

# APUNTS D'EXAMEN

---

## 1. Desenvolupament

---

### 1.1. Què és SPRING BOOT?

És una extensió del framework SPRING que ha arribat a ser l'estàndard de la indústria per la seva estabilitat, la base comunitària i la seva extensibilitat.

### 1.2. Característiques principals de SPRING BOOT

- Està basat en JAVA.
- És molt robust.
- És autoconfigurable, la qual cosa simplifica el seu desplegament.
- Starter dependencies: són paquets pre-configurats amb llibreries comunes.
- Servidor embegut com Tomcat o Jetty, que permet executar-lo de forma stand-alone.
- Production ready: proporciona observabilitat i health check "out of the box".
- Injecció de dependències.

### 1.3. Elements principals d'arquitectura de SPRING BOOT

- S'empaqueten en JAR autocontinguts que es poden executar de manera separada o en un servidor Spring.
- A nivell lògic utilitza una arquitectura: Controller → Service → DAO → DB.
- Es basa en els components principals Spring Core, SPRING MVC, SPRING DATA JPA i SPRING SECURITY.

---

### 1.4. Què és QUARKUS?

És un framework per a la implementació d'APIs optimitzat per a temps de resposta i consum de memòria.

### 1.5. Característiques principals de QUARKUS

- Cloud first.
- Arrenca en pocs mil·lisegons i consumeix poca memòria.
- Compilació nativa amb GraalVM amb configuració embedded.

- Live coding: permet recompilar i executar en viu.
- Reactive programming: enfocat en esdeveniments.

## 1.6. Elements principals d'arquitectura de QUARKUS

- GraalVM per compilar.
  - Injecció de dependències.
- 

## 1.7. Què és MYSQL?

És un RDBMS obert i gratuït, adquirit per ORACLE.

## 1.8. Característiques principals de MYSQL

- Open source, fiable amb versions TLS, àmpliament utilitzat.
- Multiplataforma.
- Seguretat robusta amb controls d'accés, gestió d'usuaris i permisos.
- Alt rendiment optimitzat en lectura.
- Llenguatges SQL i suport per a molts llenguatges de programació.
- Ús estès en la web amb LAMP.
- Flexibilitat de motor d'emmagatzematge.
- Comunitat gran.
- Versió Enterprise de pagament amb funcionalitats de load balancing i seguretat estesa.
- MARIADB és un fork open source.

## 1.9. Elements principals d'arquitectura de MYSQL

- **Servidor**: és el procés principal que gestiona totes les operacions.
  - **Motor d'emmagatzematge INNODB**: motor específic per a emmagatzemar i gestionar dades.
  - **Sqldump**: eina per fer backups.
  - **Mysql**: client per a consulta i gestió de la base de dades.
- 

## 1.10. Què és ORACLE?

És un RDBMS privatiu molt estable, segur i escalable amb tarificació per cores.

## 1.11. Característiques principals d'ORACLE

- Escalabilitat: gestiona grans volums de dades i usuaris.
- Alta disponibilitat: replicació de dades, conmutació per error i recuperació de desastres.
- Seguretat avançada: autenticació, xifratge, auditoria automàtica, control granular de rols i permisos.
- Multiplataforma.
- Llenguatges PL/SQL.
- Automatització i gestió autònoma.
- Alt rendiment i disponibilitat.

## 1.12. Elements principals d'arquitectura d'ORACLE

- **Instance:** el servidor en execució que gestiona la comunicació i processos (similar al server en MySQL).
  - **Database:** fitxers de dades (datafile en MySQL).
  - **Schema:** conjunt d'objectes de la base de dades d'un usuari (concepte similar a database en MySQL).
  - **Tablespace:** unitat lògica d'emmagatzematge que agrupa fitxers de dades.
- 

## 1.13. Què és APACHE2?

És un servidor web molt popular, open source, modular i segur.

## 1.14. Característiques principals d'APACHE2

- Arquitectura modular: estructura extensible mitjançant mòduls.
- Virtual hosts: permet gestionar múltiples servidors virtuals en un mateix servidor.
- Suport SSL/TLS per a connexions segures.
- Load balancing per distribució de càrrega.
- Multiplataforma.
- Suport per a contingut estàtic i dinàmic.

## 1.15. Elements principals d'arquitectura d'APACHE2

- **Nucli (procés):** gestiona les operacions bàsiques del servidor.
- **Mòduls:** unitats funcionals independents que proporcionen serveis específics i extensen la funcionalitat del servidor.

- **Directoris i fitxers de configuració:** contenen i organitzen la configuració del servidor.
  - **Virtual hosts:** permeten allotjar múltiples dominis o webs en un mateix servidor físic.
- 

## 1.16. Què és NGINX?

És un servidor web lleuger, open source, enfocat a contenidors i alta concurrència.

## 1.17. Característiques principals de NGINX

- **Container first:** dissenyat per a entorns amb contenidors, arrencada ràpida.
- **Observabilitat i health check** integrats.
- **Cloud first:** preparat per desplegar-se en entorns Cloud de tipus IaaS.
- **Reverse proxy (proxy invers)** per gestionar connexions i redireccions.
- **Arquitectura orientada a esdeveniments (event driven)** per eficiència.
- **Load balancing** distribuït.
- **Caching** per millorar rendiment.

## 1.18. Elements principals d'arquitectura de NGINX

- **Procés Master:** gestor central de configuració i supervisió.
  - **Processos Worker:** executen les peticions de forma asíncrona i multiplexada.
  - **Arquitectura event driven:** gestiona múltiples connexions de forma eficient.
  - **Bloc de configuració:** defineix regles i comportaments mitjançant fitxers textuais.
  - **Gestió de continguts:** serveix contingut estàtic i redirigeix peticions a contingut dinàmic.
- 

## 1.19. Què és APACHE TOMCAT?

És un servidor d'aplicacions Java J2EE, open source i estàndard de la indústria.

## 1.20. Característiques principals d'APACHE TOMCAT

- Implementa Java Servlet i JavaServer Pages (JSP).
- Servidor web autònom o integrat amb Apache2.
- Alta disponibilitat amb suport a clustering.
- Suport per a aplicacions empresarials amb alta concurrència, autenticació i seguretat.
- Arquitectura modular dissenyada per components.

## 1.21. Elements principals d'arquitectura d'APACHE TOMCAT

- **Server:** procés d'alt nivell que administra serveis.
  - **Service:** enllaça connectors amb un engine i gestiona la comunicació HTTP o AJP.
  - **Engine:** processa sol·licituds i determina el host virtual corresponent.
  - **Host:** domini virtual que conté les aplicacions web.
  - **Servlet Container:** component que executa aplicacions web Java.
  - **Connector:** gestiona la comunicació client-servidor, per exemple Coyote (HTTP) i JK2 (AJP).
  - **Context:** representa una aplicació web desplegada dins d'un host.
  - **JSP Engine:** processa pàgines JSP.
  - **JNDI Support:** suport a Java Naming and Directory Interface.
  - **Connection Pooling:** gestiona les connexions a bases de dades.
- 

## 1.22. Què és JBOSS EAP?

És un servidor d'aplicacions Java J2EE open source desenvolupat per Red Hat, robust per aplicacions empresarials.

## 1.23. Característiques principals de JBOSS EAP

- Compliment Java EE.
- Clustering per alta disponibilitat.
- Consola de gestió per administrar el servidor.
- Arquitectura modular amb mòduls carregables dinàmicament.
- Seguretat integrada i gestió avançada de sessions.

## 1.24. Elements principals d'arquitectura de JBOSS EAP

- **Contenidor modular (Modules):** sistema que permet la càrrega selectiva i modular de serveis.
- **Subsistemes:** gestionen funcions com transaccions, persistència i seguretat.
- **Servidor d'aplicacions:** executa Enterprise Java Beans (EJB) i components Java EE.
- **CDI (Contexts and Dependency Injection):** motor per a la injecció de dependències dins d'aplicacions.
- **ORM/Hibernate:** framework per a la gestió d'accés i persistència de dades en bases SQL.
- **Servidor web integrat:** utilitza Apache Tomcat.

- **Clustering i alta disponibilitat:** suporta cache distribuït i gestió de sessions persistents.
  - **Gestió i monitoratge:** consola d'administració i APIs JMX per monitoritzar el servidor.
  - **Mòduls de seguretat:** proporcionen autenticació i autorització robusta.
- 

## 1.25. Què és GIT?

És un sistema de control de versions distribuït (VCDs) utilitzat per gestionar canvis en codi font.

## 1.26. Característiques principals de GIT

- Distribuït i descentralitzat: cada desenvolupador té una còpia completa del codi.
- Rastreig precís de canvis: registre detallat de cada commit.
- Ramificació (branches) eficient i lleugera per desenvolupar noves funcions paral·lelament.
- Fusió de branques simplificada.
- Alt rendiment i eficiència en la gestió d'històrics.
- Segur i fiable.
- Suport per a col·laboració distribuïda.
- Versàtil per a diferents tipus de projectes.

## 1.27. Elements principals d'arquitectura de GIT

- **Repositori:** emmagatzema fitxers i historial del projecte.
  - **Arbre de treball (Working Directory):** còpia del codi per modificar localment.
  - **Àrea d'indexació (Staging Area):** lloc per preparar canvis abans de confirmar (commit).
  - **Commit:** instantània de l'estat del projecte amb canvis registrats.
  - **Objectes Git:** estructures internes com blob, tree, commit i tag que emmagatzemen contingut i informació.
  - **Branches:** punters a commits que permeten desenvolupaments independents.
  - **Referències (Refs):** apunten a diferents punts dins del repositori i faciliten l'accés.
- 

## 1.28. Què és GITHUB?

És una plataforma web per allotjar repositoris GIT que facilita la col·laboració i la gestió de projectes de programari.

## 1.29. Característiques principals de GITHUB

- Suporta repositoris públics i privats.
- Facilita la col·laboració mitjançant pull requests i revisió de codi.
- Integració CI/CD mitjançant GitHub Actions.
- Sistema de seguiment de problemes (issues).
- Comunitat molt activa i projectes open source massius.
- Seguretat avançada amb permisos detallats i autenticació de doble factor.
- GitHub Pages per allotjament de pàgines i wikis.

## 1.30. Elements principals d'arquitectura de GITHUB

- Plataforma SaaS (Software as a Service).
  - Connectors per integrar amb solucions corporatives CI/CD.
- 

## 1.31. Què és GITFLOW?

És un model estricte de flux de treball per a GIT que organitza el desenvolupament amb ramificacions ben definides per facilitar la col·laboració i el control de versions.

## 1.32. Característiques principals de GITFLOW

- Dues branques principals: **main** i **develop**.
  - **Feature branches**: per noves funcionalitats, creades a partir de develop.
  - **Release branches**: per fer proves i corregir errors abans d'integrar les noves versions a main.
  - **Hotfix branches**: per correccions crítiques directament sobre main.
  - Proporciona un aïllament clar del treball en curs.
  - Ideal per a projectes amb ciclicitat complexa i equips paral·lels.
  - Permet un cicle d'entrega controlat i segur.
- 

# 2. Arquitectures

---

## 2.1 Què és AWS?

Amazon Web Services és una plataforma de cloud computing més utilitzada.

## 2.2 Característiques principals de AWS

- EC2 (Elastic Compute Cloud): màquines virtuals escalables.
- Lambda: computació serverless per a funcions.
- ECS/EKS: contenidors i [translate:kubernetes] gestionats.
- Elastic Beanstalk: [translate:PaaS] multilingatge.
- S3 (Simple Storage Service): emmagatzematge d'objectes en buckets.
- EBS (Elastic Block Store): emmagatzematge en blocs per a EC2.
- EFS (Elastic File System): sistema de fitxers compartit.
- RDS (Relational Database Service): bases de dades relacionals.
- DynamoDB: base de dades [translate:NoSQL].
- ElasticCache: cache en memòria (Redis/Memcached).
- VPC (Virtual Private Cloud): xarxa virtual aïllada.
- CloudFront: [translate:CDN (Content Delivery Network)] proxy cache.
- Route 53: DNS gestionat.
- ELB (Elastic Load Balancer): balancejadors de càrrega.

## 2.3 Què és GCP?

Google Cloud Platform és el servei cloud de Google.

## 2.4 Característiques de GCP

- Compute Engine: màquines virtuals.
- Cloud Functions: computació serverless.
- GKE (Google Kubernetes Engine): [translate:kubernetes] gestionats.
- App Engine: [translate:PaaS].
- Cloud Storage: emmagatzematge d'objectes.
- Persistent Disk: discos bloc per a VMs.
- Cloud Filestorage: sistema de fitxers [translate:NFS].
- Cloud SQL: bases de dades relacionals gestionades.
- Cloud Spanner: base de dades distribuïda.
- Firestore: bases de dades [translate:NoSQL].
- VPC: xarxa privada virtual.
- Cloud CDN (Content Delivery Network).
- Cloud DNS: DNS.
- Cloud Load Balancing: balancejador de càrrega.



## 2.5 Què és Docker?

És una plataforma de contenidorització efímera que permet empaquetar aplicacions i les seves dependències en contenidors lleugers.

## 2.6 Característiques de Docker

- Contenedors lleugers i aïllats: són més lleugers que les [translate:VM] perquè es basen en versions simplifiades de [translate:SO].
- Portabilitat: com contenen totes les dependències, es poden executar en qualsevol plataforma.
- Imatges basades en capes: que es poden reutilitzar, estalviant espai i facilitant la distribució incremental.
- Gestió i execució eficient: Docker daemon s'encarrega de parar, iniciar i destruir els contenidors.
- Volumes de dades: que es connecten amb els contenidors efímers per mantenir la persistència.
- Ecosistema i integració: dins de [translate:kubernetes].

## 2.7 Elements principals d'arquitectura de Docker

- Host Docker: és la màquina física on s'executa Docker Engine.
- Docker Engine: el procés nucli de Docker format per tres components:
  - dockerd: el daemon que executa les ordres i gestiona imatges, contenidors, xarxes, etc.
  - API REST: que comunica el daemon amb el client de Docker.
  - Docker Client (CLI): interfície d'usuari.
- Imatge: plantilles readonly construïdes en capes amb el sistema operatiu, dependències, la configuració i l'aplicació.
- Contenedor: instància lleugera, aïllada i efímera de la imatge.
- Docker Registry: repositori d'imatges Docker.

## 2.8 Què és Kubernetes?

És un sistema d'orquestració de contenidors que automatitza el desplegament, l'escalat i la gestió de contenidors.

## 2.9 Característiques de Kubernetes

- Orquestració: organitza i distribueix els contenidors entre els treballadors.

- Escalabilitat automàtica: del nombre de contenidors en execució en funció de mètriques.
- Alta disponibilitat: detecta caigudes i recupera automàticament.
- Actualització i rollback en calent: actualitza versions de contenidors sense temps de parada.
- Autodetecció de serveis i balanceig: proporciona un DNS i IP interna, i balanceja la càrrega entre contenidors.
- Portabilitat: multiplataforma.
- Gestió de secrets: evita guardar informació en imatges.
- Ecosistema i comunitat.

## 2.10 Elements principals d'arquitectura de Kubernetes

- kube-apiserver: punt d'entrada API REST que gestiona ordres d'entrada.
- etcd: base de dades distribuïda amb la configuració i l'estat del clúster.
- kube-scheduler: decideix en quin node s'executarà un pod segons disponibilitat.
- kube-controller-manager: gestiona els controladors que mantenen l'estat del clúster.
- Cloud Controller Manager: interacciona amb els serveis del proveïdor cloud.
- kubelet: agent que s'executa en cada node i monitoritza els pods.
- kube-proxy: administra la xarxa i el balanç de càrrega dins del node, gestiona el trànsit cap als serveis i els pods.

## 2.11 Què és WSO2?

És una plataforma open source que gestiona tot el cicle de vida de les APIs: crear, publicar, governar, assegurar, monetitzar, observar les APIs.

## 2.12 Característiques de WSO2

- Disseny i creació API-First: APIs amb OpenAPI (Swagger).
- Publicació i portal de desenvolupadors: facilita la publicació amb un portal interactiu per descobrir, provar i subscriure una API.
- Seguretat i control d'accés: implementa OAuth2, JWT, API keys i control de permisos a nivell granular.
- Gestió del trànsit i qualitat del servei: inclou limitació de rate, polítiques de priorització i gestió del rendiment.
- Monetització i anàlisi: permet crear models de pagament, supervisar l'ús de les APIs i dades analítiques per la presa de decisions.

- Governança i lifecycle management: controla tota la vida útil, versions, actualitzacions, retirada i continuous delivery de les APIs.

## 2.13 Elements principals d'arquitectura WSO2

- API Publisher: crea, publica i gestiona les APIs; definició de recursos i paràmetres; polítiques de seguretat; gestió de versions.
- API Developer Portal: catàleg d'APIs, descobrir, provar APIs, documentació i monitorització d'ús.
- API Gateway: executa les APIs, autentica i autoritza, aplica rate limit, transformació de missatges, logging i anàlisi.
- Key Manager: gestió de tokens d'autenticació i autorització (OAuth, JWT), valida l'accés a les APIs i proporciona endpoints de renovació de tokens.

## 2.14 Què és microservei?

És un enfocament d'arquitectura de desenvolupament d'aplicacions com a conjunt de serveis petits i independents.

## 2.15 Característiques de microserveis

- Single responsibility: cada microservei té una única responsabilitat, [translate:Domain Driven Design].
- Descentralització: cada servei és independent amb bases de dades diferents, no tenen per què córrer en el mateix servidor o cloud.
- Comunicació via APIs: normalment HTTP REST, GraphQL o gRPC binària de gran rendiment.
- Deployment independent: es poden desplegar de manera autònoma sense que afecti els altres microserveis.
- Technology diversity: cadascun pot estar fet amb una tecnologia diferent.
- Escalabilitat independent segons necessitat.
- Desenvolupament i desplegament àgils.
- Resiliència a fallades.
- Equips petits, problemes simplificats.
- Complexitat de xarxa i comunicació.
- Gestió de dades més distribuïda.
- Monitorització i debugging més complex.

## 2.16 Què és Apache Kafka?

És una plataforma de streaming distribuïda que permet publicar, subscriure, emmagatzemar i processar streams de dades en temps real.

## 2.17 Característiques d'Apache Kafka

- Alta escalabilitat: horitzontal afegint brokers per gestionar TB de dades sense degradació de servei.
- Alta disponibilitat i resiliència: amb la replicació de particions entre múltiples brokers per si un d'ells falla.
- Baixa latència: de pocs mil·lisegons per missatge, ideal per a processament en temps real.
- Persistent storage: de missatges en disc en ordre dins de cada partició.
- Processament en streaming: amb la seva API integrada que transforma i processa.
- Model publicació-suscripció: de diversos productors a diversos consumidors.
- Integracions i ampli ecosistema: Apache Spark, Flink, Apache NiFi.
- Autogestió amb KRaft: redueix latències eliminant la necessitat d'Apache ZooKeeper.

## 2.18 Principals elements d'arquitectura d'Apache Kafka

- Brokers: servidors que formen el clúster, emmagatzemen i gestionen les dades dels topics i la comunicació amb els clients.
- Topics: canals o categories lògiques dels missatges; es divideixen en particions per paral·lelitzar i escalar.
- Particions: subdivisions d'un topic que garanteixen l'ordre dels missatges dins de cada partició i gestionen la tolerància a fallades.
- Replicació: de particions en els brokers; una és líder i les altres són sincronitzades per seguretat.
- Productors: aplicacions que publiquen missatges en un topic.
- Consumers: aplicacions que es subscriuen i consumeixen missatges d'un topic.
- ZooKeeper: servei extern que coordina i gestiona la configuració entre els brokers.

## 2.19 Què és Apache Spark?

És una plataforma [translate:opensource] per al processament distribuït i àgil de grans volums de dades, ràpida i eficaç, que processa tant dades estructurades com no estructurades, permetent executar anàlisis, machine learning i processament en temps real.

## 2.20 Característiques d'Apache Spark

- Processament en memòria: emmagatzema dades en RAM per un processament eficient, no com Hadoop MapReduce.
- Multimòdul i versàtil: ja integrats com Spark SQL, Spark Streaming, MLlib i GraphX per processament de gràfics.
- Escalabilitat: amb arquitectura distribuïda, per paralel·litzar tasques en clústers.
- Multi plataforma: executable de manera independent, dins d'Apache Hadoop, Kubernetes, pot accedir a AWS S3, Apache Cassandra, Kafka, etc.
- Multilenguatge de programació.

## 2.21 Principals elements d'arquitectura d'Apache Spark

- Spark Driver: controlador nucli, coordina les tasques als executors, la comunicació amb el cluster manager, construeix el gràfic acíclic dirigit (DAG).
- Executors: nodes treballadors del clúster amb una JVM pròpia, calculen les dades, mantenen les dades en memòria i comuniquen resultats al driver.
- Cluster Manager: gestiona recursos per als executors.
- Spark Context: punt d'entrada d'una aplicació Spark.

## 2.22 Què és ELK?

És un conjunt de tres eines [translate:opensource]: Elasticsearch, Logstash i Kibana, per a la gestió, anàlisi i visualització de grans volums de dades.

## 2.23 Principals elements d'arquitectura d'ELK

- Elasticsearch: motor de cerca i anàlisi distribuït. Permet buscar i analitzar grans quantitats de dades en temps real.
- Logstash: eina d'ingesta i processament de dades que recopila, filtra, transforma i envia les dades a Elasticsearch. Pot connectar diverses fonts, formats i aplicar lògiques complexes.
- Kibana: plataforma de visualització i exploració de dades, gràfics interactius, mapes, dashboards, per facilitar la interpretació.
- Beats: són els agents que recopilen informació als servidors, com Filebeat o Eventbeat.

## 2.24 Què és Splunk?

És el mateix que ELK però de pagament segons volum de dades, més intuïtiu, requereix menys configuració i té una corba d'aprenentatge menys pronunciada, amb suport empresarial.