

3.1

VALIDACIÓN DE ENTRADAS

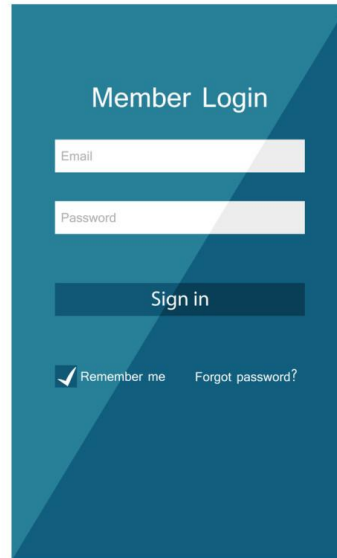
CUALQUIER DATO INTRODUCIDO POR UN USUARIO **ES SUSCEPTIBLE DE OCASIONAR ERRORES**

Es de vital importancia controlar la inserción de datos a nuestro software

VALIDACIÓN DE ENTRADAS

```
print "<html>"
print "Latest comment:"
print database.latestComment
print "</html>"
```

```
<html>
Latest comment:
<script>...</script>
</html>
```



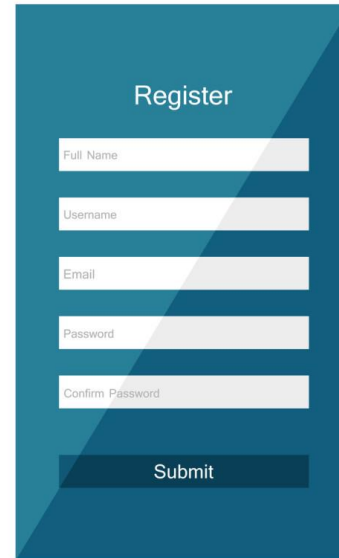
Member Login

Email

Password

[Sign in](#)

☒ Remember me [Forgot password?](#)



Register

Full Name

Username

Email

Password

Confirm Password

[Submit](#)

SQLi - INYECCIÓN DE CÓDIGO SQL

- ▶ Ataque de inyección de código
- ▶ Permite a un atacante ejecutar código SQL sobre nuestra base de datos

```
SELECT * FROM users WHERE username='blabla' or '1'='1';
```

VALIDACIÓN DE **FICHEROS SUBIDOS**

- ▶ Si nuestro software o aplicación permite subir ficheros a un determinado servidor, debemos cuidar la seguridad de esta funcionalidad
- ▶ Un atacante podría tomar el control de nuestra máquina mediante este tipo de ataques

VALIDACIÓN DE **ENTRADAS**

- ▶ Realizar todas las validaciones en un sistema de confianza
- ▶ Especificar la codificación de caracteres para todas las entradas (UTF-8...)
- ▶ Identifica **todas** las entradas de datos y clasifícalas en confiables y no confiables. Valida las no confiables
- ▶ Crea una rutina centralizada de validación de entradas para tu aplicación

VALIDACIÓN DE **ENTRADAS**

- ▶ *Canonicaliza* los datos antes de validarlos(codifica todos igual)
- ▶ **Cualquier fallo de validación debe incurrir en el rechazo de la entrada**
- ▶ Valida **todos los datos** que provenga del cliente
- ▶ Valida rangos de datos, tamaño de datos y tipos de datos
- ▶ Si es posible, valida los datos empleando una lista blanca

VALIDACIÓN DE **ENTRADAS**

- ▶ Si es necesario para tu aplicación utilizar caracteres *raros*, asegurate de implementar controles de entrada y salida específicos para estos