DIGITAL SCRAMBLEPAD

Geïntegreerde proef 2022-2023

André da Cruz Ribeiro

T.I Don Bosco Hoboken

VOORWOORD

Hallo, mijn naam is André da Cruz Ribeiro en ik ben een leerling aan Don Bosco Hoboken. Ik zit momenteel in het zesde jaar Industriële Informatica (6TIIC).

Ik presenteer mijn Geïntegreerde Proef over digitale scramblepads als veilig alternatief voor traditionele keypads. In mijn GIP bespreek ik de ontwikkeling van een digitale scramblepad.

Ik heb dit project gekozen vanwege de voordelen van digitale scramblepads ten opzichte van traditionele keypads, zoals verbeterde veiligheid en flexibiliteit.

Graag wil ik ook de heer Van Reck en de heer Thijs bedanken, die mij gedurende dit project hebben begeleid en waardevolle feedback hebben gegeven.

Ook wil ik graag mijn dank uitspreken aan al mijn klasgenoten die hebben bijgedragen aan de totstandkoming van mijn GIP.

INHOUDSOPGAVE

INLEIDING

Het doel van mijn GIP is om een veiliger keypad te ontwerpen voor mensen die hun deuren willen beveiligen met een code. Mijn keypad verschilt van de normale keypads doordat de cijfers elke keer van plaats veranderen nadat er een cijfer wordt ingedrukt. Zo kan niemand de code afkijken of raden op basis van de volgorde van de toetsaanslagen.

Ik heb de volgende onderdelen gebruikt:

- 1 ESP-32 WROM-DEV
- 1 elektronische slot
- 1 RGB LED

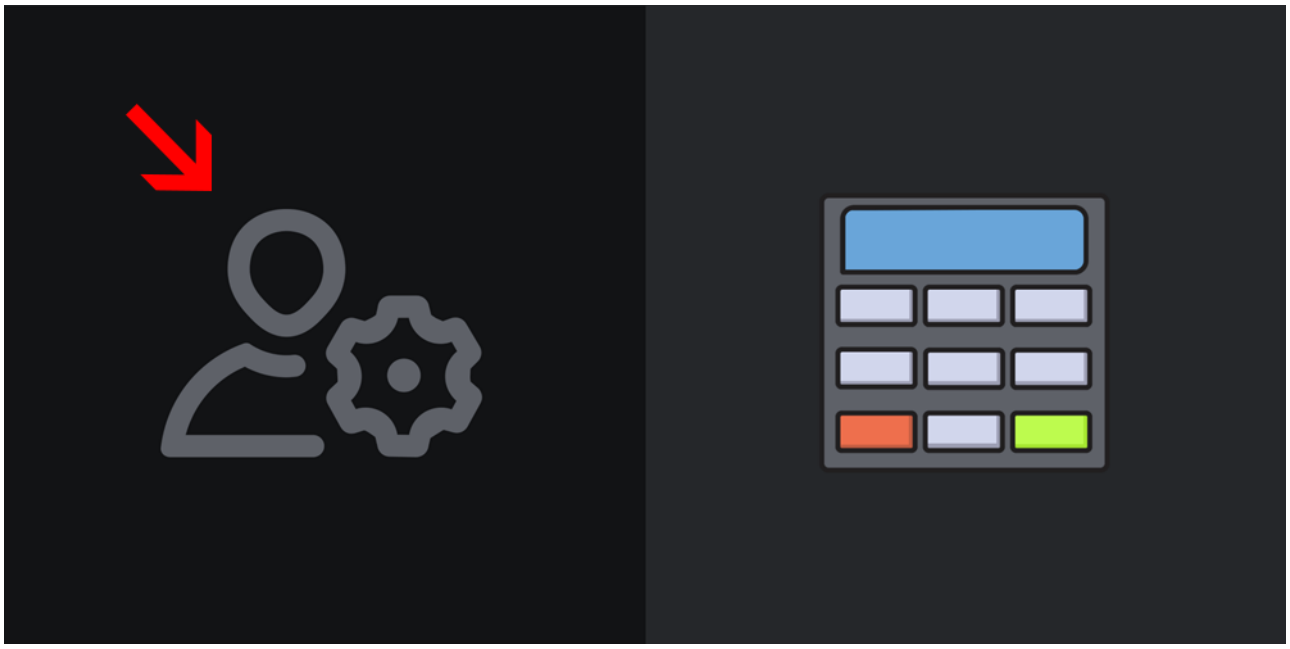
SOFTWARE

Korte uitleg

Voor de ontwikkeling van de digitale scramblepad heb ik een mix van programmeertalen gebruikt, waaronder HTML, CSS, JS, PHP en Arduino (ino). Ik koos voor deze talen omdat ik er vertrouwd mee was en omdat ik dacht dat ze goed pasten bij mijn idee voor de scramblepad. Ik heb alle code zelf geschreven, met enige hulp van de heer Van Reck en mijn medestudenten.

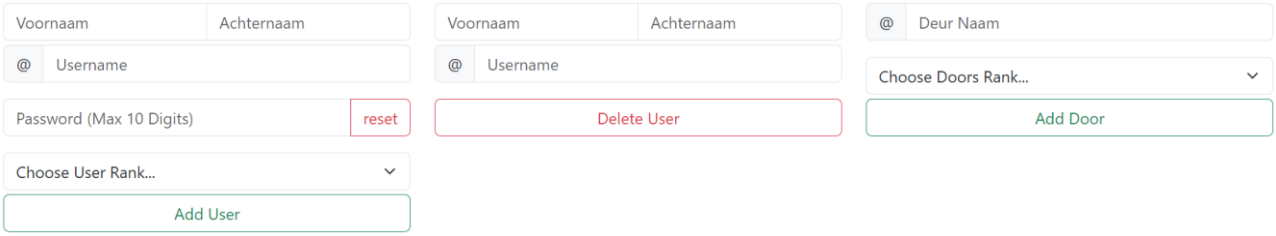
Hieronder staat een korte beschrijving van hoe iemand het systeem kan gebruiken.

Voor de initiële setup moet een beheerder op de "admin" knop klikken.



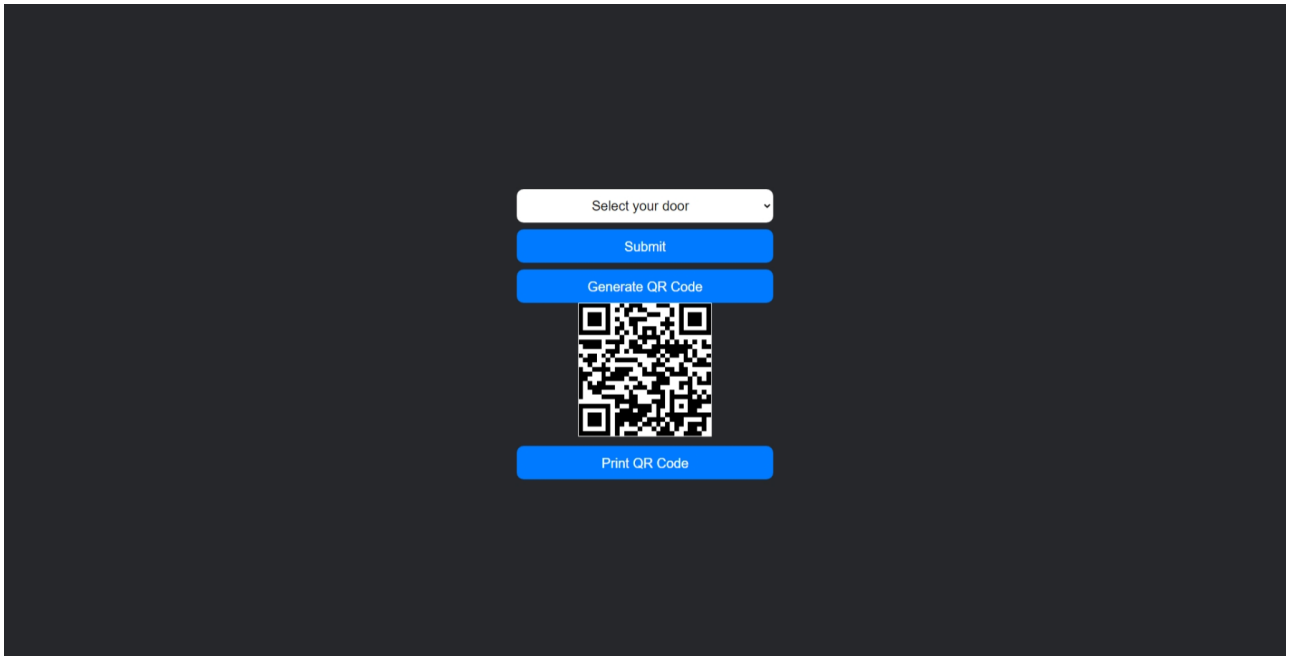
en vervolgens op "create database" om de benodigde database aan te maken.

Daarna kan de beheerder een deuren en gebruikers toevoegen aan de database.



Vervolgens kan de beheerder op "setup" klikken.

Nu hebben we de mogelijkheid om te kiezen tussen direct naar de scramblepad te gaan of een QR-code te genereren die we kunnen afdrukken en door de gebruikers kunnen laten scannen.



Nu kunnen de gebruikers zich aanmelden om de deuren te openen.

Index

Index.html

Dit HTML-bestand is de startpagina van mijn project. De pagina bevat twee knoppen, een knop links en een knop rechts.

De linkse knop geeft toegang tot de beheerderspagina, waar we nieuwe gebruikers en deuren kunnen toevoegen, oude gebruikers kunnen verwijderen en ook de database kunnen aanmaken.

De rechtse knop leidt naar de instellingenpagina, waar de beheerder de naam van de deur kan kiezen waarvoor de scramblepad wordt gebruikt.

De code bevat ook styling om de lay-out van de pagina te bepalen en de knoppen vorm te geven. Er worden hover-effecten gebruikt om de gebruiker te laten zien dat er op de knoppen geklikt kan worden. Zo zorg ik voor een mooiere en meer gebruiksvriendelijke webpagina.

HTML

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta http-equiv="X-UA-Compatible" content="IE=edge" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <link rel="stylesheet" href="index-style.css" />
    <title>Home Page</title>
  </head>
  <body>
    <div class="container">
      <button class="left">
        <a href="admin/index.html"><span></span></a>
      </button>
      <button class="right">
        <a href="setup/setup.php"><span></span></a>
      </button>
    </div>
  </body>
</html>
```

CSS

```
/* Set body margin and padding to 0 to remove default spacing */
```

```css
body {
  margin: 0;
  padding: 0;
  overflow: hidden;
  background-color: rgb(37, 39, 42);
}

/* Style the buttons */
button {
  height: 100%;
  border: none;
  outline: none;
  cursor: pointer;
  font-size: 2em;
  font-weight: bold;
  color: white;
  background-color: transparent;
  transition: all 0.3s cubic-bezier(0.25, 0.8, 0.25, 1);
}

/* Style the links inside the buttons */
a {
  display: flex;
  justify-content: center;
  align-items: center;
  height: 100%;
  width: 100%;
  padding: 5px;
  position: relative;
  color: white;
  text-decoration: none;
  font-size: 2em;
  font-weight: bold;
  background-position: center;
  background-size: cover;
}

/* Add semi-transparent overlay on hover */
a:before {
  content: "";
  position: absolute;
  top: 0;
  left: 0;
  right: 0;
  bottom: 0;
  background-color: rgba(0, 0, 0, 0.5);
  z-index: -1;
  opacity: 0;
  transition: opacity 0.3s ease-in-out;
}

a:hover:before {
```

```
  opacity: 1;
}

/* Style the text inside the links */
a span {
  position: relative;
  z-index: 1;
  color: rgb(255, 255, 255);
}

/* Style the images inside the links */
a img {
  position: relative;
  top: 0;
  left: 0;
  right: 0;
  bottom: 0;
  margin: auto;
  z-index: 0;
}

/* Center the text inside the button and allow clicking anywhere */
button a {
  display: flex;
  justify-content: center;
  align-items: center;
  height: 100%;
  width: 100%;
}

/* Center the link text inside each button */
.left a,
.right a {
  display: flex;
  justify-content: center;
}

/* Style the left button */
.left {
  width: 50%;
  float: left;
  background-image: url("img/admin3.png");
  background-position: center;
  background-repeat: no-repeat;
  background-size: 50%;
}
/* Style the right button */
.right {
  width: 50%;
  float: right;
  background-image: url("img/scramblepad.png");
  background-position: center;
```

```
  background-repeat: no-repeat;
  background-size: 50%;
}

/* Position the button container */
.container {
  position: absolute;
  top: 0;
  bottom: 0;
  width: 100%;
}
```

Setup

Setup.php

Dit is de code voor de instellingenpagina. Ik maak gebruik van een combinatie van HTML, CSS en PHP op deze pagina. Met PHP halen we alle deuren op die in de database bestaan en controleren we of ze al in gebruik zijn of niet. Als een deur al in gebruik is, wordt de optie in het dropdown-menu rood gemarkeerd. Nadat de gebruiker zijn deur heeft gekozen, heeft hij de mogelijkheid om direct naar de scramblepad te gaan of om een QR-code te genereren. Deze QR-code kunnen alle gebruikers die verbonden zijn met het netwerk scannen met hun mobiele telefoon om vervolgens naar de scramblepad te gaan en hun codes in te voeren. Voor het genereren van de QR-code maak ik gebruik van JavaScript. We halen het IP-adres van de hostmachine op, samen met de naam van de deur, en vervolgens gebruiken we de API van goqr.me om de QR-code te genereren.

HTML en PHP

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta http-equiv="X-UA-Compatible" content="IE=edge" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <title>Setup Page</title>
   <script src="setup-script.js" defer></script>
    <link rel="stylesheet" href="setup_style.css">
  </head>
  <body>
    <form action="set_qrCookies.php">
      <?php

        $link = mysqli_connect("localhost", "root", "", "locks");
```

```
        if ($link->connect_error) {
            die("Connection failed: " . $link->connect_error);
        }

        $sql = "SELECT * FROM `doors`";
        $result = $link->query($sql);
        $options = array();

        if ($result->num_rows > 0) {
            while ($row = $result->fetch_assoc()) {
                $option_value = $row["deur naam"];
                $option_class = ($row["chipid"] !== "") ? "in-use" : "";
                $options[] = array("value" => $option_value, "class" =>
$option_class);
            }
        }

        echo '<select id="select" name="deurnaam" aria-label="Select your door"
required>';
        echo '<option disabled selected value="">Select your door</option>';
        foreach ($options as $option) {
          ?>
          <option
            value="<?php echo $option["value"] ?>"
            <?php echo $option["class"] ? 'class="' . $option["class"] . '"' : ''
?>
          >
            <?php echo $option["value"] ?>
          </option>
          <?php
        }
        echo '</select>';

        $link->close();
      ?>
      <button type="submit" id="submit">Submit</button>

      <button type="button" id="generateQRCode">Generate QR Code</button>
      <div id="qr-container">
      <div id="container"></div>
      </div>
      <button type="button" id="print-btn" onclick="window.print();">Print QR
Code</button>

    </form>
  </body>
</html>
```

JavaScript

```
const generateQRCodeButton = document.getElementById("generateQRCode");
const printBtn = document.getElementById("print-btn");
```

```
const select = document.getElementById("select");

generateQRCode.addEventListener("click", function () {
  var selectedOption = select.options[select.selectedIndex];
  var deurnaamValue = selectedOption.value;

  console.log("data-value1: " + deurnaamValue);

  let ipAddress = location.hostname;
  console.log("ip = ");
  console.log(ipAddress);

  let qrURL = `https://api.qrserver.com/v1/create-qr-
code/?size=200x200&data=${ipAddress}%2FGIP%2FGIP%2Flogin%2Fset_qrCookies.php%3Fdeur
naam%3D${deurnaamValue}`;
  let qrImage = document.createElement("img");
  qrImage.src = qrURL;

  let container = document.getElementById("container");
  container.appendChild(qrImage);

  printBtn.style.display = "block";
});
```

CSS

```
body {
  background-color: rgb(37, 39, 42);
  display: flex;
  justify-content: center;
  align-items: center;
  height: 100vh;
  margin: 0;
}

form {
  display: flex;
  flex-direction: column;
  justify-content: center;
  align-items: center;
}

select,
button {
  margin-top: 10px;
  width: 20vw;
  height: 50px;
  font-size: 20px;
  border-radius: 10px;
```

```
  border: none;
  text-align: center;
}

button {
  background-color: #007bff;
  color: white;
  cursor: pointer;
}

button:hover {
  background-color: #0062cc;
}

button:focus {
  outline: none;
}

button:active {
  transform: scale(0.98);
}

/* The following styles apply to the button within the form element only */
form button {
  position: relative;
  overflow: hidden;
  text-align: center;
  line-height: 50px;
}

.in-use {
  background-color: rgba(255, 0, 0, 0.788);
  color: white;
}

#print-btn {
  display: none;
}
```

---

set_qrCookies.php

Nadat de gebruiker heeft gekozen tussen het genereren van de QR-code of direct naar de scramblepad gaan vanaf de instellingenpagina, komt hij gedurende enkele milliseconden op deze pagina terecht. Hier halen we de naam van de deur uit de URL en maken we een cookie aan met de naam van de deur, zodat we deze later kunnen gebruiken. Omdat de gebruiker deze pagina niet ziet, heb ik geen styling hiervoor toegepast.

PHP

```php
<?php
$deurnaam = $_GET['deurnaam'];
setcookie('deurnaam', $deurnaam, time() + (10 * 365 * 24 * 60 * 60), '/');
header("Location: ..\login\scramblepad.php");
?>
```

Login

Scramblepad.php

Nu komen we bij de inlogpagina met de scramblepad. Deze pagina bestaat uit HTML, CSS en JavaScript. Om de scramblepad te maken, heb ik gebruikgemaakt van een invoerveld waarin de gebruikersnaam kan worden ingevoerd en drie rijen met drie knoppen waarop willekeurige nummers worden weergegeven met behulp van JavaScript. In de JavaScript-code hebben we een array met nummers van 0 tot 9 en vervolgens een functie om deze array te schudden. Daarna roepen we de functie `getKeys()` aan om aan elke knop een nummer toe te wijzen, behalve aan de "❌"-knop en de "✓"-knop (deze symbolen zijn emoji's). Daarnaast hebben we een eventlistener die controleert of er op een knop is geklikt. Als dat het geval is, wordt de functie `getKeys()` opnieuw aangeroepen om elke knop van nieuwe nummers te voorzien. Voor de CSS maak ik vaak gebruik van hover-effecten om de gebruiker te laten zien dat hij op de knoppen kan klikken.

HTML en JS

```html
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta http-equiv="X-UA-Compatible" content="IE=edge" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <link rel="stylesheet" href="scramblepad-style.css" />
    <link
      href="https://fonts.googleapis.com/css2?family=Bebas+Neue&display=swap"
      rel="stylesheet"
    />
    <title>test page for scramblepad</title>
  </head>
  <body>
    <div class="parent">
      <form action="verify.php" method="post">
```

```
        <input type="hidden" name="password" id="password" value="" required />

        <div class="username-text-field">
          <input
            class="username-input border"
            name="username"
            id="username"
            placeholder="Username"
            aria-label="Username"
            aria-describedby="Username"
            type="text"
            required
          />
        </div>

        <div class="button-row">
          <button class="button border" type="button"></button>
          <button class="button border" type="button"></button>
          <button class="button border" type="button"></button>
        </div>

        <div class="button-row">
          <button class="button border" type="button"></button>
          <button class="button border" type="button"></button>
          <button class="button border" type="button"></button>
        </div>

        <div class="button-row">
          <button class="button border" type="button"></button>
          <button class="button border" type="button"></button>
          <button class="button border" type="button"></button>
        </div>

        <div class="button-row">
          <button class="button border no-invert ✕" type="button">✕</button>
          <button class="button border" type="button"></button>
          <button class="button border no-invert ✓" type="submit">✓</button>
        </div>
      </form>
    </div>
  </body>
</html>

<script>
  function shuffleArray(arr) {
    arr.sort(() => Math.random() - 0.5);
  }

  let arr = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9];
  shuffleArray(arr);

  getKeys();
```

```
  function getKeys() {
    let key = 0;
    document.querySelectorAll('button[type="button"]').forEach((element) => {
      if (element.textContent == "❌") return;
      element.textContent = arr[key];
      key++;
    });
  }

  document.querySelector("form").addEventListener("click", (event) => {
    if (event.target.textContent == "✓") return;
    if (event.target.textContent == "❌") {
      document.getElementById("password").value = "";
      return;
    }
    document.getElementById("password").value += event.target.textContent;
    shuffleArray(arr);
    getKeys();
  });
</script>
```

CSS

```
* {
  box-sizing: border-box;
}

body {
  overflow: hidden;
  font-family: "Bebas Neue", cursive;
  font-size: large;
  background-color: rgb(37, 39, 42);
}

.parent {
  display: grid;
  place-content: center;
  height: 100vh;
}

.border {
  border-radius: 20px;
  border-width: 2px;
  border-style: solid;
  border-color: black;
  font-size: 1.5em;
}

.username-text-field {
  padding: 10px;
}
```

```
.username-input {
  width: 60.5vw;
  height: 6vh;
}

.button-row {
  padding: 10px;
}

.button {
  position: relative;
  overflow: hidden;
  width: 20vw;
  height: 10vh;
  text-align: center;
  line-height: 50px;
}

.button::before {
  content: "";
  position: absolute;
  top: 0;
  left: 0;
  width: 100%;
  height: 100%;
  background-color: transparent;
  filter: invert(1);
}

button:not(.no-invert):hover::before {
  background-color: rgb(37, 39, 42);
}

.❌ {
  background-color: tomato;
}

.❌:hover {
  background-color: red;
}

.✓ {
  background-color: greenyellow;
}

.✓:hover {
  background-color: green;
}
```

Uitleg

Dit is de webpagina waarop de gebruiker zijn code kan invoeren. De knoppen bevatten cijfers van 0 tot 9 die bij elke invoer in een willekeurige volgorde worden geschud.

De code van deze webpagina bestaat uit HTML, CSS en JavaScript. De HTML-code definieert de structuur van de webpagina en de CSS-code geeft de webpagina stijl en opmaak. De JavaScript-code zorgt ervoor dat de knoppen worden geschud.

De JavaScript-code werkt als volgt: wanneer de pagina wordt geladen, wordt er een array gemaakt met de cijfers van 0 tot 9. Vervolgens worden deze cijfers in een willekeurige volgorde geschud met behulp van een functie genaamd shuffleArray(). Deze functie gebruikt de sort() methode van de array met een willekeurige volgorde om de cijfers te schudden.

Vervolgens wordt een functie genaamd getKeys() aangeroepen die de geschudde cijfers toewijst aan de knoppen op de webpagina. Deze functie gebruikt de forEach() methode om door alle knoppen te lopen en de cijfers toe te wijzen op basis van de volgorde van de geschudde array.

Wanneer een gebruiker op een knop klikt, wordt er een event listener geactiveerd die controleert welke knop er is ingedrukt en het bijbehorende cijfer toevoegt aan een inputveld dat is verborgen op de webpagina. De shuffleArray() en getKeys() functies worden vervolgens opnieuw aangeroepen om de knoppen opnieuw te schudden en de nieuwe geschudde cijfers toe te wijzen aan de knoppen.

De CSS-code is verantwoordelijk voor de stijl en opmaak van de webpagina. Het definieert de kleuren, de grootte en de vorm van de knoppen en het inputveld. Het zorgt ook voor animaties en visuele feedback wanneer een gebruiker op een knop klikt.

Verify.php

Na het invoeren van de gebruikersnaam en het wachtwoord wordt alles doorgestuurd naar de pagina "verificatie.php". Deze pagina bestaat alleen uit PHP, omdat de gebruiker er nooit direct toegang toe heeft. We maken opnieuw verbinding met de database en controleren of de server de juiste aanvraag stuurt. We halen de gebruikersnaam, het wachtwoord en de cookie met de naam van de deur op. Vervolgens maak ik gebruik van `mysqli_real_escape_string` om de gebruikersnaam en het wachtwoord te saneren en zo

SQL-injecties te voorkomen. (Hier zal ik even uitleggen hoe `mysqli_real_escape_string` werkt: het is een functie in PHP die speciale tekens in een string onschadelijk maakt voor gebruik in SQL-query's.)

Nu vragen we de database om het wachtwoord op te halen dat overeenkomt met de ingevoerde gebruikersnaam. Het wachtwoord in de database is gehasht, dus we gebruiken `password_verify` om te controleren of de gebruiker het juiste wachtwoord heeft ingevoerd. Vervolgens vragen we de database om de rang van de gebruiker en de rang van de deur. Als de rang van de gebruiker gelijk is aan of groter is dan die van de deur, mag hij de deur openen, anders niet.

Om de deur te openen, updaten we de status van de deur in de database van "closed" naar "open" voor 5 seconden en vervolgens weer terug naar "closed".

PHP

```php
<?php
$link = mysqli_connect("localhost", "root", "", "locks");

if ($_SERVER["REQUEST_METHOD"] == "POST") {
    $username = $_POST['username'];
    $username = strtolower($username);
    $password = $_POST['password'];
    $deurnaam_cookie = $_COOKIE['deurnaam'];
    $deurnaam_cookie = strtolower($deurnaam_cookie);
    $sanitized_username = mysqli_real_escape_string($link, $username);
    $sanitized_password = mysqli_real_escape_string($link, $password);

    $sql = "SELECT `password` FROM `users` WHERE `username`
='$sanitized_username'";
    $result = mysqli_query($link, $sql) or die(mysqli_error($link));
    $hasedPwd = mysqli_fetch_array($result);

    $deHashedPwd = password_verify($sanitized_password,$hasedPwd['password']);

    $sql = "SELECT `rank` FROM `users` WHERE `username`='$sanitized_username' AND
`password`='$deHashedPwd'";
    $result = mysqli_query($link, $sql) or die(mysqli_error($link));
    $user_rank = mysqli_fetch_array($result);

    $sql = "SELECT `rank` FROM `doors` WHERE `deur naam` = '$deurnaam_cookie'";
    $result = mysqli_query($link, $sql) or die(mysqli_error($link));
    $deurrank = mysqli_fetch_array($result);

    if($user_rank['rank'] >= $deurrank[1]) {
        $sql = "UPDATE `doors` SET `status`='open' WHERE `deur
naam`='$deurnaam_cookie'";
```

```
        $result = mysqli_query($link, $sql) or die(mysqli_error($link));
        sleep(5); // wait for 5 seconds before updating the status to close
        $sql = "UPDATE `doors` SET `status`='close' WHERE `deur
naam`='$deurnaam_cookie'";
        $result = mysqli_query($link, $sql) or die(mysqli_error($link));
        sleep(3);
        header("Location: scramblepad.php");
        exit();
    } else {
        header("Location: scramblepad.php");
        exit();
    }

    mysqli_close($link);
}
?>
```

Doorstatus.php

Op deze pagina is waar de Arduino altijd naartoe surft. De gebruiker komt hier nooit terecht, dus er is geen styling aanwezig. Op deze pagina bestaat er slechts één woord, namelijk de status van de deur: "closed" of "open". Om te weten welke Arduino welke deur moet openen, maken we gebruik van zijn chip-ID.

Een chip-ID is een unieke identificatiecode die aan elke Arduino is toegewezen. Het is een manier om elke Arduino onderscheidend te maken, zodat we ze kunnen herkennen en ermee kunnen communiceren. We halen het chip-ID uit de URL en controleren in de database of het al bestaat. Als het al in de database aanwezig is, betekent dit dat het al gekoppeld is aan een deur. We vragen dan aan de database de status van die specifieke deur op en tonen deze.

Als het chip-ID nog niet is gekoppeld aan een deur, betekent dit dat de Arduino nog niet verbonden is met een specifieke deur. In dat geval moeten we een deur aan de Arduino toewijzen. We selecteren dan de eerste beschikbare deur en koppelen deze aan de Arduino.

```
<?php
$link = mysqli_connect("localhost", "root", "", "locks");

$chipid = mysqli_real_escape_string($link, $_GET["naam"]);
if (!empty($chipid)) {

    $sql = "SELECT COUNT(*) FROM doors WHERE `chipid` = '$chipid'";
    $result = mysqli_query($link, $sql) or die(mysqli_error($link));
    $count = mysqli_fetch_array($result)[0];
```

```
    if ($count > 0) {

        $sql = "SELECT `status` FROM doors WHERE `chipid` = '$chipid'";
        $result = mysqli_query($link, $sql) or die(mysqli_error($link));
        $status = mysqli_fetch_array($result);

        echo $status[0];

    } else {

        $sql = "UPDATE `doors` SET `chipid`='$chipid' WHERE `chipid` = '' LIMIT 1";
        mysqli_query($link, $sql) or die(mysqli_error($link));

    }
} else {
    header('Location: scramblepad.html');
}
?>
```

Administrator

Index.html

Nu komen we bij de beheerderspagina's en we beginnen met de index (de startpagina). Hier zijn er 3 formulieren met verschillende invoervelden. Het eerste formulier is bedoeld om nieuwe gebruikers aan te maken, het tweede formulier is om oude gebruikers te verwijderen en het derde formulier is om nieuwe deuren toe te voegen.

Om nieuwe gebruikers aan te maken, vragen we om de voornaam, achternaam, gebruikersnaam en een wachtwoord dat maximaal 10 tekens lang kan zijn (normaal gesproken zouden we de lengte van een wachtwoord niet beperken, maar voor dit project dacht ik dat te veel tekens niet gebruiksvriendelijk zou zijn). We vragen ook om de rang die de gebruiker moet krijgen.

Om een oude gebruiker te verwijderen, vragen we alleen om de voornaam, achternaam en gebruikersnaam.

Om een nieuwe deur toe te voegen, vragen we om de naam van de deur en de minimale rang die een gebruiker moet hebben om deze te openen.

Op deze pagina is er ook een navigatiebalk met links naar de "home page" (leidt naar de startpagina), "setup" (leidt naar de setup) en een van de belangrijkste links, "Create DataBase", om de database aan te maken.

Voor de styling heb ik zelf geen CSS geschreven, maar heb ik gebruikgemaakt van Bootstrap. Bootstrap is een framework dat voorgebouwde CSS-styling en JavaScript-componenten biedt. Waarom heb ik Bootstrap gebruikt in plaats van mijn eigen CSS? Simpelweg omdat ik wilde kijken welke ik leuker vond om te gebruiken, welke er beter uitzag en welke het gemakkelijkst was. Uiteindelijk heb ik besloten dat ik mijn eigen CSS leuker en gemakkelijker vond om te gebruiken. Bootstrap is echter niet slecht; het is handig als je iets kleins maakt en/of als je niet creatief genoeg voelt om alles vanaf nul te maken.

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta http-equiv="X-UA-Compatible" content="IE=edge" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <!-- Bootstrap CSS -->
    <link
      href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-
alpha1/dist/css/bootstrap.min.css"
      rel="stylesheet"
      integrity="sha384-
GLhlTQ8iRABdZLl6O3oVMWSktQOp6b7In1Zl3/Jr59b6EGGoI1aFkw7cmDA6j6gD"
      crossorigin="anonymous"
    />
    <title>Homepage</title>
  </head>

  <body>
    <nav class="navbar navbar-expand-lg navbar-light bg-secondary">
      <a class="navbar-brand" href="../index.html">Home Page</a>
      <button
        class="navbar-toggler"
        type="button"
        data-toggle="collapse"
        data-target="#navbarNav"
        aria-controls="navbarNav"
        aria-expanded="false"
        aria-label="Toggle navigation"
      >
        <span class="navbar-toggler-icon"></span>
      </button>
      <div class="collapse navbar-collapse" id="navbarNav">
        <ul class="navbar-nav">
          <li class="nav-item">
            <a class="nav-link" href="../setup/setup.php">Setup</a>
          </li>
          <li class="nav-item">
```

```
          <a class="nav-link" href="createDb.php">Create DataBase</a>
        </li>
      </ul>
    </div>
  </nav>

  <div class="containers">
    <div class="row row-cols-md-3 p-2 mb-3 text-center">
      <form action="addUsers.php" method="post">
        <!-- USER Voornaam AND Achternaam -->
        <div class="col mb-1 themed-grid-col">
          <div class="input-group">
            <input
              required
              type="text"
              name="voornaam"
              placeholder="Voornaam"
              aria-label="Voornaam"
              class="form-control"
            />
            <input
              required
              type="text"
              name="achternaam"
              placeholder="Achternaam"
              aria-label="Achternaam"
              class="form-control"
            />
          </div>
        </div>
        <!-- USER Voornaam AND Achternaam-->

        <!-- USERNAME TEXT FIELD -->
        <div class="col mb-1 themed-grid-col">
          <div class="input-group mb-3">
            <span class="input-group-text" id="basic-addon1">@</span>
            <input
              required
              type="text"
              class="form-control"
              name="username"
              placeholder="Username"
              aria-label="Username"
              aria-describedby="basic-addon1"
            />
          </div>
        </div>
        <!-- USERNAME TEXT FIELD -->

        <!-- USER PASSWORD TEXT FIELD -->
        <div class="col mb-1 themed-grid-col">
          <div class="input-group mb-3">
```

```
            <input
              required
              type="number"
              class="form-control"
              name="password"
              id="password"
              maxlength="10"
              placeholder="Password (Max 10 Digits)"
              aria-label="Password"
              aria-describedby="button-addon2"
            />
            <button
              class="btn btn-outline-danger"
              type="button"
              id="clear-password-btn"
            >
              reset
            </button>
          </div>
        </div>
        <!-- USER PASSWORD TEXT FIELD -->
        <!-- USER RANK-->
        <div class="col mb-1 themed-grid-col">
          <div class="input-group">
            <select
              class="form-select"
              name="rank"
              id="input requiredGroupSelect04"
              aria-label="Example select with button addon"
            >
              <option selected>Choose User Rank...</option>
              <option value="1">1</option>
              <option value="2">2</option>
              <option value="3">3</option>
              <option value="4">4</option>
            </select>
          </div>
        </div>
        <!-- USER RANK-->

        <!-- SUBMITE BUTTON-->
        <div class="col mb-1 themed-grid-col">
          <div class="d-grid gap-2">
            <button class="btn btn-outline-success" type="submit">
              Add User
            </button>
          </div>
        </div>
        <!-- SUBMITE BUTTON-->
      </form>

      <form action="removeUsers.php" method="post">
```

```
        <!-- USER Voornaam en Achternaam -->
        <div class="col mb-1 themed-grid-col">
          <div class="input-group">
            <input
              required
              type="text"
              name="voornaam"
              aria-label="Voornaam"
              placeholder="Voornaam"
              class="form-control"
            />
            <input
              required
              type="text"
              name="achternaam"
              aria-label="Achternaam"
              placeholder="Achternaam"
              class="form-control"
            />
          </div>
        </div>
        <!-- USER Voornaam en Achternaam -->

        <!-- USERNAME TEXT FIELD -->
        <div class="col mb-1 themed-grid-col">
          <div class="input-group mb-3">
            <span class="input-group-text" id="basic-addon1">@</span>
            <input
              required
              type="text"
              class="form-control"
              name="username"
              placeholder="Username"
              aria-label="Username"
              aria-describedby="basic-addon1"
            />
          </div>
        </div>
        <!-- USERNAME TEXT FIELD -->

        <!-- SUBMITE BUTTON-->
        <div class="col mb-1 themed-grid-col">
          <div class="d-grid gap-2">
            <button class="btn btn-outline-danger" type="submit">
              Delete User
            </button>
          </div>
        </div>
        <!-- SUBMITE BUTTON-->
      </form>

      <form action="addDoor.php" method="post">
```

```
        <!-- deur naam TEXT FIELD -->
        <div class="col mb-1 themed-grid-col">
          <div class="input-group mb-3">
            <span class="input-group-text" id="basic-addon1">@</span>
            <input
              required
              type="text"
              class="form-control"
              id="deurnaam"
              name="deurnaam"
              placeholder="Deur Naam"
              aria-label="deur Naam"
              aria-describedby="basic-addon1"
            />
          </div>
        </div>
        <!-- deur nummer TEXT FIELD -->

        <!-- DOOR RANK-->
        <div class="col mb-1 themed-grid-col">
          <div class="input-group">
            <select
              class="form-select"
              id="input requiredGroupSelect04"
              name="rank"
              aria-label="Example select with button addon"
            >
              <option selected>Choose Doors Rank...</option>
              <option value="1">1</option>
              <option value="2">2</option>
              <option value="3">3</option>
              <option value="4">4</option>
            </select>
          </div>
        </div>
        <!-- DOOR RANK-->

        <!-- SUBMITE BUTTON-->
        <div class="col mb-1 themed-grid-col">
          <div class="d-grid gap-2">
            <button class="btn btn-outline-success" type="submit">
              Add Door
            </button>
          </div>
        </div>
        <!-- SUBMITE BUTTON-->
      </form>
    </div>
  </div>

  <script
    src="https://code.jquery.com/jquery-3.2.1.slim.min.js"
```

```
      integrity="sha384-
KJ3o2DKtIkvYIK3UENzmM7KCkRr/rE9/Qpg6aAZGJwFDMVNA/GpGFF93hXpG5KkN"
      crossorigin="anonymous"
    ></script>
    <script
      src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.12.9/umd/popper.min.j
s"
      integrity="sha384-
ApNbgh9B+Y1QKtv3Rn7W3mgPxhU9K/ScQsAP7hUibX39j7fakFPskvXusvfa0b4Q"
      crossorigin="anonymous"
    ></script>
    <script
      src="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/js/bootstrap.min.js"
      integrity="sha384-
JZR6Spejh4U02d8jOt6vLEHfe/JQGiRRSQQxSfFWpi1MquVdAyjUar5+76PVCmYl"
      crossorigin="anonymous"
    ></script>
  </body>
</html>
<script>
  const clearPasswordBtn = document.getElementById("clear-password-btn");
  const passwordField = document.getElementById("password");

  clearPasswordBtn.addEventListener("click", () => {
    passwordField.value = "";
  });
</script>
```

AddUsers.php

Dit bestand is bedoeld om gebruikers toe te voegen. Net als op de verificatiepagina ontvangen we gegevens van de gebruiker en saniteren we alles met `mysqli_real_escape_string`. Vervolgens hashen we het wachtwoord met behulp van `password_hash` en de optie `PASSWORD_DEFAULT`.

`password_hash` is een functie in PHP die wordt gebruikt om wachtwoorden te hashen voordat ze worden opgeslagen in de database. Het neemt het wachtwoord als invoer en genereert een gehashte versie van het wachtwoord met behulp van een veilige hashing-algoritme. De optie `PASSWORD_DEFAULT` geeft aan dat het standaard hashing-algoritme van PHP moet worden gebruikt (momenteel bcrypt). Dit zorgt ervoor dat het wachtwoord veilig wordt gehasht met een sterk algoritme.

Nadat het wachtwoord is gehasht, voegen we alle gegevens toe aan de database voor de nieuwe gebruiker.

```
<?php
$link = mysqli_connect("localhost","root","","locks");
```

```php
if ($_SERVER["REQUEST_METHOD"] == "POST") {
    $voornaam = $_POST['voornaam'];
    $voornaam =strtolower($voornaam);

    $achternaam =$_POST['achternaam'];
    $achternaam =strtolower($achternaam);

    $username = $_POST['username'];
    $username = strtolower($username);

    $password = $_POST['password'];
    $rank = $_POST['rank'];
    $sanitized_voornaam = mysqli_real_escape_string($link, $voornaam);
    $sanitized_achternaam = mysqli_real_escape_string($link, $achternaam);
    $sanitized_username = mysqli_real_escape_string($link, $username);
    $sanitized_password = mysqli_real_escape_string($link, $password);
    $sanitized_rank = mysqli_real_escape_string($link, $rank);

    $hasedPwd = password_hash($sanitized_password, PASSWORD_DEFAULT);

    $sql = "INSERT INTO `users`(`voornaam`, `achternaam`, `username`, `password`,
`rank`)
            VALUES
('$sanitized_voornaam','$sanitized_achternaam','$sanitized_username','$hasedPwd','$
sanitized_rank')";

    $result = mysqli_query($link, $sql)
    or die(mysqli_error($link));
}
?>

<script>
    window.addEventListener("load", () => {
        let time = 1
        const interval = setInterval(() => {
        time--
            if(time == 0) {
                window.location.replace("index.html")
                clearInterval(interval)
            }
        },1000)
    })
</script>
```

RemoveUsers.php

Dit bestand is bedoeld om gebruikers te verwijderen. Het werkt op een vergelijkbare manier als het bestand om gebruikers toe te voegen, maar in plaats van een nieuwe gebruiker aan de database toe te voegen, verwijderen we de gebruiker.

We ontvangen de gegevens van de gebruiker, zoals de voornaam, achternaam en gebruikersnaam. Vervolgens saneren we deze gegevens met behulp van `mysqli_real_escape_string`. Daarna kunnen we de gebruiker identificeren in de database en de juiste gebruiker verwijderen.

```
<?php
$link = mysqli_connect("localhost","root","","locks");

if ($_SERVER["REQUEST_METHOD"] == "POST") {
    $voornaam = $_POST['voornaam'];
    $achternaam = $_POST['achternaam'];
    $username = $_POST['username'];

    $sanitized_voornaam = mysqli_real_escape_string($link, $voornaam);
    $sanitized_achternaam = mysqli_real_escape_string($link, $achternaam);
    $sanitized_username = mysqli_real_escape_string($link, $username);


    $sql = "DELETE FROM `users` WHERE `voornaam`='$sanitized_voornaam' and
`achternaam`= '$sanitized_achternaam' and `username`='$sanitized_username'";

    $result = mysqli_query($link, $sql)
    or die(mysqli_error($link));
}
?>

<script>
    window.addEventListener("load", () => {
        let time = 1
        const interval = setInterval(() => {
        time--
            if(time == 0) {
                window.location.replace("index.html")
                clearInterval(interval)
            }
        },1000)
    })
</script>
```

AddDoors.php

Dit bestand is bedoeld om deuren toe te voegen. Het werkt op een vergelijkbare manier als het bestand om gebruikers toe te voegen, maar in plaats van een nieuwe gebruiker aan de gebruikerstabel toe te voegen, voegen we een nieuwe deur toe aan de deurentabel.

Net als bij het toevoegen van gebruikers, ontvangen we de gegevens van de deur, zoals de naam van de deur en de minimale rang die een gebruiker moet hebben om deze te openen. We saneren deze gegevens met behulp van `mysqli_real_escape_string` om te voorkomen dat er ongeldige gegevens in de database worden opgeslagen.

```
<?php
$link = mysqli_connect("localhost","root","","locks");

if ($_SERVER["REQUEST_METHOD"] == "POST") {
    $deurnaam = $_POST['deurnaam'];
    $deurnaam =strtolower($deurnaam);
    $sanitized_deurnaam = mysqli_real_escape_string($link, $deurnaam);

    $rank = $_POST['rank'];

    $sql = "INSERT INTO `doors`(`deur naam`, `status`,`rank`,`chipid`) VALUES
('$sanitized_deurnaam','close',$rank,'')";

    $result = mysqli_query($link, $sql)
    or die(mysqli_error($link));
}
?>

<script>
    window.addEventListener("load", () => {
        let time = 1
        const interval = setInterval(() => {
        time--
        if(time == 0) {
                window.location.replace("index.html")
                clearInterval(interval)
            }
        },1000)
    })
</script>
```

Database aanmaken

Nu gaan we een zeer belangrijk onderdeel behandelen: het aanmaken van de database. We zullen phpMyAdmin gebruiken om verbinding te maken en een SQL-bestand te openen met alle SQL-commando's, die we vervolgens zullen uitvoeren. Let op: bij het aanmaken van de database voegen we een standaardgebruiker toe met de volgende gegevens: 'admin', 'admin', 'admin', '0000', '4'. Het is belangrijk om deze gebruiker snel te verwijderen en je eigen admin-gebruiker aan te maken.

CreateDB.php

```php
<?php
// Database configuration
$host = 'localhost';
$username = 'root';
$password = '';
$database = 'locks';
$sql_file = 'locks.sql';

// Create connection
$conn = mysqli_connect($host, $username, $password);

// Check connection
if (!$conn) {
    die("Connection failed: " . mysqli_connect_error());
}

// Create database if it doesn't exist
$sql = "CREATE DATABASE IF NOT EXISTS $database";
if (mysqli_query($conn, $sql)) {
    echo "Database created successfully!<br>";
} else {
    echo "Error creating database: " . mysqli_error($conn) . "<br>";
}

// Select the database
mysqli_select_db($conn, $database);

// Read SQL file
$sql = file_get_contents($sql_file);

// Execute multi-query SQL commands
if (mysqli_multi_query($conn, $sql)) {
    echo "SQL script executed successfully!";
} else {
    echo "Error executing SQL script: " . mysqli_error($conn);
}

// Close connection
```

```
mysqli_close($conn);
?>

<script>
    window.addEventListener("load", () => {
                let time = 2
                const interval = setInterval(() => {
                    time--
                    if(time == 0) {
                        window.location.replace("index.html")
                        clearInterval(interval)
                    }
                },1000)
            })
</script>
```

Locks.sql

```
-- phpMyAdmin SQL Dump
-- version 5.2.0
-- https://www.phpmyadmin.net/
--
-- Host: 127.0.0.1:3306
-- Generation Time: Mar 10, 2023 at 03:26 PM
-- Server version: 8.0.31
-- PHP Version: 8.0.26

SET SQL_MODE = "NO_AUTO_VALUE_ON_ZERO";
START TRANSACTION;
SET time_zone = "+00:00";


/*!40101 SET @OLD_CHARACTER_SET_CLIENT=@@CHARACTER_SET_CLIENT */;
/*!40101 SET @OLD_CHARACTER_SET_RESULTS=@@CHARACTER_SET_RESULTS */;
/*!40101 SET @OLD_COLLATION_CONNECTION=@@COLLATION_CONNECTION */;
/*!40101 SET NAMES utf8mb4 */;

--
-- Database: `locks`
--

-- --------------------------------------------------------

--
-- Table structure for table `doors`
--

DROP TABLE IF EXISTS `doors`;
CREATE TABLE IF NOT EXISTS `doors` (
  `deur naam` varchar(30) NOT NULL,
```

```
  `status` varchar(7) NOT NULL,
  `rank` varchar(10) NOT NULL,
  `chipid` varchar(17) NOT NULL
);


-- --------------------------------------------------------

--
-- Table structure for table `users`
--

DROP TABLE IF EXISTS `users`;
CREATE TABLE IF NOT EXISTS `users` (
  `voornaam` varchar(30) NOT NULL,
  `achternaam` varchar(30) NOT NULL,
  `username` varchar(30) NOT NULL,
  `password` longtext NOT NULL,
  `rank` varchar(30) NOT NULL
);

--
-- Dumping data for table `users`
--

INSERT INTO `users` (`voornaam`, `achternaam`, `username`, `password`, `rank`)
VALUES
('admin', 'admin', 'admin', 0000, '4');
COMMIT;

/*!40101 SET CHARACTER_SET_CLIENT=@OLD_CHARACTER_SET_CLIENT */;
/*!40101 SET CHARACTER_SET_RESULTS=@OLD_CHARACTER_SET_RESULTS */;
/*!40101 SET COLLATION_CONNECTION=@OLD_COLLATION_CONNECTION */;
```

Database

Laten we eens kijken hoe jouw database eruitziet. We hebben een database genaamd 'locks', waarin twee tabellen zijn opgenomen: de 'users'-tabel en de 'doors'-tabel.

De 'users'-tabel bevat 5 kolommen: voornaam, achternaam, gebruikersnaam, wachtwoord en rang.

voornaam	achternaam	username	password	rank
john	doe	johndoe	$2y$10$aCQQbWrhM8Zyg2NiCoXM1u4F1OCBJJyE82gElMekMNC...	1
jane	smith	janesmith	$2y$10$lEHrYi8vAeDpG6w4B.rlP.ps7NzZyFmcCMR1mXfvQsa...	2
mark	johnson	markjohnson	$2y$10$9L4lNZOqlKcGVdthr.UlQu1402q9ajAXttLKpAVhjEi...	3
sarah	williams	sarahw	$2y$10$0Dyj8TSEiaIbJaVN9hsohODsOhaEBeGq.o6niBcas/c...	4

De 'doors'-tabel bevat 4 kolommen: deurnaam, status, rang en chip-ID (de chip-ID van de Arduino).

deur naam	status	rank	chipid
Voordeur	Open	1	123456
Achterdeur	Geslote	2	789012
Kantoordeur	Open	3	345678
Garagedeur	Geslote	4	901234

Esp

Hier is de code voor de ESP, waarbij we twee bibliotheken gebruiken: 'wifi.h' en 'HTTPClient.h'. Deze bibliotheken hebben verschillende functies:

1. De 'wifi.h'-bibliotheek stelt de ESP in staat om verbinding te maken met een Wi-Fi-netwerk en toegang te krijgen tot internet.

2. De 'HTTPClient.h'-bibliotheek biedt functionaliteit om HTTP-verzoeken te maken en te verwerken, waardoor de ESP kan communiceren met webpagina's.

De ESP surft naar de 'doorStatus.php'-pagina en stuurt zijn chip-ID als identificatie. Met behulp van deze chip-ID kan de webpagina bepalen welke informatie de ESP moet ontvangen. Als er "open" wordt weergegeven, zetten we de pin van de slop op 'HIGH'. Als het "closed" is, zetten we de pin op 'LOW'. Bovendien gebruik ik een RGB-ledje om verschillende statussen aan de gebruiker te communiceren: standby (in afwachting), de deur kan worden geopend en het aangeven dat de deur wordt gesloten.

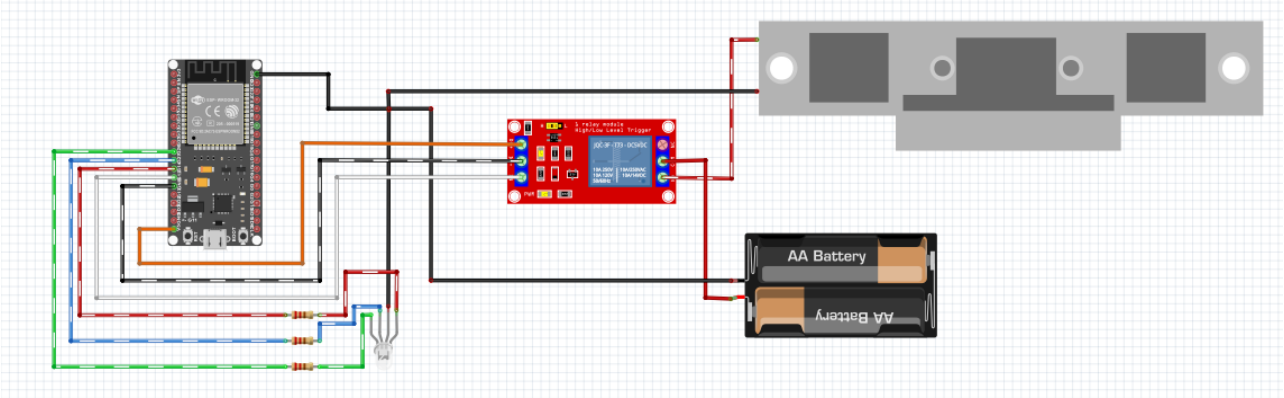
```
#include <WiFi.h>
#include <HTTPClient.h>
// #include <Preferences.h>

const char* ssid = "WIFIIICT";
const char* password = "fakatijger";
//Your Domain name with URL path or IP address with path
String serverName = "http://10.3.41.20/gip/gip/login/doorStatus.php";

// the following variables are unsigned longs because the time, measured in
// milliseconds, will quickly become a bigger number than can be stored in an int.
unsigned long lastTime = 0;
// Timer set to 10 minutes (600000)
//unsigned long timerDelay = 600000;
// Set timer to 5 seconds (5000)
unsigned long timerDelay = 1000;
const int slot = 12;
const int PIN_BLUE = 27;
```

```
const int PIN_RED = 26;
const int PIN_GREEN = 14;


// Preferences preferences;

void setup() {
  pinMode(slot,      OUTPUT);
  pinMode(PIN_RED,   OUTPUT);
  pinMode(PIN_GREEN, OUTPUT);
  pinMode(PIN_BLUE,  OUTPUT);

  Serial.begin(115200);

  WiFi.begin(ssid, password);
  Serial.println("Connecting");
  while(WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.println("");
  Serial.print("Connected to WiFi network with IP Address: ");
  Serial.println(WiFi.localIP());

  Serial.println("Timer set to 5 seconds (timerDelay variable), it will take 5
seconds before publishing the first reading.");
}

void loop() {
  digitalWrite(PIN_RED, LOW);
  digitalWrite(PIN_GREEN, LOW);
  digitalWrite(PIN_BLUE, HIGH);
  //Send an HTTP POST request every 10 minutes
  if ((millis() - lastTime) > timerDelay) {
    //Check WiFi connection status
    if(WiFi.status()== WL_CONNECTED){
      HTTPClient http;

      uint64_t chipid = ESP.getEfuseMac();
      char chipid_str[17];
      sprintf(chipid_str, "%016llX", chipid);

      String serverPath = serverName + "?naam="+chipid_str;
      Serial.println(serverPath);

      // Your Domain name with URL path or IP address with path
      http.begin(serverPath.c_str());

      // Send HTTP GET request
      int httpResponseCode = http.GET();

      if (httpResponseCode>0) {
```

```
        Serial.print("HTTP Response code: ");
        Serial.println(httpResponseCode);
        String payload = http.getString();
        Serial.println(payload);

        //check payload (door status)
        if (payload == "open" ) {
          //open door
          digitalWrite(PIN_RED, LOW);
          digitalWrite(PIN_GREEN, HIGH);
          digitalWrite(PIN_BLUE, LOW);

          digitalWrite(slot,HIGH);
        } else if (payload == "close") {
          //close door
          digitalWrite(PIN_RED, HIGH);
          digitalWrite(PIN_GREEN, LOW);
          digitalWrite(PIN_BLUE, LOW);

          digitalWrite(slot,LOW);
        }
      } else {
        Serial.print("Error code: ");
        Serial.println(httpResponseCode);
      }
      // Free resources
      http.end();
    }
    else {
      Serial.println("WiFi Disconnected");
      ESP.restart();
    }
    lastTime = millis();
  }
}
```

HARDWARE

Om de deur te openen, gebruik ik hardware zoals een ESP32 microcontroller, een relais om het slot te activeren, en een RGB-LED om de status aan te geven. De gebruiker kan de code invoeren via een tablet die bij de deur is geplaatst of een QR-code scannen met zijn of haar smartphone.



BESLUIT

Ik vond dit een erg leuk project om aan te werken. Het heeft me veel geleerd op het gebied van PHP, CSS en JavaScript. Ik heb ontdekt dat elk probleem meerdere manieren heeft om het op te lossen en dat hard werken aan iets het leuk en bevredigend kan maken. Daarnaast heb ik geleerd hoe ik GitHub kan gebruiken en ik merk dat het een handige manier is om een logboek bij te houden, toegang te hebben tot mijn oude code en om mijn code gemakkelijker met anderen te delen.

BIBLIOGRAFIE

Officiële sites

- **Bootstrap:** https://getbootstrap.com
- **W3Schools Online Web Tutorials:** https://www.w3schools.com/
- **QR code API:** https://goqr.me/api/
- **Arduino:** https://www.arduino.cc/
- **PHP: Hypertext Preprocessor:** https://www.php.net/