BEUTH HOCHSCHULE FÜR TECHNIK BERLIN
University of Applied Sciences

Beuth Hochschule für Technik Berlin
Fachbereich VI - Informatik und Medien
Luxemburger Str. 10, 13353 Berlin

Bachelorarbeit

Entwicklung einer Datenbank-Applikation für das "acaLoan-Raspi"-Projekt des PSE-Labors

Development of a database application for the "acaLoan-Raspi" project of the PSE laboratory

Oleksandra Baga

Beuth Hochschule für Technik Berlin
Matrikelnummer 849852
Bachelor of Engineering (B.Eng.)
Computer Engineering - Embedded Systems
E-Mail: oleksandra.baga@gmail.com

Supervisor Prof. Dr. Christian Forler
Fachbereich VI - Informatik und Medien
Beuth Hochschule für Technik Berlin

31.10.2020

Contents

Abbildungsverzeichnis

Abkürzungsverzeichnis

AJAX Asynchrones Javascript und XML

DOM Document Object Model

HTTP HyperText Transfer Protocol

JSON JavaScript Object Notation

Raspi Raspberry Pi Board und Minicomputer

RFID Radio-frequency identification

URI Uniform Resource Identifier, Unified Resource Identifier

1 Einleitung

1.1 Motivation und Aufgabestellung

Die vorliegende Arbeit beschäftigt sich mit der Einwicklung einer Datenbank-Applikation für das PSE-Labor[1] (Labor für Pervasive Systems Engineering), das sich an der Beuth Hoschschule für Technik Berlin befindet und seit fast 5 Jahre ein wichtiger Teil des Studiums im Studiengänge Technische Informatik und Medieninformatik ist. Die zwei Mitaraber, Andreas Döpkens und Brian Schüler, beitragen selbstmotiviert zur den Projekte, die zwar räumlich im PSE-Labor der Beuth-Hochschule sind und betrieben werden können[2], wurden aber als "das acaLab" genannt, bei dem es sich sozusagen um ein virtuelles Labor in einem reellen Labor handelt. Die acaLab-Projekte sollen als interessante Mitarbeiter-Beiträge neben den individuellen Tätigkeiten im Fachbereich 6 der Beuth-Hochschule bereichern und werden deshalb auch exemplarisch jedes Jahr auf der Langen Nacht der Wissenschaften[3] gezeigt. An dieser Stelle ist auch noch anzumerken, die acaLab-Projekte aus den Mitteln des PSE-Labors finanziert werden und damit möglichst viele Studierende für lehr- und erkenntnisreichen Abschlussarbeiten eingeladen sind, de der Vergabe die Aufgaben aus den acaLab-Projekten als Abschlussarbeiten spart dem Fachbereich ein Budget. Die Erkenntnisse und Ergebnisse aus den acaProjekt-Tätigkeiten sind später in die Lehre einfließen zu lassen. Meine Abschlussarbeit widmet sich dem neuen Projekt des Labors namens "acaLoan-Raspi".

Während im PSE-Labor stattfindenden Übungsveranstaltungen im Studiengang Technische Informatik werden vorhandene im PSE-Labor die Raspberry Pi Minicomputers (kurz: Raspi) an die Studierenden verliehen. Zu Beginn einer Laborübung werden die Raspi Boards den Studierenden vom Lehrkraft, der die Übung betreut, übergeben und am Ende der Laborübung zurückgezogen. Aus 16 vorhandenen im Labor Raspis, die markierte mit den Nummern 12-16 von Studierenden nach Hause (home-loan) genommen werden können.

Nachweislich ist das Vorgehen oft mit Reihe von Problemen verknüpft, die sich jedes Semester und fast jedes Mal wiederholen. Die folgenden Problemen wurden

[1] https://www.http://labor.beuth-hochschule.de/pse
[2] https://labor.beuth-hochschule.de/pse/acalab-aktivitaeten-und-abschlussarbeiten/
[3] https://www.langenachtderwissenschaften.de/

von Mitarbeitern des Labors bereits festgestellt und regelmäßig verlangsamen den Prozess der Verleihung und Übungsführung:

- Studierende kennen ihre am Semesteranfang zugewiesene Gruppennummer auch nach mehreren Wochen nicht und geben den Lehrkraft einen Board mit einer falschen Registriernummer, der einer anderen Gruppe früher zugewiesen wurde und nur von der zugewiesenen Gruppe benutzt werden darf.

- Studierende versuchen einen Board nach Hause auszuleihen, der zu den Lab-Boards gehört und nur im Labor während der Übungszeit verliert werden darf. Außerplanmäßig von Studierenden darf Lab-Board nicht ausgeliehen und auch mit nach Hause (home-loan) nicht genommen werden.

- Es gibt ein Verwaltungsaufwand für die ausleihbaren Home-Boards, die von den Studierenden für jeweils eine Woche mit nach Hause genommen werden können. Die Mitarbeiter müssen handlich die Studentenname, Matrikelnummer, Board und Zeit am Zettel registrieren und in einer Woche überprüfen, ob alle ausgeliehenen Boards pünktlich ins Labor zurückgekommen sind.

- Erfahrungsgemäß können Studierende nach Ablauf der Frist ein Ausleihgerät in einem sehr üblen Zustand der Verschmutzung oder Zerstörung zurückgeben, dass es besteht eine Notwendigkeit den Zustand des Gerätes stets zu kontrollieren, damit es immer bekannt wird, zum welchen Zeitraum Raspi Board zum letzten Mal funktionsfähig war und von wem ausgeliehen wurde.

- Falls gilt ein Raspi Board als verloren, es sollte eine Möglichkeit geben, alle vorherigen Ausleihen anzuschauen und festzustellen, von welchem Studierende es ausgeliehen und nicht zurückgegeben wurde. Mit den Zettelchen, auf denen einen Name von Studierende und eine Board Nummer gemerkt werden, ist es zu aufwändig nachvollziehen.

Somit ist schlusszufolgern, dass eine Notwendigkeit das Verleihprozedere für die Loan-Boards (Lab und Home) mit modernen Mitteln der Technischen Informatik zu lösen schon dringend besteht und eine lohnenswerte Aufgabe für zukünftige Abschlussarbeit ist.

1.2 Aufbau der Arbeit

Obwohl das Thema der Bachelorarbeit "Entwicklung eine Datenbank-Applikation" lautet, ist es keine einzige Aufgabe nur eine Datenbank zur Ausleihverwaltung zu

schrieben ist. Es lässt sich die folgenden Struktur definieren, indem drei verteilte Teilen zu entwickeln ist.

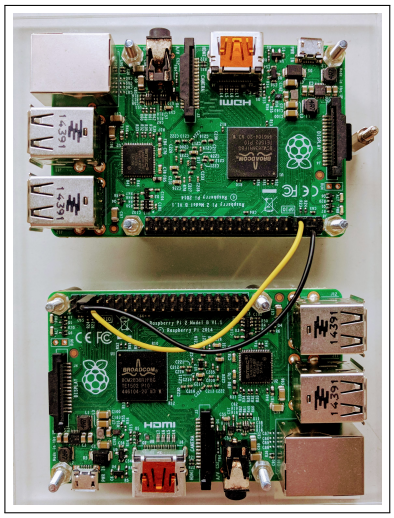
Erstens wird an einem uComputer ein sogenannten Register-Client realisiert. Dafür ein RFID-Leser an uComputer angeschlossen wird. Register-Client ist neben dem Eingangstür eines kleinen Lagerraums des PSE-Labors zu platzieren ist, wo Raspi-Boards aufbewahrt werden. Es ist geplant, dass Studierende einen Board selbst aus dem Fach nehmen könnte und dann mit Hilfe des Register-Clients den genommenen Board auf sich oder seine Gruppe registrieren lassen. Der Register-Client hat selbst keinen Zugriff zur Datenbank und sollte nur die abgelesene Daten von der Smartcard der Studierende zur Server schicken.

Zweitens ist ein Display-Client zu entwickeln, der den Studierenden es zulässt, die Begrüßung des System und eine Beschreibung die für Ausleihe notwendigen Schritten zu sehen. Es sollte in einem Browser-Fenster die aktuelle Server-Kommunikation und Auskunft angezeigt wird, ob die Ausleihe gelang oder ein Fehler aufgetreten war. Ein Android/iOS-Tablett ist eine gute Wahl für die technische Realisierung, da es die Kommunikation zwischen den Mensch und das System leicht und ohne erweiterte Hinweise zulässt.

Drittens ist ein Web-Server für die Datenbank-Applikation schließlich zu implementieren. Er umfasst alle Datensätze über die vorhandenen im Labor Raspi-Boards, registrierten zum Kurs Studierenden und die abgewickelten Leihvorgänge. Web-Server wird mit einem Web-Framework Django erstellt. Django verfügt nun über die Funktionalität und Datenbasis, um die dafür erforderlichen Aktionen durchzuführen. Als Web-Framework bietet Django eine Reihe von Komponenten und Funktionen (Benutzerauthentifizierung, Hochladen von Dateien, Umgang mit Daten usw.), die bei jeder Webanwendungen benötigt werden. Mit einem Web-Framework muss ein Entwickler keine Zeit damit verschwenden, denselben Code von Grund auf neu jedes Mal zu schreiben.

Wie erfolgt nun die Abwicklung des eigentlichen Leihvorgangs von der Ausleihe bis zur Rückgabe eines Boards? Zuerst wird eine Studentenkarte am Register-Client ablesen und nachdem sollte einen Name und die Anzahl schon ausgeliehenen Boards am Display-Client angezeigt werden. Falls der Studierende zum Kurs zugelassen ist, darf dann ein gewünschten Board am Register-Client abgelesen werden. Es ist möglich, dass zu den schon ausgeliehenen Lab-Board noch zusätzlich einen Home-Board nach Hause mitgenommen wird. Das abgelesene Board ist entweder auszuleihen oder zurückzugeben. Sämtliche im Verleih befindlichen Geräte werden von den Mitarbeitern des Labors regelmäßig nach jeder Übung vor dem nächsten Ausleihevorgang auf Funktionsfähigkeit geprüft. Es kann sein, dass einem Studierenden die Home-Loan-Absicht eines Boards (12-16) verweigert wird, da in der Vergangenheit schon

einmal vom Studierende ein Board in einem inakzeptabel Zustand zurückgeben war und die Ursachen mit den Mitarbeiter des Labor nicht geklären hat. Falls der Studierende, dem Home-Loan verboten wurde, ein Board auszuleihen versucht, wird eine Fehlermeldung auf dem Bildschirm gezeigt und der Leihvorgang mit dem Fehlerzustand terminiert.



Pic. 1.: Das Raspi Loan-Board

Nachdem nun grundlegenden Funktionsweisen der Bestandteilen des Verteilten Systems geklärt sind, geht es zur Auswahl der Hardware. Der wichtigste Aspekt beim Hardwarekauf ist einerseits das vorhandene Budget des PSE-Labors und zum Anderen die Aufgaben, die zusammenspielenden Hardware erfüllen sollen. Zunächst müssen die 16 für die Ausleihe zur Verfügung stehenden Raspi Loan-Boards mit einem geeigneten RFID-Tag ausgestattet werden. Es ist selbstklebende NFC Tags zu verwenden, die klein und dünn sind und lassen sich auf der Rückseite des Schutzschirms zu befestigen und ein Aussicht und Funktionen des Geräts nicht zu beeinflussen. Es ist nicht unerwähnt zu lassen, dass Tags nicht ausgelesen werden können, wenn diese auf einen metallischen Gegenstand/Oberfläche geklebt wurden, da die Kommunikation aus physischen Gründen zwischen Tag und Lesegerät gestört werden kann. Die Raspi Loan-Boards sind mit einem Schutzschirm aus dem ein transparenter thermoplastischer Kunststoff geschützt und es wurde ins Labor getestet, dass die geklebte auf der Rückseite RFID-Tag lassen sich ablesen und den Board auf RFID Chip-Kartenleser zu platzieren. Erfolgreich getestet wurde RFID Leser/Schreiber ACR122U [4], der wurde auf der Basis der 13,56 MHz kontaktlosen (RFID) Technologie entwickelt. Der ACR122U USB Kartenleser unterstützt nicht nur Mifare ® Technologien, sondern auch alle vier Typen von NFC -Tags. Diese Anforderung ist für vorliegende Abschlussarbeit wichtig, da Studierenden-Ausweise der Beuth Hochschule mit dieser Technologie gelesen werden können. Desweiteren soll an dem oben genannten Register-Client der dort angeschlossene RFID Chip-Kartenleser über geeignete Software in Betrieb genommen und die eingelesenen Daten an den Server übermittelt werden. Dafür wird ein Raspberry Pi[5] verwendet, der zählt zu den beliebtesten Single Board Computern zält und lässt die andere Schnittstellen anzubinden. Der kompakter Einplatinencomputer wenig verbraucht Strom aber dennonch genügend Power für ein Linuxbetriebsystem liefert. Detaillierte Beschreibungen zu jeder getroffenen Auswahl finden Sie im entsprechenden Abschnitt des Kapitels "Theoretische Grundlagen"

[4] https://www.conrad.de/de/p/rfid-reader-acr122u-nfc-usb-802554010.html
[5] https://www.raspberrypi.org/

2 Theoretische Grundlagen

Als schon im Abschnitt "Einleitung" erwähnt wurde, bei der Aufgabestellung es um eine Entwicklung einer Webanwendung geht. Die zu realisierende Software basiert sich, wie die meisten Webanwendungen, auf einer Client-Server-Architektur, wobei der Client Informationen eingibt, während der Server die eingegebene Daten empfängt, bearbeitet und speichert. Eine Webanwendung ist ein Computerprogramm, das eine bestimmte Funktion unter Verwendung eines Webbrowsers als Client ausführt. Die Webanwendung kann so einfach wie ein Kontaktformular auf einer Website oder so komplex wie eine Textverarbeitungs- oder Bildbearbeitungsprogramm sein, die Sie auf Ihr Computer im Browser ausführen können. Um die Webanwendung zu starten, muss der Benutzer keine zusätzlichen Programme installieren. Sie wird auf jedem Gerät mit Browser und Internetzugang ausgeführt. Der Client ist nicht vom Betriebssystem auf dem Computer des Benutzers abhängig. Bei der Entwicklung von Webanwendungen müssen daher keine separaten Versionen für Windows, Linux, Mac OS und andere Betriebssysteme geschrieben werden. Zur Implementierung der Clientseite werden HTML, CSS, JavaScript und Ajax verwendet. Zum Erstellen der Serverseite von Webanwendungen werden Programmiersprachen wie PHP, ASP, ASP.NET, Perl, C / C ++, Java, Python, Ruby und NodeJS verwendet. In dem vorherigen Schritt wurde es die Entscheidung getroffen, die Serverseite mit Web-Framework Django zu erstellen. Django wird mit einem leichten Webserver geliefert, mit dem eine Website schnell zum Laufen gebracht werden kann, ohne Zeit mit der Einrichtung eines Servers verschwenden zu müssen. Wenn der Entwicklungsserver von Django gestartet wird, überwacht er die Codeänderungen in Echtzeit. Es wird nach dem Ändern des Codes automatisch neu geladen.

In der zu realisierenden Abschlussarbeit ist auch notwendig einen dritten Teil namens Register-Client zu entwickeln, an dem ein RFID-Leser angeschlossen wird. Der Register-Client verfügt selbst über keinen Datenbank und darf nur die abgelesene Daten dem Server schicken. Es geht um eine Simplex-Verbindung, da ein Nachrichtenverkehr asymmetrisch ist, weil der Register-Client keine Daten vom Server zurückbekommen darf und über den erfolgreiche oder gescheiterte Leihvorgang nicht wissen muss. Für die Implementierung des Register-Clients wird uComputer Raspberry Pi benutzt, der möglicherweise nicht der einzige Single-Board-Computer (SBC) auf dem Markt ist, aber bei weitem der beliebteste und schon zur Verfügung im PSE-Labor steht und ergänzend nicht geliefert werden muss. Der

Raspberry Pi wird von einer großen Anzahl von Menschen verwendet und viele offizielle und inoffizielle Ressourcen und Produkte umfasst - von Büchern und Zubehör bis zu Schulungen. Dank der Raspberry Pi Foundation werden regelmäßig neue SBC-Modelle veröffentlicht. Der Raspberry Pi eignet sich aufgrund seiner hohen Rechenleistung gut als Desktop-Computer und völlig für die Verwendung in oben geschrieben Zwecken passt .

2.1 Grundlagen der Webanwendungen

In diesem Kapitel werden die grundlegenden Begriffe zum Entwerfen und Erstellen eines Webanwendung erläutert. Hierbei liegt unser Fokus nicht auf den grafischen Aspekten der Browserfunktionalität (d. H. Dem Layout von Seiten, dem Rendern von Bildern). Stattdessen konzentrieren wir uns auf die Probleme im Zusammenhang mit der Verarbeitung von HTTP-Anfragen und -Antworten. Webanwendungen können abhängig von den verschiedenen Kombinationen ihrer Hauptkomponenten in verschiedene Bestandteilen unterteilt werden. Erstens wird Das Backend (Backend oder Serverteil der Anwendung) auf einem Remotecomputer ausgeführt, der von dem Endnutzer weit entfernt werden kann oder im anliegenden Raum stehen kann. Es ist in verschiedenen Programmiersprachen zu schrieben. Die Abschlussarbeit wird auf Python Sprache programmiert. Zweitens ist Das Frontend (Frontend oder Client-Teil der Anwendung) im Browser des Benutzers auszuführen. Die Webanwendung könnte theoretisch nur aus dem Client-Teil bestehen, wenn Benutzerdaten nicht länger als eine Sitzung gespeichert werden müssen. Dies aber ist nicht der Fall der Abschlussarbeit, da die Studentenkarten und der Verlauf des Verleihablaufs mindestens für ein laufenden Semester gespeichert werden muss, damit die Mitarbeiter des PSE-Labor immer eine Zugang zu allen gespeicherten vorherigen Leihvorgangs von der Ausleihe bis zur Rückgabe eines Boards. Es ist vorgesehen, dass am Ende des Semester nach dem letzte Rückgabe eines Boards die Datensätzen des zu Ende gegangen Semesters gelöscht wird.

Da in den letzten Jahren sich Webanwendungen rasant weiterentwickelt und die Desktop-Lösungen schrittweise ersetzt haben und sind zu einem wesentlichen Bestandteil des Geschäfts in der modernen Welt geworden haben, es sollte auch nicht unerwähnt bleiben, welche Vorteile die Webanwendungen haben:

- **Zugriff von jedem Gerät** Die Webanwendung kann überall auf der Welt von einem Computer, Tablet oder Smartphone zugegriffen und verwendet werden. Notwendig ist, dass dem Gerät eine Internetverbindung zur Verfügung steht.

- **die Kostenersparnis** Webanwendungen können auf allen Plattformen ausgeführt werden und müssen nicht mehr separat für Android und iOS entwickelt werden.

- **Anpassungsfähigkeit** Wenn native Anwendungen bestimmte Betriebssysteme erfordern, jedoch können jedes Betriebssystem (Windows, MAC, Linux usw.) und jeder Browser (Internet Explorer, Opera, FireFox, Google Chrome usw.) für die Arbeit mit einer Webanwendung.) verwendet werden.

- **Keine Software zum Herunterladen** Günstig und einfach dem Endnutzer zu liefern, zu warten und zu aktualisieren. Das Aktualisieren auf die neueste Version erfolgt beim nächsten Laden der Webseite.

- **Netzwerksicherheit** Das Websystem verfügt über einen einzigen Einstiegspunkt, der zentral geschützt und konfiguriert werden kann.

- **Skalierbarkeit** Mit zunehmender Belastung des Systems ist es nicht erforderlich, die Leistung des Computer von Endbenutzer zu erhöhen. Mit einer Webanwendung kann in der Regel nur mithilfe von Hardwareressourcen eine größere Datenmenge verarbeitet werden, ohne den Quellcode neu zu schreiben und die Architektur ändern zu müssen.

- **Verhinderung von Datenverlust** Benutzerdaten werden in der "Cloud" gespeichert, für deren Integrität die Hosting-Anbieter verantwortlich sind, deswegen vom Verlust geschützt, falls die Festplatte des Computers beschädigt wird.

Als nächstes geht die Frage nach, wie Client und Server miteinander kommunizieren können. Der Client spricht mit dem Server über das HTTP-Protokoll. Das HTTP-Protokoll setzt die Verwendung einer Client-Server-Datenübertragungsstruktur voraus. Die Clientanwendung generiert eine Anforderung und sendet sie an den Server. Anschließend verarbeitet die Serversoftware diese Anforderung, generiert eine Antwort und sendet sie an den Client zurück. Die Clientanwendung kann dann weiterhin andere Anforderungen senden, die auf ähnliche Weise behandelt werden. Die Anforderung kann mit den GET-Methoden erfolgen, wenn wir Daten empfangen möchten, und mit POST, wenn wir die Daten ändern möchten. Die Anfrage enthält auch den Host (Site-Domain), den Anfragetext (wenn es sich um eine POST-Anfrage handelt) und viele zusätzliche technische Informationen. Das HTTP-Protokoll ist ein zustandsloses Protokoll auf Anwendungsebene und bietet keine Speicherung von Informationen über die Sitzung des Benutzers; jede Datenübertragung wird als neue Sitzung betrachtet.

Am Rande sei auch eine weitere Technologie erwähnt, die uns ständig begegnet. Cache (Cache) ist ein Konzept in der Entwicklung, bei dem häufig verwendete Daten, anstatt jedes Mal aus der Datenbank abgerufen, berechnet oder auf andere Weise vorbereitet, an einem schnell zugänglichen Ort gespeichert werden. Einige Beispiele für die Verwendung des Caches:

- Django erhielt eine Anfrage, Daten für ein Diagramm in einem Bericht abzurufen. Daten aus der Datenbank werden abgeholt, vorbereitet und abgelegt in einer Datenbank mit schnellem Zugriff z.B. "memcached", die beispielsweise für eine Stunde zwischengespeichert wird. Bei der nächsten Anfrage werden die notwendigen Daten sofort von "memcached" erhalten und an das Frontend gesendet. Wenn es festgestellt wird, dass die Daten nicht mehr relevant sind, werden sie ungültig beziehungsweise gelöscht aus dem Cache.

- CDN-Anbieter (Content Delivery Network) werden zum Zwischenspeichern statischer Dateien verwendet. Hierbei handelt es sich um Servers auf der ganzen Welt, die für die Bereitstellung statischer Inhalte optimiert sind. Manchmal ist es effizienter, Bilder, Videos und JS-Skripte auf einem CDN anstatt auf Ihrem eigenen Server abzulegen.

- In allen Browsern ist das statische Zwischenspeichern von Dateien standardmäßig aktiviert. Da die Website nicht zum ersten Mal geöffnet wird, werden die dazugehörige Daten aus dem Zwischenspeichern viel schneller geladen. Der Nachteil für den Entwickler ist, dass bei aktiviertem Cache die vorgenommenen Änderungen nicht immer sofort sichtbar sind.

Daraus lässt sich die Schlussfolgerung ziehen, dass eine Webanwendung für einen Endnutzer wie eine Website aussieht, auf der die Webseiten mit teilweise oder vollständig nicht formatiertem Inhalt sich befinden. Die Endfertigung des Inhalts findet nur dann statt, nachdem ein Website-Besucher die Seite vom Webserver angefordert hat.

In diesem Kapitel wurden die Grundlagen der Webanwendungen und des HTTP-Protokolls erörtert. Diese Diskussion war nicht als umfassende Beschreibung aller Funktionen gedacht, eher als Überblick über das Verständnis und die Arbeit mit aktuellen Stand der Technologie und zukünftigen Verwendung der entwickelte Software im PSE-Labor.

2.2 Raspberry Pi Board und OS

3 Systemdesign

4 Implementation

4.1 Register-Client

Der Standard legt das physikalische Layout der mikroBus-Pinout-Verbindung, die verwendeten Kommunikations- und Stromversorgungspins auf dem Mainboard fest.

4.2 Server

Die *Cloud* oder *Cloud Computing* Begriff kommt offensichtlich aus dem Englischen und heißt auf Deutsch "Wolke". Der Begriff beschreibt einen oder mehrere entfernte Server, auf die man seine Daten von einem Gerät über das Internet hochladen kann. Dann übernimmt die Cloud die Aufgaben wie die Datenverarbeitung oder komplizierte Programmabläufe. Während der Datenverarbeitung weißt der Nutzer nicht, wie viele Server hinter der Cloud stecken und welche komplizierte Hardware für die Berechnungen benötigt werden.

4.3 Display-Client

Die *Cloud* oder *Cloud Computing* Begriff kommt offensichtlich aus dem Englischen und heißt auf Deutsch "Wolke". Der Begriff beschreibt einen oder mehrere entfernte Server, auf die man seine Daten von einem Gerät über das Internet hochladen kann. Dann übernimmt die Cloud die Aufgaben wie die Datenverarbeitung oder komplizierte Programmabläufe. Während der Datenverarbeitung weißt der Nutzer nicht, wie viele Server hinter der Cloud stecken und welche komplizierte Hardware für die Berechnungen benötigt werden.

5 Application

5.1 Usage in laboratory

Der Standard legt das physikalische Layout der mikroBus-Pinout-Verbindung, die verwendeten Kommunikations- und Stromversorgungspins auf dem Mainboard fest. Der Zweck von mikroBUS ist es, eine einfache Erweiterbarkeit der Hardware mit einer großen Anzahl von standardisierten kompakten Zusatzboards zu ermöglichen, von denen jede einen einzelnen Sensor, Display, Encoder oder Motortreiber, eine integrierte Schaltung hat. Der von MikroElektronika entwickelte mikroBUS ist ein offener Standard - jeder kann mikroBUS in seinem Hardwaredesign implementieren. Die Abbildung[1] **??** zeigt die Pinout Spezifikation des Herstellers, die man entsprechend ändern kann und die neue Verbindungen für den eigenen Projekt feststellen. Wenn ein Modul eine Schnittstelle verwendet, die bereits auf mikroBUS vorhanden ist, benutzt man diese exakten Pins und markiert diese entsprechend. Wenn ein Pin nicht verwendet wird, sollte er als NC (für "Not Connected") markiert sein.

[1] https://download.mikroe.com/documents/standards/mikrobus/mikrobus-standard-specification-v200.pdf

6 Results and conclusion

Appendix

A

AJAX

AJAX (asynchrones Javascript und XML) ist der allgemeine Name für Technologien, mit denen asynchrone Anforderungen (ohne erneutes Laden von Seiten) an den Server gestellt und Daten ausgetauscht werden können. Da die Client- und Serverteile der Webanwendung in verschiedenen Programmiersprachen geschrieben sind, müssen zum Austausch von Informationen die Datenstrukturen (z. B. Listen und Wörterbücher), in denen sie gespeichert sind, in das JSON-Format konvertiert werden.

DOM

DOM (Document Object Model) ist die Struktur einer HTML-Seite. Bei der Arbeit mit dem DOM werden HTML-Tags (Elemente auf einer Seite) gefunden, hinzugefügt, geändert, verschoben und entfernt.

HTML-Vorlage

Eine HTML-Vorlage ist eine intelligente HTML-Seite, die Variablen anstelle bestimmter Werte verwendet und verschiedene Operatoren bereitstellt: if (if-then), for-Schleife (Durchlaufen einer Liste) und andere. Das Abrufen einer HTML-Seite aus einer Vorlage durch Ersetzen von Variablen und Anwenden von Operatoren wird als Vorlagenrendering bezeichnet. Die resultierende Seite wird dem Benutzer angezeigt. Falls einen anderen Abschnitt zu öffnen ist, muss ein anderen Musters geladen werden. Wenn andere Daten in der Vorlage verwendet werden müssen, werden sie vom Server angefordert. Alle Formularübermittlungen mit Daten sind auch AJAX-Anforderungen an den Server.

HTTP

HTTP steht für HyperText Transfer Protocol, Hypertext Transfer Protocol". HTTP ist ein weit verbreitetes Datenübertragungsprotokoll, das ursprünglich für die Übertragung von Hypertextdokumenten vorgesehen war (Dokumente, die möglicherweise Links enthalten, mit denen Sie den Übergang zu anderen Dokumenten organisieren können). Die Basis dieses Protokolls ist eine Anforderung von einem Client (Browser) an einen Server und eine Serverantwort an einen Client.

JSON

JSON (JavaScript Object Notation) ist ein universelles Format für den Datenaustausch zwischen einem Client und einem Server. Es ist eine einfache Zeichenfolge, die in jeder Programmiersprache verwendet werden kann.

PORT

Dies ist ein optionaler Teil der URL, der die Portnummer angibt [**shklar:webapplication**], die der Zielwebserver abhört. Die Standardportnummer für HTTP-Server ist 80, einige Konfigurationen sind jedoch so eingerichtet, dass sie eine alternative Portnummer verwenden. In diesem Fall muss diese Nummer in der URL angegeben werden. Die Portnummer wird direkt mit einem Doppelpunkt, der unmittelbar auf den Servernamen oder die Adresse folgt, eingegeben.

RFID

RFID (englisch radio-frequency identification oder „Identifizierung mit Hilfe elektromagnetischer Wellen“) bezeichnet eine Technologie für Sender-Empfänger-Systeme und wird bei der Abschlussarbeit verwendet, um die vorhandenen zur Ausleihe Raspberry Pi Boards zu markieren und identifizieren.

URI

Uniform Resource Identifier ist ein Pfad zu einer bestimmten Ressource (z.B. einem Dokument), für die eine Operation ausgeführt werden muss (z. B. bei Verwendung der GET-Methode bedeutet dies das Abrufen einer Ressource). Einige Anforderungen

beziehen sich möglicherweise nicht auf eine Ressource und in diesem Fall kann der Startzeile anstelle des URI ein Sternchen (Symbol "*") hinzugefügt werden.