Optional: Class-based Guards

The previous lecture introduced you to guards using the modern, recommended function-based approach.

But you can also use an older (deprecated!) class-based approach:

```
@Injectable({ providedIn: 'root' })
class CanMatchTeamSection implements CanMatch {
  constructor(private router: Router) {}
  canMatch(route: Route, segments: UrlSegment[]) {
    const shouldGetAccess = Math.random();
    if (shouldGetAccess < 0.5) {
      return true;
    }
    return new RedirectCommand(this.router.parseUrl('/unauthorized'));
  }
}
```

And attach the guard to a route like this:

```
{
  path: 'users/:userId', // <your-domain>/users/<uid>
  component: UserTasksComponent,
  children: userRoutes,
  canMatch: [CanMatchTeamSection],
  data: {
    message: 'Hello!',
  },
  resolve: {
    userName: resolveUserName,
  },
  title: resolveTitle,
},
```