@name Nullish coalescing not nullable

@description

This diagnostic detects a useless nullish coalescing operator (`??`) in Angular templates. Specifically, it looks for operations where the input is not "nullable", meaning its type does not include `null` or `undefined`. For such values, the right side of the `??` will never be used.

```
import {Component} from '@angular/core';

@Component({
  // Template displays `foo` if present, falls back to 'bar' if it is `null`
  // or `undefined`.
  template: `<div>{{ foo ?? 'bar' }}</div>`,
  // ...
})
class MyComponent {
  // `foo` is declared as a `string` which *cannot* be `null` or `undefined`.
  foo: string = 'test';
}
```

What's wrong with that?

Using the nullish coalescing operator with a non-nullable input has no effect and is indicative of a discrepancy between the allowed type of a value and how it is presented in the template. A developer might reasonably assume that the right side of the nullish coalescing operator is displayed in some case, but it will never actually be displayed. This can lead to confusion about the expected output of the program.

What should I do instead?

Update the template and declared type to be in sync. Double check the type of the input and confirm whether it is actually expected to be nullable.

If the input should be nullable, add `null` or `undefined` to its type to indicate this.

```
import {Component} from '@angular/core';

@Component({
  template: `<div>{{ foo ?? 'bar' }}</div>`,
  // ...
})
class MyComponent {
  // `foo` is now nullable. If it is ever set to `null`, 'bar' will be displayed.
  foo: string | null = 'test';
}
```

If the input should *not* be nullable, delete the `??` operator and its right operand, since the value is guaranteed by TypeScript to always be non-nullable.

```
import {Component} from '@angular/core';

@Component({
  // Template always displays `foo`, which is guaranteed to never be `null` or
  // `undefined`.
  template: `<div>{{ foo }}</div>`,
  // ...
})
class MyComponent {
  foo: string = 'test';
}
```

Configuration requirements

`strictTemplates` must be enabled for any extended diagnostic to emit. `strictNullChecks` must also be enabled to emit any `nullishCoalescingNotNullable` diagnostics.

What if I can't avoid this?

This diagnostic can be disabled by editing the project's `tsconfig.json` file:

```
{
  "angularCompilerOptions": {
    "extendedDiagnostics": {
      "checks": {
        "nullishCoalescingNotNullable": "suppress"
      }
    }
  }
}
```

See extended diagnostic configuration for more info.