3.3 Beveiliging en authenticatie

- Hoe wordt de authenticatie en autorisatie geregeld?
    - Gebruikers moeten inloggen via **beveiligde sessies**.
    - Wachtwoorden worden **gehasht** met **bcrypt**.
- Welke beveiligingsmaatregelen worden getroffen tegen aanvallen (bijv. SQL-injectie, XSS)
    - SQL-injectiepreventie
    - XSS-bescherming

```php
<?php
include 'config.php';
session_start();

if ($_SERVER["REQUEST_METHOD"] == "POST") {
    $gebruikersnaam = $_POST['gebruikersnaam'];
    $wachtwoord = $_POST['wachtwoord'];

    $sql = "SELECT * FROM gebruikers WHERE gebruikersnaam = ?";
    $stmt = $conn->prepare(query: $sql);

    if ($stmt) {
        $stmt->bind_param(types: "s", var: &$gebruikersnaam);
        $stmt->execute();
        $result = $stmt->get_result();

        if ($result->num_rows > 0) {
            $row = $result->fetch_assoc();

            if (password_verify(password: $wachtwoord, hash: $row['wachtwoord'])) {
                $_SESSION['user_id'] = $row['id'];
                header(header: "Location: forum.html");
                exit();
            } else {
                echo "Ongeldige gebruikersnaam of wachtwoord.";
            }
        } else {
            echo "Ongeldige gebruikersnaam of wachtwoord.";
        }

        $stmt->close();
    } else {
        echo "Fout bij het voorbereiden van de query: " . $conn->error;
    }

    $conn->close();
}

?>
```

- Hoe wordt dataversleuteling toegepast?

- Wachtwoorden worden gehasht met bcrypt zoals hierboven (Zie afbeelding) zodat er geen exposure is aan gebruikers als ze bijv. willen inloggen

Prestatie- en beschikbaarheidseisen

- Hoe wordt de systeemprestatie gemonitord en geoptimaliseerd

- Bijvoorbeeld/Voorstel: Gebruik van **Google Analytics** of **New Relic** om te meten hoe lang pagina's laden en fouten opsporen.

- Hoe wordt de belasting verdeeld (bijv. load balancing)
- Bijvoorbeeld: Grote websites gebruiken **load balancers** om het verkeer te verdelen over meerdere servers, zodat het systeem niet vastloopt.

3.5 Deployment en onderhoud

- Hoe wordt het systeem uitgerold en geüpdatet
    - Versiebeheer bijhouden met nieuwe ideeën
    - **Versie beheer** via: bijvoorbeeld GitHub
    - **Repository**: https://github.com/username/project)
    - **Gebruik van Gitflow**: Er worden **feature branches** gebruikt voor elke nieuwe functie, en de code wordt pas gemerged naar de **main branch** na een code review.

- Welke infrastructuur wordt gebruikt (bijv. cloud, on-premise)
    - Bijvoorbeeld: Een klein forum kan draaien op een eenvoudige **shared hosting** server, terwijl een grote webshop een **cloud-oplossing** nodig heeft.

- Hoe wordt logging en foutafhandeling ingericht
    - Bijvoorbeeld: **Logbestanden** (zoals error logs in PHP) of externe tools zoals **Sentry** om crashes automatisch te detecteren.

- Wat is het onderhoudsplan en de verwachte levenscyclus van het systeem
    - Bijvoorbeeld: Een forumsoftware kan elk jaar een grote update krijgen, terwijl een kortlopend project misschien geen onderhoud nodig heeft.
    - Dit geval → nieuwe features indien nodig