■ JavaScript Promises – Interview Q&A;

1. What is a Promise in JavaScript?
A Promise is an object that represents the eventual result of an asynchronous operation.
States: Pending, Fulfilled, Rejected.

2. Why do we need Promises?
- Avoid callback hell
- Cleaner async code
- Better error handling

3. Difference between Callbacks and Promises?
Callbacks: nested, hard to read.
Promises: chainable with .then().

4. What are .then(), .catch(), and .finally()?
- .then(): handles resolved value
- .catch(): handles errors
- .finally(): always runs

5. What happens if you don't handle a rejected Promise?
You get an 'UnhandledPromiseRejection' warning.

6. What is Promise Chaining?
Passing result of one .then() into the next.

7. Difference between resolve and reject?
- resolve(value): success
- reject(error): failure

8. Explain Promise.all()
Runs multiple promises in parallel, resolves if all succeed, rejects if one fails.

9. Explain Promise.race()
Settles as soon as the first promise resolves or rejects.

10. Explain Promise.allSettled()
Waits for all promises to settle (fulfilled or rejected). Returns array of results.

11. Explain Promise.any()
Resolves as soon as one promise fulfills. Rejects only if all fail.

12. Microtasks vs Macrotasks?
Promises use microtasks (executed before setTimeout macrotasks).

13. What happens if you return a value from .then()?
It is passed to the next .then().

14. What happens if you return a Promise from .then()?
The next .then() waits until that promise settles.

15. Difference between .then/.catch and async/await?
Async/await is cleaner syntax, but internally uses Promises.

16. Can Promises be cancelled?

No true cancellation. Use AbortController or flags as workarounds.