A FULLSTACK REACT *MASTERCLASS*
TINYHOUSE

Quiz Questions

Module 3 | Lesson 3.3 - Querying and mutating listings data with GraphQL

1. Which of the following statements best describe what the GraphQL `ID` scalar type is?

A: It is used to represent the type of a field as a double-precision floating-point value.
B: It is used to represent a unique identifier and gets serialized as the GraphQL `String` scalar type.
C: It is identical to the GraphQL `String` type and can be used interchangeably.
D: GraphQL `ID` is not a default scalar type.

2. Why would the following `name` field throw an error when queried in certain cases?

```
const query = new GraphQLObjectType({
  name: "Query",
  fields: () => ({
    name: {
      type: GraphQLNonNull(GraphQLString),
      args: {
        id: { type: GraphQLID }
      },
      resolve: (obj, args) => {
        const { id } = args;
        return id ? getUser(id).name : null;
      }
    }
  })
});
```

A: Querying the `name` field will always return either the `name` of the appropriate user or `null` so it won't ever throw an error.
B: GraphQL forbids returning `null` values from object fields.
C: We've used the `GraphQLNonNull` wrapper to state that the `name` field should always be defined and never to be `null`. When an `id` argument isn't provided, we return `null` from our resolver function which will trigger a GraphQL execution error.
D: The `GraphQLNonNull` wrapper does not exist in the GraphQL JS library.

3. Which of the following defines a potentially `null` list type where the values in the list can never be `null` but of type `User`?

A: `GraphQLNonNull(GraphQLList(GraphQLNonNull(User)))`
B: `GraphQLNonNull(User)`
C: `GraphQLList(User)`
D: `GraphQLList(GraphQLNonNull(User))`