**Search**

**ORM:**

ORM stands for Object-Relational Mapping. It’s a programming technique that allows developers to interact with a database using an object-oriented paradigm. Instead of writing raw SQL queries, developers can use their programming language's classes and objects to represent database tables and records.

Here are some key points about ORM:

1. **Abstraction**: ORM abstracts the database interactions, making it easier to work with complex data models without needing to write SQL.
2. **Data Mapping**: It maps classes to database tables and class attributes to table columns, allowing for a more intuitive way to manipulate data.
3. **Productivity**: By reducing boilerplate code and allowing developers to work with high-level objects, ORM can enhance productivity.

**Mass Assignment**

Mass assignment is a programming technique commonly used in web development to simplify the process of assigning multiple properties to an object at once, typically in the context of forms or API requests. Instead of assigning each property individually, developers can pass an entire array or object of attributes, which the system then uses to populate the object.

**Key Points:**

1. **Efficiency**: It reduces the amount of code needed to create or update an object, making the code cleaner and easier to read.
2. **Common in Frameworks**: Many web frameworks (like Ruby on Rails, Laravel for PHP, and Django for Python) support mass assignment, allowing developers to map form inputs directly to model attributes.
3. **Security Risks**: One of the main concerns with mass assignment is security. If not properly managed, it can lead to vulnerabilities like **mass assignment vulnerabilities**, where users can set attributes they shouldn't have access to, potentially compromising data integrity or security.
4. **Strong Parameters**: To mitigate these risks, frameworks often implement features like "strong parameters" or "whitelisting," where developers specify which attributes are safe for mass assignment.

**Create vs Make:**

|  |  |
| --- | --- |
| Create | Make |
| The create method is used to create a new record in the database | The make method is used to create an instance of a model without saving it to the database. |
| It takes an array of attributes, saves the model to the database, and returns the instance of the created model. | It is typically used when you want to instantiate a model object, often for validation or manipulation, but you don't want to persist it yet. |

**PUT vs PATCH:**

|  |  |
| --- | --- |
| PUT | PATCH |
| used to update or replace a resource at a specific URL | used to apply partial modifications to a resource |
| It typically requires the client to send the entire representation of the resource. If any attributes are omitted, they may be removed from the resource | It only requires the client to send the changes (i.e., the fields that need to be updated). It does not require the entire resource representation |