1) Cosa si intende per API REST ?

REST è un’architettura software volta alla trasmissione di dati nei sistemi distribuiti attraverso il protocollo http, quindi attraverso le chiamate GET, POST, PUT, DELETE, ecc...

Un’api invece è un’interfaccia che permette ad un programmatore di utilizzare determinate funzioni ottenendo un certo livello di astrazione, senza doversi preoccupare di come funziona ciò che c’è dietro.

Un’API REST è quindi un’ API che rispetta tutti i principi di un servizio REST e i suoi controlli sono basati quindi su http.

2) Cosa si intende per servizio SOAP ?

SOAP in realtà è un protocollo che permette lo scambio di informazioni tra un sistemi distribuiti tramite l’uso di messaggi strutturati in XML, senza tener conto se i sistemi siano stati sviluppati con tecnologie e linguaggi differenti.

Parliamo di servizi SOAP nel momento in cui ci troviamo a parlare di un web service che utilizza questo protocollo per lo scambio di informazioni.

A differenza di REST, SOAP diventa un protocollo molto utile nel momento in cui bisogna effettuare uno scambio di informazioni che non sia stateless o uno scambio di informazioni più sicuro.

3) Cos'è un database relazionale ?

Un’ insieme organizzato di dati, gestiti da un DBMS che ne garantisce determinate proprietà, basato sul modello relazionale.

In un DB relazionale i dati vengono organizzati tramite l’uso delle tabelle.

Per la creazione di queste, per l’inserimento, la manipolazione e il prelievo di questi dati viene usato un linguaggio standard chiamato SQL.

Ogni DBMS poi effettua le sue personalizzazioni per favorirne l’uso al programmatore.

Tra i DBMS relazionali più usati troviamo (Oracle, MySql, SQL Server, PostgreSQL, ecc…)

Tra le caratteristiche principali rispettate da un DBMS relazionale vanno nominate le proprietà ACID, ovvero quelle proprietà che ci garantiscono caratteristiche quali la permanenza dei dati una volta scritti nel database, l’atomicità delle operazioni che effettuiamo, e quindi che le operazioni di un utente siano disgiunte e non vadano in conflitto con operazioni effettuate da altri utenti o ancora la loro completezza, e quindi la garanzia che un’operazione o viene eseguita in toto o non ne viene eseguito nemeno una parte.

4) Cos'è un database NoSQL ?

Come scritto prima parliamo di un’insieme organizzato di dati gestiti da un DBMS, ma questa volta non ci si baserà sul modello relazionale per la rappresentazione di questa gestione. I modelli utilizzati dai DBMS NoSQL variano da strutture dati quali i grafi al modello chiave-valore.

Tra questi DBMS i più conosciuti sono MongoDB, Neo4j, ecc…

A differenza dei DBMS relazionali non sempre i NoSQL DBMS tengono fede a proprietà ACID nelle loro transazioni.

Cos'è un ORM ? Fai almeno un esempio.

È un’interfaccia che permette allo sviluppatore di creare astrazione tra l’applicativo e il DB permettendo quindi di gestire le entità e le relazioni nel DBMS tramite il linguaggio di programmazione utilizzato per l’applicativo.

Un’ esempio di ORM è Eloquent utilizzato in Laravel per il linguaggio PHP.

Eloquent permette, una volta definite la classe del modello equivalente all'entità del DB, di chiamre delle funzioni attraverso l'interfaccia definita dal programmatore semplificando l'uso dell'applicativo.
Ad esempio, una volta definita l'interfaccia User potremo usare User::all() anziché la query "SELECT * FROM User".

6) Cos'è la SQL Injection ?

Si tratta di un attacco ai danni di un applicativo web che si basa su DBMS relazionali.
Consiste nello sfruttare un mancato controllo sull'input di informazioni da parte dell'utente per poter effettuare operazioni che permettano la lettura, o la modifica o ancora la cancellazione di dati nel database. In casi estremi potrebbe addirittura fungere da ponte per l'OS del server in cui è in esecuzione l'applicativo web.
Supponendo di avere un form che ci chiede l'inserimento del nome, un attaccante potrebbe volontariamente inserire un carattere come un apice singolo per controllare se il servizio restituisce un errore SQL, ed a quel punto l'attaccante saprebbe che potrebbe essere in grado di manipolare la nostra query SQL a suo favore.
Possiamo parlare di Blind SQL injection nel momento in cui l'attaccante non è in grado di avere un feedback visivo di eventuali successi o fallimenti dei suoi tentativi.
Nell'esempio di prima non necessariamente potremo vedere visivamente l'errore generato da SQL lato client ma comunque potremmo provare a manipolare lo stesso la query. Questo è un esempio di Blind injection.
I controlli sull'input lato server sono un'arma di difesa contro l'SQL Injection. Impedendo di immettere determinati caratteri, o effettuando un escape dei caratteri prima di passarli alla query.

7) Cos'è l'autenticazione a 2 fattori? Descrivi brevemente un esempio.

In ambito di Internet Security o Web security l'autenticazione può basarsi su:
- Conoscenza(Password, pin, passphrase, ecc…)
- Possesso(carte magnetiche, token, smart card, ecc…)
- Biometria(impronte, iride, ecc…)

Nel momento in cui combiniamo due di questi fattori otteniamo un autenticazione a 2 fattori.
Potremmo ad esempio combinare la conoscenza di una password al possesso di un numero di cellulare.
In questo modo anche qual'ora si conoscesse la password per accedere ad un determinato servizio sarebbe comunque necessario autenticarsi per possesso rispondendo correttamente ad una domanda che richiede un pin inviato tramite SMS.

8) Descrivi brevemente un metodo sicuro per salvare le password degli utenti sul DB.

Un metodo potrebbe essere l'utilizzo di hashing combinato all'aggiunta di sale alla password dell'utente.
Quindi una volta presa la password immessa dall'utente alla registrazione possiamo aggiungere una stringa randomica di x bit alla password e calcolarne l'hash.
A questo punto nel DB verrano salvati il sale in chiaro per poter effettuare il controllo all'accesso dell'utente e l'hash del sale concatenato alla password.
Sarebbe opportuno utilizzare un algoritmo di hashing SHA-256 o superiore.

9) Cos'è una Sticky Session in un sistema con Load Balancing?