

Ecco una possibile “banca” di domande, sia chiuse (a scelta multipla) sia aperte, nello **stile** di quelle che ti ha fatto il professore per WEB1/WEB2, ma sugli argomenti delle slide REST + Cloud. Ogni domanda ha la soluzione.

Domande chiuse – API, HTTP, REST

1. In un'API REST, quale verbo HTTP è più adatto per aggiornare **parzialmente** una risorsa esistente?

- A) GET
- B) POST
- C) PUT
- D) PATCH

Soluzione: D) PATCH.[1]

2. Quale status code HTTP rappresenta correttamente la creazione di una nuova risorsa?

- A) 200 OK
- B) 201 Created
- C) 204 No Content
- D) 404 Not Found

Soluzione: B) 201 Created.[1]

3. In un'API REST, l'endpoint '/users/123/orders/5' cosa rappresenta più probabilmente?

- A) Una collezione di utenti
- B) L'ordine con id 5 dell'utente con id 123
- C) Una risorsa generica non identificata
- D) Un'azione di login

Soluzione: B) L'ordine con id 5 dell'utente con id 123.[1]

4. Quale delle seguenti **non** è una caratteristica tipica delle architetture REST?

- A) Stateless
- B) Cacheable
- C) Tight coupling tra client e server
- D) Uniform interface

Soluzione: C) Tight coupling tra client e server.[1]

5. Un client invia una richiesta 'DELETE /products/10' e il server risponde '204 No Content'. Cosa significa?

- A) Errore lato client
- B) Il prodotto 10 non esiste
- C) Il prodotto 10 è stato eliminato e non c'è body in risposta
- D) Il server è momentaneamente non disponibile

Soluzione: C) Il prodotto è stato eliminato e non c'è body.[1]

6. Quale combinazione verbo/operazione è la più corretta in REST?

- A) GET – creare una risorsa
- B) POST – eseguire un'azione non idempotente di creazione
- C) PUT – ottenere una lista di risorse
- D) DELETE – aggiornare una risorsa

****Soluzione:** B) POST – creazione non idempotente.[1]**

7. Che cosa restituisce tipicamente un server REST ben progettato quando un client chiede una risorsa inesistente?

- A) 200 OK con body vuoto
- B) 400 Bad Request
- C) 404 Not Found
- D) 500 Internal Server Error

****Soluzione:** C) 404 Not Found.[1]**

8. Quale protocollo di trasporto è usato più comunemente dalle API REST?

- A) FTP
- B) SMTP
- C) HTTP/HTTPS
- D) SSH

****Soluzione:** C) HTTP/HTTPS.[2]**

Domande chiuse – CORS, Same Origin, Preflight

9. La Same-Origin Policy considera ****diversa origine**** se cambia:

- A) Solo il path dell'URL
- B) Solo il protocollo (http/https), l'host o la porta
- C) Solo i parametri di query
- D) Solo il fragment ('#')

****Soluzione:** B) Protocollo, host o porta diversi = origine diversa.[3]**

10. Una “simple request” CORS ****non**** richiede preflight se:

- A) È una richiesta GET/HEAD/POST con header semplici
- B) Usa sempre l'header 'Authorization'
- C) È sempre una richiesta POST con JSON
- D) È una richiesta PUT

****Soluzione:** A) Verbi “semplici” e header semplici → niente preflight.[3]**

11. Il preflight CORS è implementato tramite quale verbo HTTP?

- A) GET
- B) POST
- C) OPTIONS
- D) HEAD

****Soluzione:** C) OPTIONS.[3]**

12. Quale header ****deve**** essere presente nella risposta del server per permettere a un origin specifico di accedere?

- A) Content-Type
- B) Access-Control-Allow-Origin
- C) Access-Control-Request-Method
- D) Accept

****Soluzione:** B) Access-Control-Allow-Origin.[3]**

Domande chiuse – Richardson Maturity Model

13. Nel modello di maturità di Richardson, che cosa caratterizza il ****livello 0****?

- A) Uso corretto di HTTP e status code
- B) Un solo endpoint con uso di HTTP solo come tunnel
- C) Uso di HATEOAS
- D) Risorse con URI dedicati

****Soluzione:** B) Un solo endpoint “RPC-like”, HTTP come tunnel.[4]**

14. Quale livello introduce l'uso sistematico dei verbi HTTP (GET, POST, PUT, DELETE...) in modo coerente con le operazioni?

- A) Livello 0
- B) Livello 1
- C) Livello 2
- D) Livello 3

****Soluzione:** C) Livello 2.[4]**

15. A cosa è associato il livello 3 del modello di Richardson?

- A) Formati binari
- B) HATEOAS (link nel body per navigare l'API)
- C) Uso di SOAP
- D) Solo richieste GET

****Soluzione:** B) HATEOAS.[4]**

Domande chiuse – Cloud Computing base

16. Quale definizione descrive meglio il ****cloud computing**** secondo NIST?

- A) Un singolo server remoto per backup
- B) Modello per accedere on demand a risorse condivise via rete, con provisioning rapido e minimo sforzo di gestione
- C) Un servizio di posta elettronica
- D) Una VPN aziendale

****Soluzione:** B) Modello on demand di risorse condivise.[5]**

17. Quale delle seguenti è una caratteristica essenziale del cloud secondo NIST?

- A) Accesso solo da rete interna
- B) On demand self service
- C) Assenza di virtualizzazione
- D) Nessuna elasticità

****Soluzione:** B) On demand self service.[5]**

18. Quale modello di servizio cloud fornisce ****macchine virtuali, rete e storage****, lasciando al cliente il sistema operativo e sopra?

- A) SaaS

- B) PaaS
- C) IaaS
- D) FaaS

****Soluzione:** C) IaaS.[5]**

19. In quale modello l'utente usa direttamente un'applicazione completa gestita dal provider (es. Gmail, Salesforce)?

- A) IaaS
- B) PaaS
- C) SaaS
- D) On premise

****Soluzione:** C) SaaS.[5]**

20. Un'azienda utilizza un'infrastruttura cloud dedicata solo a quella organizzazione, ma ospitata dal provider. Qual è il modello di deployment?

- A) Public cloud
- B) Private cloud
- C) Hybrid cloud
- D) Community cloud

****Soluzione:** B) Private cloud.[5]**

Domande chiuse – Storage e altri concetti Cloud

21. Quale tipo di storage è più usato per oggetti come immagini, PDF, backup, indirizzati da un ID univoco?

- A) Block storage
- B) File storage
- C) Object storage
- D) Tape storage

****Soluzione:** C) Object storage.[5]**

22. In un cloud pubblico, la condivisione delle risorse tra più clienti si chiama:

- A) Single tenant
- B) Multi tenant
- C) Dedicated hosting
- D) Bare metal

****Soluzione:** B) Multi tenant.[5]**

Domande aperte – API, HTTP, CORS

23. Spiega cosa si intende per ****API REST**** e descrivi almeno tre principi che devono rispettare.

****Soluzione indicativa:****

- Una API REST espone risorse tramite URI e usa HTTP per operazioni standard.[1]

- Principi:

- ****Stateless****: ogni richiesta contiene tutte le info necessarie, il server non mantiene stato di

sessione.[1]

- Uniform interface: uso coerente di URI, verbi HTTP, status code e formati.[1]
- Cacheable: le risposte indicano se e per quanto possono essere messe in cache.[1]
- Client server separati: il client gestisce UI, il server dati/logica.[1]

24. Descrivi la differenza tra i verbi HTTP **GET**, **POST**, **PUT**, **PATCH** e **DELETE** in un'API REST, includendo per ciascuno un esempio di endpoint.

Soluzione indicativa:

- GET: legge una risorsa senza modificarla, es. 'GET /users/10'.[1]
- POST: crea una nuova risorsa nella collezione, es. 'POST /users'.[1]
- PUT: sostituisce interamente una risorsa esistente, es. 'PUT /users/10'.[1]
- PATCH: aggiorna parzialmente una risorsa, es. 'PATCH /users/10' per cambiare solo l'email.[1]
- DELETE: elimina una risorsa, es. 'DELETE /users/10'.[1]

25. Un browser deve chiamare da 'https://app.example.com' un'API a 'https://api.other.com'. Spiega:

- cos'è la **Same Origin Policy**,
- perché serve il **CORS**,
- il ruolo del preflight 'OPTIONS'.

Soluzione indicativa:

- La Same Origin Policy blocca per sicurezza le richieste JS verso origini diverse (protocollo, host o porta diversi).[3]
- CORS permette al server di dichiarare quali origini sono autorizzate tramite header come 'Access-Control-Allow-Origin'.[3]
- Per richieste non "semplici", il browser invia prima un preflight 'OPTIONS' con gli header proposti; il server risponde specificando metodi e header ammessi, e solo se la risposta è ok la richiesta vera viene inviata.[3]

26. Spiega cosa sono i **livelli di maturità** di Richardson per le API e descrivi brevemente i livelli 0, 1, 2 e 3 con un esempio per ciascuno.

Soluzione indicativa:

- È un modello che classifica quanto un'API sfrutta bene HTTP e lo stile REST.[4]
- Livello 0: un solo endpoint, ad esempio 'POST /api' con un campo 'action' che specifica cosa fare.[4]
- Livello 1: più URI che rappresentano risorse diverse, ma ancora uso non corretto dei verbi HTTP (si usa sempre POST).[4]
- Livello 2: URI per risorse + uso corretto di verbi e status code (GET per leggere, POST per creare, ecc.).[4]
- Livello 3: come il livello 2, ma le risorse contengono link per indicare le operazioni possibili (HATEOAS).[4]

Domande aperte – Cloud Computing

27. Definisci **cloud computing** secondo la visione NIST e illustra le cinque caratteristiche essenziali.

Soluzione indicativa:

- Cloud computing: modello che consente accesso comodo, on demand, via rete a un insieme condiviso di risorse configurabili (reti, server, storage, applicazioni, servizi).[5]
- Caratteristiche:
 - On demand self service.[5]
 - Broad network access (accesso tramite rete da vari dispositivi).[5]
 - Resource pooling (multi tenant).[5]

- Rapid elasticity (scalabilità rapida, anche automatica).[5]
- Measured service (risorse misurate e fatturate a consumo).[5]

28. Confronta i modelli **IaaS, PaaS e SaaS** dal punto di vista di cosa gestisce il provider e cosa gestisce il cliente. Fai un esempio di servizio per ogni modello.

Soluzione indicativa:

- IaaS: il provider gestisce infrastruttura fisica, virtualizzazione e spesso rete di base; il cliente gestisce OS, runtime, dati e applicazioni (es. Amazon EC2).[5]
- PaaS: il provider gestisce anche sistema operativo, middleware, runtime e spesso database; il cliente carica il proprio codice/applicazione (es. Heroku, App Engine).[5]
- SaaS: il provider gestisce tutta la stack fino all'applicazione; il cliente usa solo la funzionalità tramite browser o client (es. Gmail, Salesforce).[5]

29. Spiega cosa si intende per **public cloud, private cloud e hybrid cloud** e indica per ciascuno un possibile scenario d'uso.

Soluzione indicativa:

- Public cloud: infrastruttura condivisa tra più clienti, gestita dal provider, accessibile via Internet; tipico per startup che vogliono scalare velocemente.[5]
- Private cloud: infrastruttura dedicata a una sola organizzazione (on premise o ospitata), usata quando servono forti requisiti di controllo o compliance.[5]
- Hybrid cloud: combinazione di public e private cloud con integrazione tra i due, ad esempio dati sensibili su private e frontend scalabile su public.[5]

30. Un'azienda deve scegliere tra **object storage** e **block storage** per archiviare:

- backup di grandi volumi di dati,
 - database transazionale ad alte prestazioni.
- Spiega quale tipo di storage useresti per ciascun caso e perché.

Soluzione indicativa:

- Per i backup: object storage, perché è economico, scalabile e adatto a file grandi, immutabili o raramente modificati.[5]
- Per il database transazionale: block storage, perché fornisce I/O a bassa latenza e si presenta come disco a blocchi montato dal sistema operativo.[5]

Ecco un **simulatore d'esame completo** con **20 domande** (10 chiuse + 10 aperte), nello stile del professore, basate **solo** sulle slide REST API + Cloud Computing. Usa questo per prepararti: fai prima le domande **senza guardare le soluzioni**, poi verifica. Tempo consigliato: 45 minuti.

SIMULATORE ESAME - REST API & CLOUD COMPUTING

Istruzioni: Rispondi alle domande chiuse scegliendo una sola opzione. Per le aperte, sii preciso e cita esempi dalle slide.

DOMANDE CHIUSE (scegli A, B, C o D) [10 punti]

1. Quale verbo HTTP è **idempotente** (cioè, ripetere la richiesta ha lo stesso effetto)?

- A) POST
- B) PUT
- C) PATCH

D) Nessuna di queste

2. In una richiesta CORS **preflight**, quale header il **client** invia per specificare i metodi che intende usare?
- A) Access-Control-Allow-Methods
 - B) Access-Control-Request-Method
 - C) Content-Type
 - D) Origin
3. Secondo il **Richardson Maturity Model**, il **Livello 1** si concentra principalmente su:
- A) Uso corretto dei verbi HTTP
 - B) Identificazione delle risorse tramite URI
 - C) HATEOAS
 - D) Status code HTTP
4. Quale status code indica che una risorsa è stata **creata con successo** da un POST?
- A) 200 OK
 - B) 201 Created
 - C) 202 Accepted
 - D) 204 No Content
5. Nel cloud computing NIST, quale caratteristica permette di scalare risorse automaticamente senza intervento umano?
- A) On-demand self-service
 - B) Rapid elasticity
 - C) Resource pooling
 - D) Measured service
6. Quale modello di servizio cloud **non** gestisce il sistema operativo per il cliente?
- A) IaaS
 - B) PaaS
 - C) SaaS
 - D) FaaS
7. Un endpoint REST `/api/v1/users/42/orders` rappresenta più probabilmente:
- A) Tutti gli ordini di un utente
 - B) Creare un nuovo utente
 - C) Eliminare un ordine
 - D) Loggarsi
8. Quale header CORS **il server** deve includere nella risposta per permettere `https://client.com` di leggere dati?
- A) Access-Control-Allow-Origin: `https://client.com`
 - B) Origin: `https://client.com`
 - C) Allow-Origin: `*`
 - D) CORS-Allow-All
9. In quale deployment model cloud le risorse sono **condivise tra più organizzazioni** ma dedicate a una specifica community?
- A) Public
 - B) Private
 - C) Community
 - D) Hybrid
10. Quale tipo di storage cloud è ottimale per **video streaming** e grandi file non strutturati?

- A) Block storage
- B) File storage (NFS)
- C) Object storage
- D) Local disk

DOMANDE APERTE [20 punti - 2 punti ciascuna]

11. **Definisci "stateless"** in un contesto REST e spiega perché è importante per la scalabilità.
12. **Descrivi una "simple request" CORS** vs **preflight request**. Quando si usa ciascuna?
13. **Elenca i 4 livelli del Richardson Maturity Model** con **un esempio pratico** per il Livello 2.
14. **Qual è la differenza tra PUT e PATCH**? Dai un esempio JSON per `/users/123`.
15. **Quali sono le 5 caratteristiche essenziali del cloud** secondo NIST? (elenca solo i nomi).
16. **Confronta IaaS, PaaS e SaaS** con un esempio di servizio reale per ciascuno.
17. **Spiega Same-Origin Policy**: cosa blocca e perché serve?
18. **Cosa fa un OPTIONS preflight**? Quali header **riceve** il server dalla richiesta?
19. **Public vs Private vs Hybrid cloud**: pro e contro di ciascuno (1 riga per modello).
20. **Object storage vs Block storage**: usa caso, caratteristiche e un esempio provider.

SOLUZIONI COMPLETE (guarda solo DOPO aver risposto!)

DOMANDE CHIUSE

1. **B) PUT** - PUT, GET, DELETE sono idempotenti; POST e PATCH no.[1]
2. **B) Access-Control-Request-Method** - Il client chiede, il server autorizza con Allow.[2]
3. **B) Identificazione delle risorse tramite URI** - Livello 1 = risorse, Livello 2 = verbi HTTP.[3]
4. **B) 201 Created** - POST crea → 201 con Location header.[1]
5. **B) Rapid elasticity** - Scalabilità rapida e automatica.[4]
6. **A) IaaS** - Cliente gestisce OS e sopra.[4]
7. **A) Tutti gli ordini di un utente** - `/users/{id}/orders` = collezione.[1]
8. **A) Access-Control-Allow-Origin: https://client.com** - Specifica l'origine permessa.[2]
9. **C) Community** - Es. cloud per enti governativi.[4]
10. **C) Object storage** - S3, blob storage per media.[4]

DOMANDE APERTE (risposte modello)

11. **Stateless**: ogni richiesta HTTP deve contenere tutte le informazioni necessarie per essere processata; il server non mantiene stato di sessione tra chiamate. Importante per scalabilità: ogni richiesta può andare a qualsiasi server del cluster senza sincronizzazioni.[1]
12. **Simple request**: GET/HEAD/POST con header standard (no custom), no preflight. **Preflight**: per PUT/DELETE, POST con custom header o Content-Type non standard; browser invia prima OPTIONS per chiedere permesso.[2]

13. ****Livelli****: 0 (tunnel RPC), 1 (risorse URI), 2 (****verbi HTTP + status**** es. 'GET /users/42' → 200, 'POST /users' → 201), 3 (HATEOAS).[3]

14. ****PUT****: sostituisce ****tutta**** la risorsa. ****PATCH****: aggiorna solo campi specifici.
Es. PATCH '/users/123': '{"email": "new@example.com"}' (solo email cambia).[1]

15. 1. On-demand self-service, 2. Broad network access, 3. Resource pooling, 4. Rapid elasticity, 5. Measured service.[4]

16. ****IaaS**** (AWS EC2): gestisce hardware. ****PaaS**** (Heroku): gestisce OS/runtime. ****SaaS**** (Google Workspace): gestisce tutto.[4]

17. ****Same-Origin Policy****: browser blocca JS da leggere/scrivere risorse di dominio diverso (protocollo+host+porta) per prevenire attacchi cross-site (XSS, CSRF).[2]

18. ****OPTIONS preflight****: client invia 'Origin', 'Access-Control-Request-Method', 'Access-Control-Request-Headers'. Server risponde con 'Allow-Origin', 'Allow-Methods', 'Allow-Headers', 'Max-Age'.[2]

19. ****Public****: pro scalabilità/costo basso, contro minor controllo. ****Private****: pro sicurezza/compliance, contro costo alto. ****Hybrid****: pro flessibilità, contro complessità integrazione.[4]

20. ****Object****: file grandi immutabili (S3 per backup/video), accesso via HTTP/URL. ****Block****: dischi performanti per DB/OS (EBS), montati come volume.[4]

****Punteggio perfetto**** Sei pronto! Vuoi altre 20 domande diverse, un focus su CORS/REST/cloud, o domande con codice React dalle slide? Dimmi!

[1](https://roadmap.sh/questions/rest-api)

[2](https://www.browserless.io/blog/rest-api-interview-questions-answers-2025)

[3](https://www.merge.dev/blog/rest-api-interview-questions)

[4](https://www.sanfoundry.com/1000-cloud-computing-questions-answers/)

[5](https://ppl-ai-file-upload.s3.amazonaws.com/web/direct-files/attachments/109239915/bf5cc321-5f53-4403-bb24-409dbb481456/Progettazione-REST-ITS-ICT-Academy.potx.pptx-3-3.pdf)

[6](https://ppl-ai-file-upload.s3.amazonaws.com/web/direct-files/attachments/109239915/4ecef017-1772-4412-8c74-82197078a802/Progettazione-2-Le-basi-del-Cloud-ITS-ICT-Academy-1.pdf)

[7](https://ppl-ai-file-upload.s3.amazonaws.com/web/direct-files/attachments/109239915/8fcf1aad-a9bb-486a-8f50-2a57a70d2657/Progettazione-REST-ITS-ICT-Academy.potx.pptx-3-3.pdf)

[8](https://ppl-ai-file-upload.s3.amazonaws.com/web/direct-files/attachments/109239915/d04b8920-fb9d-4b8d-baaa-c8a29565ed15/Progettazione-2-Le-basi-del-Cloud-ITS-ICT-Academy-1.pdf)

[9](https://ppl-ai-file-upload.s3.amazonaws.com/web/direct-files/attachments/images/109239915/269da99d-30d3-4443-a9b9-a7ec7d4308f6/IMG_1231.jpg)

[10](https://ppl-ai-file-upload.s3.amazonaws.com/web/direct-files/attachments/images/109239915/3b31e189-f58f-4870-b7cd-8598f45b2aa7/IMG_1229.jpg)

[11](https://ppl-ai-file-upload.s3.amazonaws.com/web/direct-files/attachments/images/109239915/75c29529-a4f1-40aa-9b07-196f10f8e786/IMG_1228.jpg)

[12](https://ppl-ai-file-upload.s3.amazonaws.com/web/direct-files/attachments/images/109239915/a02a8cb0-65af-4ac7-b379-1931e75a2ac0/IMG_1230.jpg)