Great one 👍 — this is **WSTG-INPV-07: Testing for ORM Injection**.  
Since you already saw **SQLi** and **NoSQLi**, now we move into frameworks that use **Object Relational Mappers (ORMs)**.

**🧪 Testing for ORM Injection**

**🔎 What is it?**

* **ORM Injection** happens when input is passed unsafely into an ORM query method.
* ORMs (like **Hibernate, Doctrine, SQLAlchemy, Django ORM, Rails ActiveRecord**) map database tables into objects.
* If developers build queries dynamically from user input (instead of parameterized ORM methods), injection is possible.

**⚡ Example Scenarios**

**1. Django ORM**

# vulnerable code

User.objects.raw("SELECT \* FROM users WHERE username = '%s'" % uname)

Attacker input:

' OR '1'='1

**2. SQLAlchemy (Python)**

# vulnerable

session.execute("SELECT \* FROM users WHERE id = %s" % user\_input)

Payload:

1; DROP TABLE users--

**3. Rails ActiveRecord (Ruby)**

# vulnerable

User.where("name = '#{params[:name]}'")

Payload:

' OR '1'='1

**🔧 Testing with Burp Suite**

1. Intercept login / search parameters.
2. Try classic SQLi payloads (' OR 1=1--, ') OR ('a'='a) → if ORM is not safe, you’ll see success.
3. Look for error messages:
   * ActiveRecord::StatementInvalid (Rails)
   * Doctrine\DBAL\Exception (PHP)
   * sqlalchemy.exc.StatementError (Python)

**🔧 Testing with sqlmap**

Even though it’s **ORM**, under the hood it still runs SQL. If injection is possible, **sqlmap can detect it**.

Example:

sqlmap -u "http://target.com/user?id=1" --dbms=mysql --level=5 --risk=3

* sqlmap sends various payloads (' OR '1'='1--, time delays, UNION SELECT).
* If ORM-generated SQL is vulnerable, sqlmap will exploit it.

**🔍 Signs of ORM Injection**

* Errors referencing ORM frameworks.
* Different behavior when sending injected queries.
* Time delays (sleep(5)) proving query manipulation.

**✅ Mitigation**

* Always use ORM **safe query methods**:
  + Django: User.objects.get(username=uname)
  + SQLAlchemy: session.query(User).filter(User.id == id).all()
* Avoid **string concatenation** inside ORM queries.
* Apply **input validation** and least-privilege DB accounts.

⚡ **Summary**:

* ORM Injection = SQL Injection via ORM frameworks.
* Burp Suite helps with fuzzing, error detection.
* **sqlmap still works**, because ORM eventually runs SQL.
* Mitigation = parameterized ORM queries only.

👉 Do you want me to also include **specific sqlmap payloads that are more effective against ORM frameworks** (beyond the default ones), so you can test them quickly in labs?