## Wie kann ich die theoretischen Konzepte anwenden, um Angriffe zu verhindern?

In diesem Abschnitt werden einige der im Abschnitt Sicherheitsmaßnahmen vorgestellten Konzepte praktisch umgesetzt.

1. Nachdem du jetzt die Konzepte der NoSQL-Injections verstanden und gesehen hast, wie man einen Angriff durchführen kann, wollen wir in diesem ersten Abschnitt die unsichere Login-Seite sicher machen. Dafür liegt im PHP-Playground die Version „login.php“. Diese Version besitzt keine Schutzmaßnahmen der zuvor vorgestellten Schutzmaßnahmen und du hast jetzt die Möglichkeit ihr neu erlerntes Wissen einzusetzen. Versuche selbst die Methoden wie Search and Replace und Type Casting umzusetzen und danach im Zusammenhang mit BurpSuite einen Angriff durchzuführen und die Wirksamkeit der Schutzmaßnahmen zu testen.

Unabhängig davon, ob deine eigene Umsetzung funktioniert hat oder nicht, findest du im folgenden Abschnitt mehrere Beispiele, wie Schutzmaßnahmen umzusetzen sind.

1. Als erstes Beispiel steht eine Login Website mit den Funktionen [Search and Replace](https://www.php.net/manual/en/function.preg-replace.php) und [TypeCasting](https://www.php.net/manual/en/language.types.type-juggling.php) mit PHP zur Verfügung. Search and Replace kann in PHP mit folgender Funktion umgesetzt werden: „*preg\_replace("/[^a-zA-Z0-9\_-]/", "", $input)“ .* Type Casting kann mit dem gewünschten Datentyp vor der Variable umgesetzt werden: „*(string)$input“* . Diese beiden Methoden beziehen sich auf das unschädlich machen des Nutzerinputs und können ohne weitere Schutzmaßnahmen keine absolute Sicherheit vor Angriffen garantieren.

Versuche diese Funktionen selber umzusetzen oder schaue dir das erste Codebeispiel zu den Schutzmaßnahmen ist im PHP-Playground unter „loginPHPFunctions.php“ zu finden.

1. Die Überprüfung kann auch mit unterschiedlichen verfügbaren Packages umgesetzt werden. Paketbeispiele für MongoDB: z.B. Mongoose, mongo-sanitize. In diesem zweiten Beispiel wurde ein Implementierung mit mongo-sanitize umgesetzt.
   1. Für die PHP-Umgebung im Beispiel wurde die [mongo-sanitize](https://github.com/Aldin-SXR/mongo-sanitize) PHP-Version genutzt. Diese wird über den Dependency Manager [composer](https://getcomposer.org/) installiert und kann dann in PHP genutzt werden. Mit der Nutzung eines Packages spart man sich die selbstständige Implementierung und unterschiedliche je nach Package werden umgesetzt, um den Input zu sichern. Dafür muss das Package nach dem Installieren geladen werden und dann kann die Funktion genutzt werden.

Der angegebene Pfad ist je nach Installationsort von Composer anzupassen.



Nachdem das Package installiert wurde und geladen ist, können die Funktionen benutzt werden.  Versuche diese Funktionen selber umzusetzen oder schaue dir das zweite Codebeispiel zu den Schutzmaßnahmen ist im PHP-Playground unter „loginMongoSanitize.php“ zu finden.

1. Pentesting & Security Audits Beispiel mit Umsetzung von [Mongoaudit](https://github.com/stampery/mongoaudit):

Nachdem die beiden zuvor genannten Beispiele umgesetzt wurden, kann man auch die Sicherheit seiner Anwendung überprüfen. Dafür gibt es unterschiedliche Möglichkeiten. In diesem Fall nutzen wir Mongoaudit. Dieses Tool überprüft die Sicherheit der Datenbank auf bekannte Bugs und auch andere relevante Sicherheitseinstellungen. Mit diesem Tool kann man also zusätzlich zum Überprüfen des Inputs auch die serverseitige Sicherheit überprüfen/verbessern.

Um Mongoaudit zu nutzen, muss Python installiert sein und danach kannst du das Github Repository klonen und installieren. Dann wird man durch unterschiedliche Tests geführt und kann abhängig vom Ergebnis Einstellungen ändern.