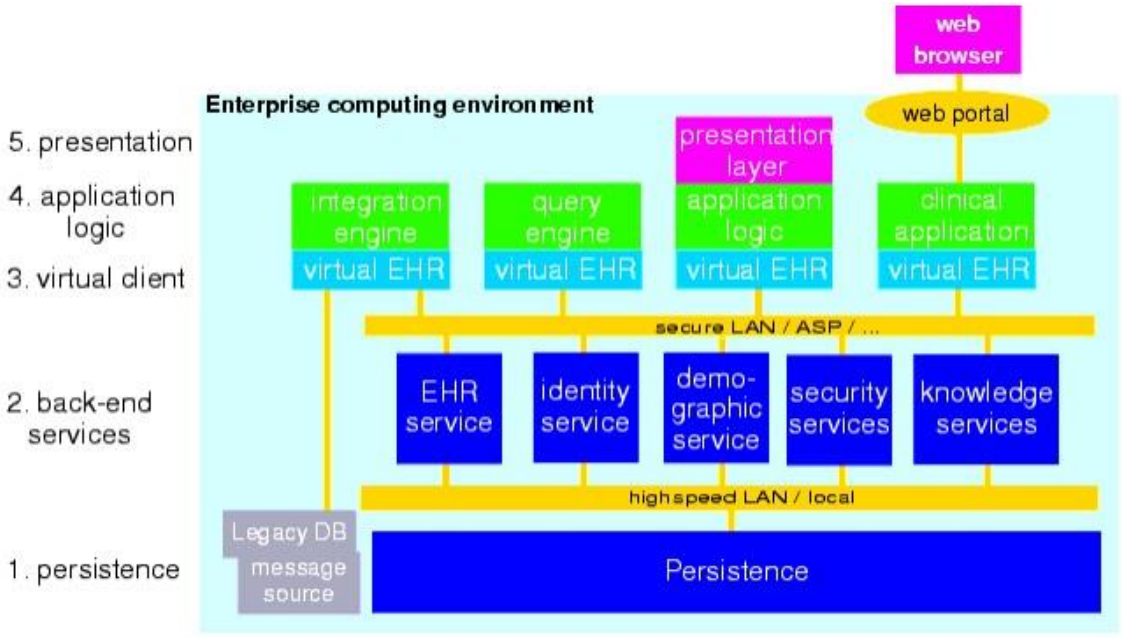

FIGURE 37 Basic Enterprise EHR System Architecture

Description of 5-tier System Architecture

- ✓ **Persistence:** Data storage and retrieval.
- ✓ **back-end services**

Including EHR, demographics, terminology, archetypes, security, record location, and so on. In this layer, the separation of the different services is transparent, and each service has a coarse-grained service interface.

- ✓ **Virtualization**

Virtualization describes a **technology** in which an application, guest operating system or data storage is abstracted away from the true underlying hardware or software.

- ✓ **application logic**

This tier consists of whatever logic is specific to an application, which might be a user application, or another service such as a query engine.

- ✓ **presentation layer**

This layer consists of the graphical interface of the application, where applicable.

- **Exploration of Virtualization technology**
  - ✓ **Description of virtualization**

Virtualization describes a **technology** in which an application, guest operating system or data storage is abstracted away from the true underlying hardware or software. A key use of **virtualization technology** is server **virtualization**, which uses a software layer called a hypervisor to emulate the underlying hardware.

The different types of virtualization that can be implemented.

Server Virtualization – consolidating multiple physical servers into virtual servers that run on a single physical server.

Application Virtualization – an application runs on another host from where it is installed in a variety of ways. It could be done by application streaming, desktop virtualization or VDI, or a VM package (like VMware ACE creates with a player). Microsoft Softgrid is an example of Application virtualization.

Network Virtualization – with network virtualization, the network is "carved up" and can be used for multiple purposes such as running a protocol analyzer inside an Ethernet switch. Components of a virtual network could include NICs, switches, VLANs, network storage devices, virtual network containers, and network media.

Storage Virtualization – with storage virtualization, the disk/data storage for your data is consolidated to and managed by a virtual storage system. The servers connected to the storage system aren't aware of where the data really is. Storage virtualization is sometimes described as "abstracting the logical storage from the physical storage.

- **Properties and benefits of Virtual Machine**

The main advantages of virtual machines:

- Multiple OS environments can exist simultaneously on the same **machine**, isolated from each other;
- **Virtual machine** can offer an instruction set architecture that differs from real **computer's**;
- Easy maintenance, application provisioning, availability and convenient recovery.

Virtual Machine Properties

The Virtual Machine Properties tab shows details about a client computer that is deployed as a virtual machine (VM).

- **Virtualization Client Name**

Name of the client computer that provides a management access point for the hypervisor where the virtual machine is deployed.

- **Virtual Machine Name**

Name of the client computer for the virtual machine.

- **Vendor**

Hypervisor for the virtual machine.

- **Uuid**

Universally unique ID (UUID) for the virtual machine.

- **Hardware Version**

The hardware version of the virtual machine on the host computer, which determines the virtual hardware functions supported for the VM.

- **Host**

IP address for the host where the virtual machine is located.

- **Host Version**

Version and build for the hypervisor software on the host.

- **Hyper-visor technology**

A **hypervisor** or virtual machine monitor (VMM) is computer software, firmware or hardware that creates and runs virtual machines. A computer on which a **hypervisor** runs one or more virtual machines is called a host machine, and each virtual machine is called a guest machine.

Examples of this type of **hypervisor** include VMware Fusion, Oracle Virtual Box, Oracle VM for x86, Solaris Zones, Parallels and VMware Workstation.

3. Identify deployment tools, equipment and materials in accordance with application requirements.

- **Description of software deployment Tools**
  - ✓ **Jenkins**

Jenkins is an open source automation server written in Java. Jenkins helps to automate the non-human part of the software development process, with continuous integration and facilitating technical aspects of continuous delivery. ***It is a server-based system that runs in servlet containers such as Apache Tomcat.*** an open source automation server which enables developers around the world to reliably build, test, and *deploy* their software

- ✓ **Google Cloud Deployment Manager**

Google Cloud Deployment Manager is an *infrastructure management service* that makes it simple to *create, deploy, and manage Google Cloud Platform resources*. With Deployment Manager, you can create a static or dynamic template that describes the configuration of your Google Cloud environment and then use Deployment Manager to create these resources as a single deployment. **CircleCI**

CircleCI's continuous integration and delivery platform helps software teams rapidly release code with confidence by automating the build, test, and deploy process. CircleCI offers a modern software development platform that lets teams ramp quickly, scale easily, and build confidently every day.

- ✓ **Octopus Deploy**

Octopus is the deployment automation server for your entire team, designed to make it easy to orchestrate releases and deploy applications, whether on-premises or in the cloud.

Octopus Deploy *is an automated* ***deployment*** *and release management server*. It is designed to simplify **deployment** of ASP.NET applications, Windows Services and databases. **AWS(Amazon Web Servise) Code Deploy**

AWS CodeDeploy is a fully managed deployment service that automates software deployments to a variety of compute services such as Amazon EC2, AWS Fargate, AWS Lambda, and your on-premises servers. AWS CodeDeploy makes it easier for you to rapidly release new features, helps you avoid downtime during application deployment, and handles the complexity of updating your applications. You can use AWS CodeDeploy to automate software deployments, eliminating the need for error-prone manual operations. The service scales to match your deployment needs.

Benefits
Automated deployments
Centralized control
Minimize downtime
Easy to adopt

- **Description of hardware deployment equipment**
  - ✓ **Server**
  - ✓ **Computer**
  - ✓ **Operating system**
  - ✓ **Antivirus**

Learning Unit 2: Analyze web application deployment requirements

Learning Outcomes:

1. **Explore web application deployment server in line with application requirements.**
   - **Analysis of application requirements**
     - ✓ **Customer Requirements**

Requirements are those characteristics that determine whether or not the **customer** is happy.

Application should fulfill the customer needs and requirements for satisfactions.

- ✓ **Functional Requirements**

Functional requirement. In software engineering and systems engineering, a **functional requirement** defines a **function** of a system or its component, where a **function** is described as a specification of behavior between outputs and inputs.

Typically, functional requirements will specify a behaviour or function, for example:
"Display the name, total size, available space and format of a flash drive connected to the USB port." Other examples are "add customer" and "print invoice".

- ✓ **Non-functional Requirements**

In systems engineering and **requirements** engineering, a **non-functional requirement** (NFR) is a **requirement** that specifies criteria that can be used to judge the operation of a system, rather than specific behaviors. They are contrasted with **functional requirements** that **define** specific behavior or functions.

for example: "Modified data in a database should be updated for all users accessing it within 2 seconds."

As said above, non-functional requirements specify the system's 'quality characteristics' or 'quality attributes'.

- ✓ **Performance Requirements**

Performance should never be an afterthought, but instead, a high-priority **requirement** that gets defined clearly and early on in the **application** lifecycle management process.

Using the SMART mnomic is a good start, meaning that any performance requirement you have should be:

- Specific.
- Measurable.
- Attainable.
- Relevant.
- Timed.

- **Design Requirements**

Requirements analysis is the identification and documentation of the real requirements for a system or change. This is a critical step to ensure success in the development of the project.

The design phase takes the documented requirements specification, and determines and documents how the required new functionality or modification is to be done by the programming and development team.

 Delivery Requirements

Application delivery is a mechanism to deliver application functionality quickly and efficiently to users. Traditionally, hardware application delivery controllers (ADCs) were used to deliver applications. **Allocation Requirements**

Requirements Allocation. **Requirements Allocation** is the act of decomposing higher level **requirements** and assigning tem to lower level functions. All **requirements** of the top-level functions must be met by the aggregate of those for all lower level functions.

- **Identification of suitable deployment server**
  - **Backup Services**

A remote, online, or managed backup service, sometimes marketed as cloud backup or backup-as-a-service, is a service that provides users with a system for the backup, storage, and recovery of computer files. Online backup providers are companies that provide this type of service to end users **Customer Service**

Customer service is the act of taking care of the **customer's** needs by providing and delivering professional, helpful, high quality **service** and assistance before, during, and after the **customer's** requirements are met. **Adequate Scalability**

Scalability is an attribute that describes the ability of a process, network, software or organization to grow and manage increased demand. A system, business or software that **is** described as **scalable** has an advantage because it **is** more adaptable to the changing needs or demands of its users or clients.

Scalability is the property of a system to handle a growing amount of work by adding resources .

- ✓ **Guaranteed Uptime**

Server uptime refers to the amount of time in a given period a **server** stays up and running. Some web hosts may **guarantee** a certain **uptime**, like 100%, 99.9%, or 99%, but then go on to exclude planned maintenances, acts of God, and a whole list of other things.

- ✓ **Reputation**

Talos Intelligence is a product of Cisco and provides you with the tools to check your **reputation** by ranking you as Good, Neutral, or Poor.

The server reputation determines how the server is known interms of offering services.

2. Interpret hardware and software deployment requirements.

- **Identification of hardware deployment requirements**
  - ✓ **Server processor**

The server processor is the first consideration on the server, it determines the processing power of the server which will be responsible to answer requests from clients. The processor capacity is measured in htz (**heltz).** The biger the processor, the powerful is the server.

- ✓ **Server RAM**

As the it is said to be working memory, any application to be accessed by users (clients) should be loaded into RAM. So whatever process needed by processor, is loaded in RAM. Hence the bigger the RAM, the better the server.

✓ **Disk storage**

The role of the Disk storage is to store data which serve to different clients.

If the disk storage is small, it will store fiew data, and if it is big it will store enough data.

- **Identification of Software deployment requirements**
  - **Operating system**

Popular **server operating systems** include Windows **Server**, Mac **OS** X **Server**, and variants of Linux such as Red Hat Enterprise Linux (RHEL) and SUSE Linux Enterprise **Server**. It uses these kind of OS, because they the one offering high security compared to windows OS.

✓ **Third party software's compatibility**

Here you need to know different software the server uses, if they will be compatible with the software installed on the local machine.

✓ **Firewalls settings**

A **firewall** is a system that provides network security by filtering incoming and outgoing network traffic based on a set of user-defined rules.

- **Production of analysis report**

  - **Server**

A **server** is a **computer** that provides data to other **computers**. It may serve data to systems on a local area network (LAN) or a wide area network (WAN) over the Internet. Many types of **servers** exist, including web **servers**, mail **servers**, and file **servers**. Each type runs software specific to the purpose of the **server**.

  - **Deployment package**

A deployment package is a ZIP archive that contains your function code and dependencies. You need to create a deployment package if you use the Lambda API to manage functions, or if you need to include libraries and dependencies other than the AWS SDK. You can upload the package directly to Lambda, or you can use an Amazon S3 bucket, and then upload it to Lambda. If the deployment package is larger than 50 MB, you must use Amazon S3.

3. Identify technologies used and network compatibility.

- **Exploration of Network compatibility.**
  - **Protocols**

The Internet relies on a number of *protocols* in order to function properly. A protocol is simply a standard for enabling the connection, communication, and data transfer between two places on a network. Here are some of the key protocols that are used for transferring data across the Internet.

HTTP

HTTP stands for Hypertext Transfer Protocol. It is the standard protocol for transferring web pages (and their content) across the Internet.

HTTPS

HTTPS stands for Hypertext Transfer Protocol over Secure Socket Layer. Think of it as a secure version of HTTP. HTTPS is used primarily on web pages that ask you to provide personal or sensitive information (such as a password or your credit card details).

When you browse a web page using HTTPS, you are using **SSL** (***Secure Sockets Layer***). For a website to use HTTPS it needs to have an *SSL certificate* installed on the server.

FTP

FTP stands for File Transfer Protocol. It is used to transfer files across the Internet. FTP is commonly used by web developers to publish updates to a website (i.e. to upload a new version of the website).

- ✓ **Server accessibility**

Enter the IP address or hostname of the **server** into the web browser to **access your server.**

After accessing it, you will need the authentication parameters to allow you enter in it.

Authentication parameters can be Username and Password **" used In cpanel",** or Username, Password, and port "**used on FTP".**

- ✓ **Server visibility**

Server visibility determines the server settings and structures including all possible services which you can benefit.

Identification of web technology used for web development.

- **PHP Version**

With a variety of available PHP versions it can be difficult to identify which version your site is taking advantage of. This can be even more challenging to determine given that PHP versions are defined in the .htaccess file, meaning that subfolders can inherit the versions their parent directories are using.

The easiest way to be certain what version of PHP a folder is using is to create a phpinfo.php file in that directory and navigate to it. The top of the phpinfo page will tell you what version of PHP is being referenced by that directory.

- ✓ **MySQL version**

The new version offers a great support in different services compared to the old versions.

The MySQL support team, which includes both MySQL developers and support engineers, offer 24/7 support for customers

- ✓ **Apache server**

Apache is an open-source and free **web server** software that powers around 46% of websites around the world. The official name is **Apache** HTTP **Server**, and it's maintained and developed by the **Apache** Software Foundation. It allows website owners to serve content on the **web** — hence the name "**web server**".

Learning Unit 3: Distribute the application

Learning Outcomes:

1. **Inspect web application deployment model.**
   - **Analysis of web application deployment models**
     - **Platform as a service (PaaS)**

Platform as a Service (**PaaS**) or **Application Platform as a Service** (**aPaaS**) or platform-based service is a category of cloud computing services that provides a platform allowing customers to develop, run, and manage applications without the complexity of building and maintaining the infrastructure typically associated with developing and launching an app.

Examples: AWS Elastic Beanstalk, Windows Azure, Heroku, Force.com, Google App Engine, Apache Stratos.

The advantages of PaaS are primarily that it allows for higher-level programming with dramatically reduced complexity; the overall development of the application can be more effective, as it has built-in/self up-and-down ramping infrastructure resources; and maintenance and enhancement of the application is thus easier.

Disadvantages of various PaaS providers as cited by their users include increased pricing at larger scales, lack of operational features, reduced control, and the difficulties of traffic routing systems

- **Infrastructure as a service (IaaS)**

Infrastructure as a service (**IaaS**) is a **service** model that delivers computer **infrastructure** on an outsourced basis to support enterprise operations. Typically, **IaaS** provides hardware, storage, servers and data center space or network components; it may also include software.

Infrastructure as a service (*IaaS*) is a form of cloud computing that provides virtualized computing resources over the internet. *IaaS* is one of the three main categories of cloud computing *services*, alongside software as a *service* (SaaS) and platform as a *service* (PaaS).

Examples of IaaS cloud computing:

- Amazon, Abiquo Enterprise Edition, CloudStack, Citrix Cloud, CtrlS, DigitalOcean

- **Software as a service (SaaS)**

Software as a service (**SaaS**) is a **software** distribution model in which a third-party provider hosts applications and makes them available to customers over the Internet.

SaaS examples: BigCommerce, Google Apps, Salesforce, Dropbox, MailChimp, ZenDesk, DocuSign, Slack, Hubspot

2. **Allocate the web application package.**
   - **Allocation of web application deployment package.**
     - ✓ **Application files**

It is a set of all files and folders which make up your application, the containing folder not included. Here you wil find the index file, and other files of codes, and also folders like images folder, and others.

- **Database file**

The database file is made by your exported database file which represent the database and its tables and related schema.

- **Application Code/logic configuration**
  - ✓ **Client-side code**

Code running in the browser is known as **client-side code** and is primarily concerned with improving the appearance and behavior of a rendered web page. This includes selecting and styling UI components, creating layouts, navigation, form validation, etc.

Client-side code is written using HTML, CSS, and JavaScript — it is run inside a web browser and has little or no access to the underlying operating system (including limited access to the file system).

Server-side code

Server-side code can be written in any number of programming languages — examples of popular serverside web languages include PHP, Python, Ruby, C#, and NodeJS(JavaScript). The server-side code has full access to the server operating system and the developer can choose what programming language (and specific version) they wish to use.

Developers typically write their code using **web frameworks**. Web frameworks are collections of functions, objects, rules and other code constructs designed to solve common problems, speed up development, and simplify the different types of tasks faced in a particular domain.

Most of the code to support a dynamic website must run on the server. Creating this code is known as "**server-side programming**" (or sometimes "**back-end scripting**").

- **Configuration of database server**
  - **Server configuration**

Database server configuration will need to configure the following:

- **Server Name**: the name of the database server ***"localhost"***
- **User Name:** the user of your database by default is **"root",** but this can change to the created user.
- **Password:** the password used to access db, by default is ***empty***, but it can change to any password created.
- **Database Name:** name of the database you need to use. **Server validation**

Validations can be performed on the **server** side or on the client side (web browser). The user input **validation** take place on the **Server** Side during a post back session is called **Server** Side **Validation** and the user input **validation** take place on the Client Side (web browser) is called Client Side **Validation**.

The server validation is a process of saving changes made to the server configurations.

3. Experiment and launch application in line with customer needs.

- **Collection process of web application files.**
  - ✓ **Index file**

An **index file** is the first web page that anyone visiting your website will see, and it need to be named "**index**.htm" or "**index**.html" in order for web browsers to find it.

✓ **Resources folder**

contains **folders** for storing binary files, data files, image files, and Include and Library **folders** that are used to store code used by external functions. The **Resources folder** can contain more or less **folders.**

Includes folder

Is another solution you can use, but you must have all your files inside a folder of your application, you can then search a folder for all PHP files and include all these files in your application.

✓ **Classes**

a **class** contains both data (variables) and functions that form a package called an: 'object'. When you create a variable inside a **class**, it is called a 'property'.

 Database

The database created must be exported to be deployed to the server. Then the creation of the database on the server, and import to the server the exported database on the local machine.

Deployment in staging environment

A **stage** or **staging environment** is an **environment** for testing that exactly resembles a production **environment**. It seeks to mirror an actual production **environment** as closely as possible and may connect to other production services and data, such as databases.

 Staging server

A **staging server** is a type of **server** that is used to test a software, website or service in a production-similar environment before being set live. It is part of a **staging** environment or **staging** site, where it serves as a temporary hosting and testing **server** for any new software or websites. **Staging Database**

A *staging* area, or landing zone, is an intermediate storage area used for *data* processing during the extract, transform and load (ETL) process.

A *staging database* is used as a "working area" for your ETL. Olaf has a good definition: A *staging database* or area is used to load *data* from the sources, modify & cleansing them before you final load them into the server. **Staging application files**

A **Staging application** offers an isolated environment that allows you to push and pull changes between Live and **Staging applications** without impacting the Live **application.**

Testing phase and validation process

✓ **Smoke test**

SMOKE TESTING, also known as "Build Verification **Testing**", is a type of software **testing** that comprises of a nonexhaustive set of **tests** that aim at ensuring that the most important functions work. The result of this **testing** is used to decide if a build is stable enough to proceed with further **testing.**

✓ **User acceptance**

user acceptance testing (UAT)—also called application **testing**, and end **user testing**—is a phase of software development in which the software is tested in the "real world" by the intended audience. **Validity**

Is the most important criteria for the quality of a **test**. The term **validity** refers to whether or not the **test** measures what it claims to measure. On a **test** with high **validity** the items will be closely linked to the **test's** intended focus

- **Deployment in production environment.**
  Deployment server

deployment server Instances that are remotely configured by *deployment servers* are called *deployment* clients. The *deployment server* downloads updated content, such as configuration files and apps, to *deployment* clients. Units of such content are known as *deployment* apps.

✓ **Database**

A *database* is a collection of information that is organized so that it can be easily accessed, managed and updated. Computer *databases* typically contain aggregations of data records or files, containing information about sales transactions or interactions with specific customers.

 Application files

Is a set of files which make up the application and which basically are used to run the

application. **Final delivery**

A **Web application** (**Web app**) is an **application** program that is stored on a remote server and **delivered** over the Internet through a browser interface. **Web** services are **Web apps** by definition and many, although not all, websites contain **Web apps**.