1. **Projektdurchführung**
   1. **Systemkonzept**

Abbildung 1: Systemkonzept

In dieser Abbildung sehen wir, dass wir drei Blöcke haben, die die allgemeine Funktionsweise unseres Projekts erklären.

* 1. **Blöcke Funktionen**

Im ersten Block (rot) haben wir es im Grunde mit unserer Webapplikation zu tun, deren Zweck es ist, die Interaktionen des Benutzers mit der Datenbank durchzuführen und Informationen an das ESP-32 zu übermitteln. Benutzer loggt sich in sein Konto ein Von dort erhält er Informationen aus der Datenbank und kann, wenn er möchte, einige seiner Werte ändern, z. B. den Status der Tür oder der Kamera (d. h. offen oder geschlossen). All dies geschieht mithilfe der Anfrage/Antwort-Prinzips. Der Benutzer, der den Status eines Elements ändern möchte, sendet eine SQL-Anfrage an den Server, der Server bearbeitet die Anfrage, sendet die Antwort an den Client zurück und aktualisiert die Datenbank.

Im zweiten Block (grün) ist der ESP-32 Mikrocontroller der Dirigent, da er die Befehle des Webservers interpretiert. Zuerst verbindet sich der Mikrocontroller mit einem WiFi, was ihm Zugang zum Internet verschafft, dann empfängt er kontinuierlich den Status der Tür, der Kamera und der Benachrichtigungen aus der Datenbank. Anschließend interpretiert er die Daten.

Im dritten Block (blau) haben wir es mit einem Benachrichtigungssystem zu tun, das uns warnt, wenn die Post schon da ist. Wie es funktioniert: Es handelt sich um ein Applet, das es uns ermöglicht, eine E-Mail an den Benutzer zu senden, um ihn über die Anwesenheit des Postboten zu informieren. Dieses Applet wird vom Mikrocontroller ausgeführt, wenn der Benachrichtigungsstatus auf hoch gesetzt wird. Der Benutzer erhält dann drei Nachrichten in seinem E-Mail-Postfach und kann sich einloggen und die Plattform manipulieren.

* 1. **Auswahl der Komponenten**

Für die Durchführung dieses Projekts haben wir folgende Materialien verwendet:

|  |  |  |
| --- | --- | --- |
| **Bauteil** | **Anzahl** | **Kosten** |
| Website-Hosting | 1 | 39.31 € |
| ESP32-DevKit | 1 | 11.15 € |
| ESP32-CAM | 1 | 12.99 € |
| Breadboard-Kit | 1 | 5.99 € |
| Rote LED | 1 | 0.90 € |
| grüne LED | 1 | 1.60 € |
| blaue LED | 1 | 1.60 € |
| Widerstände | 3 | 0.06 € |
| Gesamtkosten | | 73.6 € |

1. Hardwareteil
   1. Design erstellen

Um den elektronischen Teil des Projekts zu entwerfen, mussten wir die Montage in der Software-Proteus 8 zeichnen. Diese Simulations- und CAD-Software ermöglicht es uns, elektronische Schaltpläne zu entwerfen, die später zu einer Serienfertigung in den Labors führen können.

Abbildung 2: Schaltplan in ISIS

* 1. Beschreibung der Verschiedenen elektronischen Komponenten
* ESP32 – DEVKITC: [[1]](#footnote-1)ist eine Reihe von System-on-a-Chip (SoC)-Mikrocontrollern von Espressif Systems, die auf der Xtensa LX6-Architektur von Tensilica (en) basieren, Wi-Fi und Bluetooth (bis LE 5.0 und 5.11) im Dualmodus verarbeiten und einen DSP integrieren.

Er enthält:

* ESP32-WROOM-32 Modul: Bei manchen Boards kann hier auch das größere ESP32- WROVER-Modul sein.
* Antenne
* Header-Pins: Die meisten Pins des ESP-Moduls sind auf die Pin-Header auf der Platine herausgeführt. Es gibt zwei Reihen zu je 19 Pins.
* Reset-Knopf: Bei manchen Boards ist dieser Knopf auch als EN bezeichnet.
* USB-Anschluss: Stromversorgung für das Board sowie die Kommunikationsschnittstelle zwischen einem Computer und dem ESP32-Modul.
* Boot-Knopf (Download-Knopf): Je nach gewählter Flash-Methode bzw. Entwicklungsumgebung ist der Knopf für das Aufspielen der Firmware zu betätigen. Es gibt da unterschiedliche Szenarien: – Arduino-IDE: Manchmal wird der Upload ohne zusätzliches Betätigen von Knöpfen initiiert. In der Regel müssen Sie aber den Boot-Knopf für das Initiieren des Uploads drücken. – Offiziell: Halten Sie den Boot-Knopf gedrückt, um den Firmware-DownloadModus zum Herunterladen der Firmware über die serielle Schnittstelle zu starten und drücken Sie ggf. anschließend EN für einen Reboot.
* LED: Sie leuchtet kurz auf, wenn an das Board Spannung angelegt wird. (Bei manchen alternativen Boards ist die LED auch mit GPIO2 verbunden.)
* USB-UART-Bridge: Verbaut ist eine CP2102 USB to UART Bridge, die Transferraten bis zu 3 Mbps unterstützt

Seine Stromversorgung ist folgendes:

* Micro-USB-Port: Dies wird als hauptsächliche Stromversorgungsoption gesehen. Da USB normalerweise mit +5 V arbeitet, wird die Spannung durch einen OnBoard-Chip auf +3,3 V reduziert. Warnung: Periphere Geräte können nicht ausnahmslos und unbesehen an den 3V3-Ausgang angeschlossen werden, da der OnBoard-Chip nur begrenzt belastbar ist.
* 5V/GND-Header-Pins
* 3V3/GND-Header-Pins. Je nach Einsatz (z. B. intensive WLAN-Nutzung) kann der ESP32 viel Strom ziehen. Laut dem Datenblatt können dies bis zu 500 mA sein.

Abbildung 3: ESP32-DevKitC V4 mit ESP32-WROOM-32

* ESP32-CAM: [[2]](#footnote-2)Die ESP32-CAM ist ein sehr kleines Kameramodul, das mit dem ESP32-S-Chip ausgestattet ist. Neben der Kamera OV2640 und mehreren GPIOs zum Anschluss von Peripheriegeräten verfügt es auch über einen microSD-Kartensteckplatz, der nützlich sein kann, um die mit der Kamera aufgenommenen Bilder zu speichern oder um Dateien zu speichern, die den Kunden serviert werden sollen.

Hier ist eine Liste der Funktionen des ESP32-CAM:

* Das kleinste SoC-Modul 802.11b/g/n Wi-Fi BT.
* Stromsparende 32-Bit-CPU, kann auch als Anwendungsprozessor verwendet werden.
* Taktgeschwindigkeit bis zu 160MHz, summarische Rechenleistung bis zu 600 DMIPS.
* Integriertes 520 KB SRAM, externes 4MPSRAM.
* Unterstützt UART/SPI/I2C/PWM/ADC/DAC.
* Unterstützt die Kameras OV2640 und OV7670, eingebaute Blitzlampe.
* Unterstützt das Hochladen von Bildern über WiFI.
* Unterstützt TF-Karte
* Unterstützt mehrere Standby-Modi
* Eingebettet Lwip und FreeRTOS
* Unterstützt den Betriebsmodus STA/AP/STA+AP.
* Unterstützt die Smart Config/AirKiss-Technologie.
* Unterstützt lokale und entfernte Firmware-Updates über die serielle Schnittstelle (FOTA)
* Es gibt drei GND-Pins und zwei Pins für die Stromversorgung: entweder 3,3 V oder 5 V.

Abbildung 4: ESP32-CAM mit ESP32-S Chip

* Widerständen: Der elektrische Widerstand ist der Widerstand, den die Spannung in einem Stromkreis durch den elektrischen Leiter erfährt. Dadurch wird die Stromstärke verringert. Der Widerstand übernimmt die Aufgabe, elektrische Bauteile vor hohen Stromstärken zu schützen.
* LEDs (Light Emitting Diodes) sind winzige elektronische Chips, die aus speziellen Halbleitern bestehen. Wenn Strom durch diesen Festkörper fließt, beginnt er zu leuchten; er "emittiert" Licht. In der Beleuchtungstechnik wird dieser Vorgang als "Elektrolumineszenz" bezeichnet. Sie fungiert bei der Simulation als Zeuge. Bei der Durchführung soll ein elektrischer Türöffner verwendet werden, der mit 12 V Wechselstrom betrieben wird und mit einem 3,3 V / 12 V Relais verbunden ist.
  1. Mikrocontroller Programmierung

Abbildung 3: Programmieralgorithmus des Mikrocontrollers

**Erklärung:**

Ganz zu Beginn erstellen und initialisieren wir einige Variablen wie den Namen des WLANs, das Passwort, einen Schlüssel für unsere API und die Zustände von Tür und Kamera. Wir initialisieren auch die Ausgänge des Mikrocontrollers, die für unsere Geräte verwendet werden sollen, z. B.: Pin 27 ist die Tür, Pin 26 die Kamera und Pin 23 eine LED, die uns die Verbindung mit dem WLAN signalisiert.

Dann stellen wir die Wifi-Verbindung her und vergewissern uns, dass wir mit einem Wifi verbunden sind, weshalb es eine While-Schleife gibt, die den Mikrocontroller dazu bringt, einen weiteren Versuch zu unternehmen, eine Verbindung zum Wifi herzustellen. Wenn die Verbindung zum WLAN hergestellt ist, können wir uns mit unserem Webserver verbinden. Wenn der Mikrocontroller dies nicht schafft, wird eine Fehlermeldung angezeigt, andernfalls verbinden wir uns mit einer API, die wir zuvor entworfen haben und die uns den Status der Tür, der Kamera und der Benachrichtigung in Abhängigkeit vom Parameter Benutzername mitteilen soll. Diese API ist eine PHP-Datei, die auf dem Server unter dem Pfadnamen "/API/read\_all.php" gespeichert ist. Nachdem wir die Daten aus der Datenbank in einem JSON-Format (PayLoad) gelesen haben, können wir diese Daten jeder der oben definierten Variablen (Tür-, Camara- und Benachrichtigungsstatus) zuweisen. Anschließend können wir die Werte der ausgewählten Pins des Mikrocontrollers aktualisieren oder überschreiben.

Wir testen, ob wir eine Benachrichtigung erhalten haben, und wenn ja, dann verbindet sich der Mikrocontroller wie folgt mit der API: String ("POST ") + "/trigger/notification/with/key/"+apiKey + " HTTP/1.1\r\n" +"Host: maker.ifttt. com" + "\r\r" + "Content-Type: application/x-wwww-form-urlencoded\r\r" + "Content-Length: 13\r\r\r" + "value1=" + notifString + "\r\r", und führt dann das Applet aus, das die in der Variablen notifString gespeicherte Nachricht als E-Mail an den Benutzer senden soll. Der Pfad zur API-Datei lautet "maker.ifttt.com/trigger/notification/with/key/"+apiKey, wobei apiKey eine Variable ist, die den Schlüssel der API enthält. Durch Aufruf der POST-Methode werden die Daten an den Server "maker.ifttt.com" gesendet. Dieser Vorgang wird so lange ausgeführt, bis der Status der Notifikation von 1 auf 0 wechselt. Nach insgesamt 03 Datenübertragungen an den Server beendet der Mikrocontroller seine Verbindung zu diesem Server.

1. Softwareteil
   1. Webapplikation

Dies ist die Schnittstelle, auf der der Benutzer seine Manipulationen vornimmt, z. B. die Tür öffnen und schließen oder die Kamera ein- und ausschalten. Die Anwendung besteht aus zwei Teilen: dem Frontend, das für den Benutzer sichtbar ist, und dem Backend, das für den Benutzer unsichtbar und unzugänglich im Hintergrund läuft.

* + 1. Frontend

Wie bereits erwähnt, ist das Frontend die Spitze des Eisbergs oder das, was der Nutzer der Anwendung sieht. Um es zu entwerfen, müssen wir zuerst ein Modell entwerfen, um die Position jedes Elements, die Farbe, die Größe oder die Schriftart zu bestimmen. Um dies zu tun, verwenden wir eine Online-Anwendung namens Figma. Anschließend legen wir die Anzahl der Seiten fest, die wir benötigen. In unserem Fall benötigen wir vier Seiten:

* eine Begrüßungsseite oder Homepage, um sich einzuloggen
* eine Anmeldeseite, um ein Konto zu erstellen
* eine Arbeitsseite, auf der der Benutzer seine Eingaben machen kann
* eine Seite für Postboten, die dem Postboten helfen soll, den Benutzer zu kontaktieren.

1. Begrüßungsseite

Wir beginnen mit der Startseite, die aus drei Blöcken besteht, die mit den anderen Seiten identisch sind: einem Header, einem Body und einem Footer. Der Body ist für uns von Interesse, da der Inhalt für jede Seite spezifisch ist. Der Benutzer gibt seinen Namen und sein Passwort ein, um sich anzumelden, aber falls er noch kein Konto hat, hilft ihm ein Link, auf die Seite zu gelangen, auf der er ein Konto erstellen kann.

Hier ist der Link <https://mypaket.org/Home2/Accueil.php>

Abbildung 4: Begrüßungsseite in Figma

1. Anmeldeseite

Auf dieser Seite kann der Nutzer dann ein neues Konto erstellen. Um ein neues Konto zu erstellen, muss der Benutzer einen Namen, ein Passwort und eine Adresse (hier ist der Ort gemeint) eingeben. Ein Fehler wird zurückgegeben, wenn nicht alle Informationen eingegeben wurden oder wenn der Benutzer versucht, ein neues Konto mit demselben Namen, Passwort oder derselben Adresse zu erstellen. Wenn er die Erstellung seines Kontos ohne Fehlermeldung abgeschlossen hat, kann er auf die Schaltfläche "ich schon ein Konto" klicken und wird automatisch zur Homepage weitergeleitet, wo er seine Daten für die Unterkunft eingeben kann.

Abbildung 5: Anmeldeseite in Figma

1. Arbeitsseite

Diese Seite ist die Arbeitsseite des Benutzers, da er hier alle seine Manipulationen vornehmen wird. Auf dieser Seite haben wir ein Bild einer Tür, die ihren Status ändert (d.h. offen oder geschlossen), wenn der Benutzer daraufklickt, dasselbe gilt für die Kamera. Ein kleines Bild einer Glocke symbolisiert den Status der Benachrichtigungen, die vibriert, um das Vorhandensein von Benachrichtigungen mit der Nachricht "Postbeamte ist schon da" anzuzeigen. Der Nutzer kann einfach auf die Glocke klicken, um die Benachrichtigung zu entfernen, und die Nachricht wird ausgeblendet.

Abbildung 6: Arbeitsseite in Figma

1. Seite für Postboten

Diese Seite ist für den Postboten bestimmt, der den Nutzer erreichen kann, um ihn darüber zu informieren, dass er schon da ist. Der Postbeamte gibt den Namen des Nutzers ein (den Namen, der auf dem Paket steht) und klickt dann auf die Schaltfläche "klingelt". Dadurch wird der Nutzer erreicht und die Benachrichtigung wird aktiviert.

Abbildung 7: Seite für Postboten in Figma

Nachdem wir die Modelle entworfen haben, können wir nun zur Kodierung übergehen, indem wir die Auszeichnungssprache HTML verwenden, mit der wir unsere Elemente wie Schaltflächen, Bilder und Texte im Browser positionieren und erstellen können. Für die Stilisierung und eine gute visuelle Darstellung haben wir CSS und das Framework Bootstrap 5 verwendet. Um die Interaktionen des Benutzers mit den Elementen zu verwalten, haben wir JavaScript verwendet.

* + 1. Backend

Wir haben gesehen, dass das Frontend allein eine Website nicht performant und sicher machen kann, d.h. unsere Website ist statisch, was bedeutet, dass alle Benutzer auf die gleiche Seite zugreifen und sehen können, was die anderen Fonds haben, so dass niemand seine eigene Sitzung hat. Um dies zu lösen, führen wir ein Datenbanksystem ein, das für jeden Benutzer seine persönlichen Informationen speichert und die Seite jedes Benutzers wird anders gestaltet als die der anderen, je nachdem, wie der Benutzer sie konfiguriert. PHP und SQL sind die Sprachen, die diese Aufgabe übernehmen können. Wenn sich ein Benutzer zum Beispiel anmelden möchte, gibt er seine Daten (Name und Passwort) ein, die dann als Abfrage an den Webserver gesendet werden. Der Server führt die Anfrage aus und sendet die Antwort an den Benutzer zurück, die dann vom Browser ausgeführt werden, kann. In diesem Fall spricht man von einer statischen Website, ein Client/Server-Website oder einer Anfrage-Antwort-Website. [[3]](#footnote-3)In der Regel basieren dynamische Websites auf dem Datenbanksystem MySQL und den Programmiersprachen Perl, PHP oder Ruby.

Abbildung 8: Dynamische Webseite Prinzip

Wir müssen zunächst einen Webserver einrichten, das kann man tun:

* lokal, hier wird unsere Website nicht im Internet sichtbar sein, sondern nur für uns. Das ist eine Lösung, die man verwendet, wenn man etwas ausprobieren oder lernen will, ohne wirklich Geld auszugeben.
* Online über einen Webhosting-Anbieter. Diese Lösung ist praktischer, aber auch teuer, denn um eine Website zu hosten, muss man je nach Anbieter einen bestimmten Betrag bezahlen.

In unserem Fall entschieden wir uns für Bluehost, nicht nur wegen des Preises, sondern auch wegen der positiven Meinungen der Nutzer über den Kundenservice, die Datensicherheit und die Möglichkeit, WordPress mit nur einem Klick zu installieren und dann zu verwenden.

Nun haben wir einen Webserver. Wir beginnen damit, unsere Datenbank mit dem Namen "mypaketo\_projekt" mit den folgenden Feldern zu erstellen:

* id: Eindeutige Kennung vom Typ integer, die sich selbst erhöht.
* username: ein Benutzername des Typen varchar
* password: ein Passwort des Typen varchar
* adresse: eine Adresse des Standorts des Typen varchar
* state\_tuer: der Status der Tür vom Typ integer
* state\_camera: der Status der Kamera vom Typ integer
* notification: der Status der Benachrichtigung vom Typ integer

Abbildung 9: DataBase erstellen

Nachdem wir nun alle Felder definiert haben, die in unserer Datenbank verfügbar sein werden, können wir mit der funktionalen Analyse fortfahren. Diese Analyse wird mit Hilfe der UML-Sprache durchgeführt, in der zwei Diagramme verwendet werden: das Use Case-Diagramm und das Sequenzdiagramm.

1. Use Case-Diagramm

[[4]](#footnote-4)Ein Use Case Diagramm ist eine grafische Repräsentation von Anwendungsfällen inklusive deren Beziehungen zur Umwelt und zu anderen Anwendungsfällen. Damit beschreibt es in einer hohen Abstraktion, welche Funktionen und Dienste ein System für einen Anwender bereitstellt.

Für unser Projekt haben wir drei Anwendungsfälle:

* Erstellung eines Kontos: Nach diesem Schritt kann der Benutzer die Anwendung nutzen.
* Registrierung: Nachdem Sie ein Konto erstellt haben, müssen Sie sich natürlich registrieren. Im Falle eines Betrugs wird der Nutzer sofort benachrichtigt und weiß, was er tun muss, um ihn zu beheben.
* Nutzung des Kontos: Hier kann der Nutzer alle Manipulationen auf der Plattform vornehmen.

Abbildung 10: Use Case-Diagramm

1. Sequenzdiagramm

[[5]](#footnote-5)Ein Sequenzdiagramm ist eine Art von Interaktionsdiagramm in der Unified Modeling Language (kurz: UML) und bildet die Interaktion zwischen einer Gruppe von Objekten sowie die Reihenfolge ab.

Wir haben für jedes Use Case Diagramm ein Sequenzdiagramm erstellt.

* Konto erstellen:

Der Benutzer öffnet die Anwendung, die Anwendung bietet ihm an, ein Konto zu erstellen, indem sie einen Link zu dieser Seite angibt. Der Benutzer geht zu dieser Seite und gibt die geforderten Informationen ein, andernfalls wird eine explizite Fehlermeldung angezeigt. Diese Informationen werden dann an den Server gesendet, der sie verarbeitet und in der Datenbank speichert, falls der Benutzer noch kein Konto erstellt hat, andernfalls wird eine Fehlermeldung angezeigt.

Abbildung 11: Konto erstellen

* Konto anmelden:

Der Benutzer hat bereits ein Konto erstellt, jetzt muss er sich registrieren, um die Anwendung zu nutzen. Der Benutzer startet die Anwendung, wird aufgefordert, sich zu registrieren, und gibt seine Daten ein, die dann überprüft und an den Server gesendet werden. Wenn die Überprüfung erfolgreich war, wird der Benutzer direkt zu seiner Arbeitsseite oder Sitzung weitergeleitet, andernfalls wird eine Fehlermeldung angezeigt.

Abbildung 12: Konto anmelden

* Konto benutzen:

Der Nutzer hat bereits ein Konto und kann sich auch darauf anmelden. In diesem Fall ist er in der Lage, die Anwendung zu nutzen. Wenn der Benutzer sich registriert, werden seine Informationen an den Server gesendet, der die Existenz des Clients überprüft und eine Antwort zurückgibt, die vom Browser des Clients verarbeitet wird, indem er den aktuellen Status der Elemente (Tür und Kamera) anzeigt, das gleiche Prinzip gilt, wenn er die Seite aktualisiert. Wenn der Benutzer auf ein Element (z. B. die Tür) klickt, ändert er den Status dieses Elements, der sofort in der Datenbank gespeichert wird. Der Server gibt dann eine Antwort (Status der in der Datenbank gespeicherten Elemente) an die Anwendung zurück, die sich durch die Änderung des Bildes der Benutzerseite oder des Frontend materialisiert.

Abbildung 13: Konto benutzen

* 1. QR-Code erstellen

Wir werden hier einen QR-Code erstellen, der die Adresse der Seite des Postboten liefert, so dass dieser den Code einfach scannen und direkt auf die Seite gehen kann. Die Erstellung des Codes erfolgt in Python mithilfe der Bibliothek "qrcode". Das Ergebnis sieht wie folgt aus.

Abbildung 14: QR-Code erstellen

* 1. API erstellen

[[6]](#footnote-6)Eine Programmierschnittstelle (oder Anwendungsschnittstelle), häufig auch nur kurz API (Application Programming Interface) ist Code, der es zwei Softwareprogrammen ermöglicht, miteinander zu kommunizieren.

Eine API definiert die korrekte Art und Weise, wie ein Entwickler Dienste von einem Betriebssystem (OS) oder einer anderen Anwendung anfordert und Daten in verschiedenen Kontexten und über mehrere Kanäle ausgibt.

Wir wollen eine API erstellen, die es uns ermöglicht, bestimmte Benutzerdaten aus der Datenbank abzurufen und an den Mikrocontroller zu senden. Diese Daten sind der Status der Tür, der Kamera und der Benachrichtigung. Natürlich ist diese API eine PHP-Datei, die SQL-Befehle enthält, um die Daten aus der Datenbank zu lesen.

Wir müssen uns mit der Datenbank verbinden, indem wir ein neues Objekt "PDO" erstellen, das als Parameter den Servernamen, den Namen der Datenbank, das Passwort und den aktuellen Benutzernamen enthält. Wir testen, ob ein Benutzername eingegeben wurde (dies ermöglicht uns, eigene Informationen über einen Benutzer zu erhalten), und wenn ja, bereiten wir die SQL-Abfrage vor: "SELECT \*FROM table\_name WHERE username = $\_GET['username']", was bedeutet: Wählt alle Elemente der Tabelle "table\_name" aus, wenn das Feld "username" der Tabelle gleich dem oben angeforderten ist.

Nach der Ausführung der Abfrage erhalten wir ein Ergebnis in Form einen assoziativen Array, in der die Schlüssel den Feldern der Datenbankelemente entsprechen. In unserem Fall benötigen wir die Felder "state\_tuer", "state\_camera" und "notification", die wir in einer Tabelle speichern. Wenn wir kein Ergebnis erhalten oder kein Benutzername eingegeben wurde, zeigen wir einen Fehler an

Abbildung 15: API-Erstellungsalgorithmus

Wir können uns das Ergebnis in "Postman" ansehen. https://mypaket.org/API/read\_all.php?username=Admin, ist die so erstellte API, oder "https://mypaket.org" ist der Server, auf dem sich die PHP-Datei befindet, "/API/" ist der Ordner, in dem sich die Datei befindet, "read\_all.php" ist die Datei selbst, "username=Admin" ist der Benutzer, dessen Informationen wir haben möchten.

Abbildung 16: API erstellen

* 1. Einrichtung des Notifikationssystems

Wir müssen hier ein System schaffen, das den Nutzer darüber informiert, dass der Postbote da ist. Wir dachten also an ein E-Mail-Benachrichtigungssystem, das einen kostenlosen Online-Dienst namens IFTTT ("If This Then That") verwendet.

[[7]](#footnote-7)IFTTTT ist eine Plattform, die Ihnen die kreative Kontrolle über Dutzende von Produkten und Anwendungen ermöglicht. Sie können Apps zusammenarbeiten lassen. Wenn Sie z. B. eine bestimmte Anfrage an IFTTT senden, wird ein Applet ausgelöst, das etwas auslöst, z. B. eine E-Mail-Benachrichtigung.

Um diesen Dienst nutzen zu können, müssen wir zunächst ein Konto einrichten. Die Schritte sind wie folgt:

* Gehen Sie auf die offizielle Website https://ifttt.com/ und klicken Sie auf Get Started, um ein Konto zu eröffnen, oder auf Signup für diejenigen, die bereits ein Konto haben.
* Als Nächstes müssen Sie in IFTTT ein Applet erstellen. Ein Applet verbindet zwei oder mehr Anwendungen oder Geräte miteinander.

Dieses Applet besteht aus:

* Triggers: Weisen ein Applet an, zu starten. Der ESP32 wird eine Anfrage (WebHooks) senden, die das Applet auslöst.
* Aktionen: sind das Endergebnis der Ausführung eines Applets. In unserem Fall das Versenden einer E-Mail.

1. Simulation
   1. Ergebnisse

Nach der Simulation erhielten wir die folgenden Ergebnisse:

Wir kommen auf die Startseite und dort müssen wir ein Konto erstellen, also klicken wir auf den Weiterleitungslink zur Seite für die Kontoerstellung. Der Link ist: https://mypaket.org/Home2/Accueil.php

Abbildung 17: Schritt 1 Kontoerstellung

Jetzt müssen wir die geforderten Informationen eingeben. Dann klicken wir auf "Erstellen" und kehren danach zur Startseite zurück.

Abbildung 18: Schritt 2 – Informationen eingeben

Wir können uns nun einloggen. Nachdem wir unsere Daten eingegeben haben, klicken wir auf "Anmelden".

Abbildung 19: Schritt 3 - einloggen

Wir kommen auf die Arbeitsseite, wo alle Manipulationen vorgenommen werden. Hier sehen Sie, dass alles geschlossen ist und dass Sie keine Benachrichtigungen erhalten.

Abbildung 20: Schritt 4 - Arbeitsseite öffnen

Hier können wir sehen, wie dieser Benutzer mit einem verschlüsselten Passwort in die Datenbank aufgenommen wurde, dies dient der Datensicherheit.

Abbildung 21: schritt 5 - Benutzer in der Datenbank hinzugefügt

Wenn der Postbeamte nun den QR-Code gescannt hat, der eine Adresse angibt, zu der er gehen und so den Kunden über seine Anwesenheit informieren kann. Er gibt den Namen des Nutzers ein und klickt auf "Klingelt".

Abbildung 22: Schritt 6 – Postbeamte Seite

Wir erhalten eine Benachrichtigung in Form einer E-Mail:

Abbildung 23: Schritt 7 – Notifikation als E-Mail erhalten

Wir gehen nun in die Anwendung und stellen tatsächlich fest, dass eine Benachrichtigung eingetroffen ist.

Abbildung 24: Schritt 8 – Notifikation im Webapp

Abbildung 25: Schritt 9 - Tür öffnen im Webapp.

Abbildung 26: Schritt 9 - Tür öffnen

Dann aktiviert man die Kamera

Abbildung 27: Schritt 10 - Camara öffnen im Webapp

Abbildung 28: Schritt 10 - Camara öffnen

Nachdem der Postbote das Paket abgegeben hat, kann der Benutzer nun die Tür schließen und die Kamera ausschalten.

Abbildung 29: Schritt 11 - Alle sind im Webapp zu

Abbildung 30: Schritt 11 – Alle sind zu

* 1. Probleme

Während der Durchführung dieses Projekts sind wir mit Schwierigkeiten konfrontiert, von denen einige gelöst werden konnten und andere, für die wir leider keine angemessene Lösung finden konnten, vielleicht hätten wir mit ein wenig mehr Zeit unsere Nachforschungen vertiefen können, um das Problem zu lösen. Als Problem haben wir folgendes:

* Lesen von Daten aus der Datenbank über eine Android-Anwendung
* Es ist unmöglich, Bilder von einer Videokamera in einem anderen Netzwerk als dem, mit dem sie verbunden ist, zu erhalten. Im lokalen Netzwerk geht alles gut, aber wenn man das Netzwerk wechselt, hat man nichts mehr.
* Aus Zeitmangel konnten wir das Türöffner System für die Tür mit Relais realisieren. Wir haben uns nur auf die Simulation beschränkt.

1. Erweiterung

Als zukünftige Verbesserungen für dieses Projekt haben wir uns Folgendes überlegt:

* ein System zur Gesichtserkennung einzuführen, wofür wir mit der Post zusammenarbeiten müssen.
* eine Lösung finden, um die Anwendung auf mobilen Geräten, d. h. auf Smartphones, laufen zu lassen.

1. Zusammenfassung

Zusammenfassend können wir sagen, dass dieses Projekt uns ermöglicht hat, noch weiter über unser Wissen hinauszugehen, und die Tatsache, dass wir in Gruppen arbeiten konnten, hat uns geholfen, unsere Schwierigkeiten zu erkennen. Der gegenseitige Austausch hilft uns, schneller in diesem Bereich zu forschen. Am Anfang wussten wir nicht, wie wir das Projekt umsetzen sollten, da es keine existierende Grundlage gab, an der wir uns orientieren konnten, aber durch Suchen und Ausprobieren, Simulation für Simulation, konnten wir Ideen entwickeln und uns auch von einigen Themen inspirieren lassen, die dem Projekt nahekamen. Dieses Projekt ist noch nicht ganz abgeschlossen, da es Teile gibt, die wir nicht ausreichend bearbeitet und erforscht haben. Aufgrund von Fehlern werden wir in Zukunft Verbesserungen vornehmen. Dieses Projekt ist nur eine Testversion, eine Demo.

Diese Anwendung wäre für jeden nützlich und würde die Arbeit der Briefträger und Postboten erleichtern und das Leben der Nutzer dieses Systems angenehmer machen.

**Quellcode-Link:**

* https://github.com/Steve2404/Projekt1.git

**Quellen:**

* <https://www.manomano.fr/cat/diode+led+rouge>
* <http://www.fit4php.net/einstieg/einrichten-der-arbeitsumgebung/client-server-modell/>
* <https://t2informatik.de/wissen-kompakt/use-case-diagramm/>
* <https://www.youtube.com/watch?v=eIvp8e_MDB0>
* <https://www.lucidchart.com/pages/de/uml-sequenzdiagramme>
* <https://www.computerweekly.com/de/definition/Programmierschnittstelle-API>
* <https://randomnerdtutorials.com/esp32-door-status-monitor-email/>
* <https://fr.wikipedia.org/wiki/ESP32>
* <https://www.seeedstudio.com/media/catalog/product/cache/ef3164306500b1080e8560b2e8b5cc0f/b/a/bazaar1003542_esp32cam2.jpg>
* <https://www.heinzinger.de/glossar/elektrischer-widerstand/#:~:text=Der%20elektrische%20Widerstand%20ist%20der,Sinne%20wie%20eine%20mechanische%20Bremse>
* <https://iotmonk.com/lessons/creating-your-own-iot-cloud-from-scratch-using-php-mysql-esp12e-part-4/>
* Udo Brandes, Mikrocontroller ESP32. RheinWerk Technik

1. <https://fr.wikipedia.org/wiki/ESP32> [↑](#footnote-ref-1)
2. https://randomnerdtutorials.com/esp32-cam-video-streaming-face-recognition-arduino-ide/ [↑](#footnote-ref-2)
3. http://de.wikipedia.org/wiki/Webseite#Dynamische\_Webseiten [↑](#footnote-ref-3)
4. https://t2informatik.de/wissen-kompakt/use-case-diagramm/ [↑](#footnote-ref-4)
5. https://www.lucidchart.com/pages/de/uml-sequenzdiagramme [↑](#footnote-ref-5)
6. https://www.computerweekly.com/de/definition/Programmierschnittstelle-API [↑](#footnote-ref-6)
7. https://randomnerdtutorials.com/esp32-door-status-monitor-email/ [↑](#footnote-ref-7)