Diplomarbeit

Entomophagie

Untertitel der Arbeit

Imst, 22. März 2018

Eingereicht von

Leonid Hammer	Verantwortlich für IT: HTML, CSS, BWL: Kaufvertrag
Kevin Glatz	Verantwortlich für IT: SQL, C BWL: Kostenrechnung, Einsatz von Arduino
Florian Tipotsch	Verantwortlich für IT: SQL, C, Website BWL: Nutzen von Websites

Eingereicht bei

Stefan Stolz und Nina Margreiter

Eidesstattliche Erklärung

Ich erkläre an Eides statt, dass ich die vorliegende Diplomarbeit selbst verfasst und keine anderen als die angeführten Behelfe verbwendet habe. Alle Stellen, die wörtlich oder inhaltlich den angegebenen Quellen entnommen wurden, sind als solche kenntlich gemacht. Ich bin damit einverstanden, dass meine Arbeit öffentlich zugänglich gemacht wird.

_____

Ort, Datum

_____ _____

Leonid Hammer Kevin Glatz

_____

Florian Tipotsch

Abnahmeerklärung

Hiermit bestätigt der Auftraggeber, dass das übergebene Produkt dieser Diplomarbeit den dokumentierten Vorgaben entspricht. Des Weiteren verzichtet der Auftraggeber auf unentgeltliche Wartung und Weiterentwicklung des Produktes durch die Projektmitglieder bzw. die Schule.

_____

Ort, Datum

_____

Thorsten Schwerte

Vorwort

Wir wollen uns bei unseren Betreuern unserer Schule und bei unserem Projektpartner für die Unterstützung und die Zurverfügungstellung von Hilfsmittel, mit welcher wir diese Projekt abschließen konnten, bedanken.

Abstract (Deutsch)

Unser Projekt Entomophagie handelt von dem Züchten und Verzehr von Insekten. Dafür entwickeln wir einen Brutkasten Prototyp, welcher in Zukunft als Basis für weitere Brutkästen dient. Das Ziel unseres Produktes ist eine umweltfreundlichere Nahrungsquelle zu erschließen, die in der Westlichen Welt fast komplett ungenutzt ist. Mithilfe des Brutkasten soll es für jeden möglich sein, Essen günstig und selber 'anzubauen'. Insekten sind aufgrund ihres niedrigen Energieverbrauches während der Züchtung viel umweltfreundlicher als beispielsweise Kühe.

Abstract (Englisch)

Our project Entomophagie is about growing and eating insects at home. To complete our task, we create a prototype of a breeding hub, which could be used for future adaptations of our model. The goal of our product is to show the western world the environmental friendly food source of insects, which is almost unused there. Furthermore we want to make it available for everyone to create their own food at home by simply breeding insects and eating them. The main reason insects are that more environmental friendly than, for example, cows is the energy consumption while breeding them.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Quelltexte

Notationen

Beschreibung wie Code, Hinweise, Zitate etc. formatiert werden

1 Einleitung

Das Projekt entstand als der biologische Bereich der Universität Innsbruck das Thema Entomophagie behandelte. Entomophagie beschreibt den menschlichen Konsum von Insekten, eine Tätigkeit die im asiatischen Raum gängig ist. Das Ziel von diesem Projekt ist es aufzuzeigen wie simpel und ressourcenschonend die Zucht von Insekten in einem künstlichen automatisierten Lebensraum ist.

Vorteile

Massentierhaltung und riesige Monokulturen zerstören das natürliche Ökosystem der Erde, das Problem wird nicht vereinfacht indem die gesamte Bevölkerung jährlich steigt. Insekten sind nicht nur eine nahrhafte Möglichkeit diese Nachfrage zu erfüllen, sondern sind auch weniger schadhaft für den Planeten.

Ein weiterer Vorteil besteht auch darin das jeder Insekten bei sich daheim züchten kann und dabei auch vergleichsweise kostengünstig bei der Anschaffung bleibt.

2 Projektmanagement

2.1 Metainformationen

2.1.1 Team

Unser Team bestand ursprünglich aus 4 Personen. Da ein Schüler sich ungeplant vom Team entfernt hat, daher umfasst das Projektteam 3 Schüler. Projektleiter Leonid Hammer sowie die Gruppenmitglieder Florian Tipotsch und Kevin Glatz.

2.1.2 Betreuer

Die Lehrpersonen Stefan Stolz, MSc und Mag. Nina Margreiter erklärten sich bereit diese Projekt zu betreuen. Herr Professor Stolz betreute den technischen Teil der Arbeit und Frau Professor Margreiter den betriebswirtschaftlichen Bereich.

2.1.3 Partner

Der Projektpartner ist die technische Universität Innsbruck. Das Projekt umfangt die Erstellung eines Prototypen für einen automatisierten Insektenbrutkasten.

2.1.4 Ansprechpartner

Die Ansprechpartner zu diesem Projekt waren selbstverständlich unsere Projektbetreuer sowie ao. Univ.-Prof. Thorsten Schwerte.

Projektziele-Hierarchie - SMART

2.1.5 Projektumfeld

- Identifikation der Stakeholder
- Charakterisierung der Stakeholder
- Maßnahmen
- Grafische Darstellung des Umfeldes

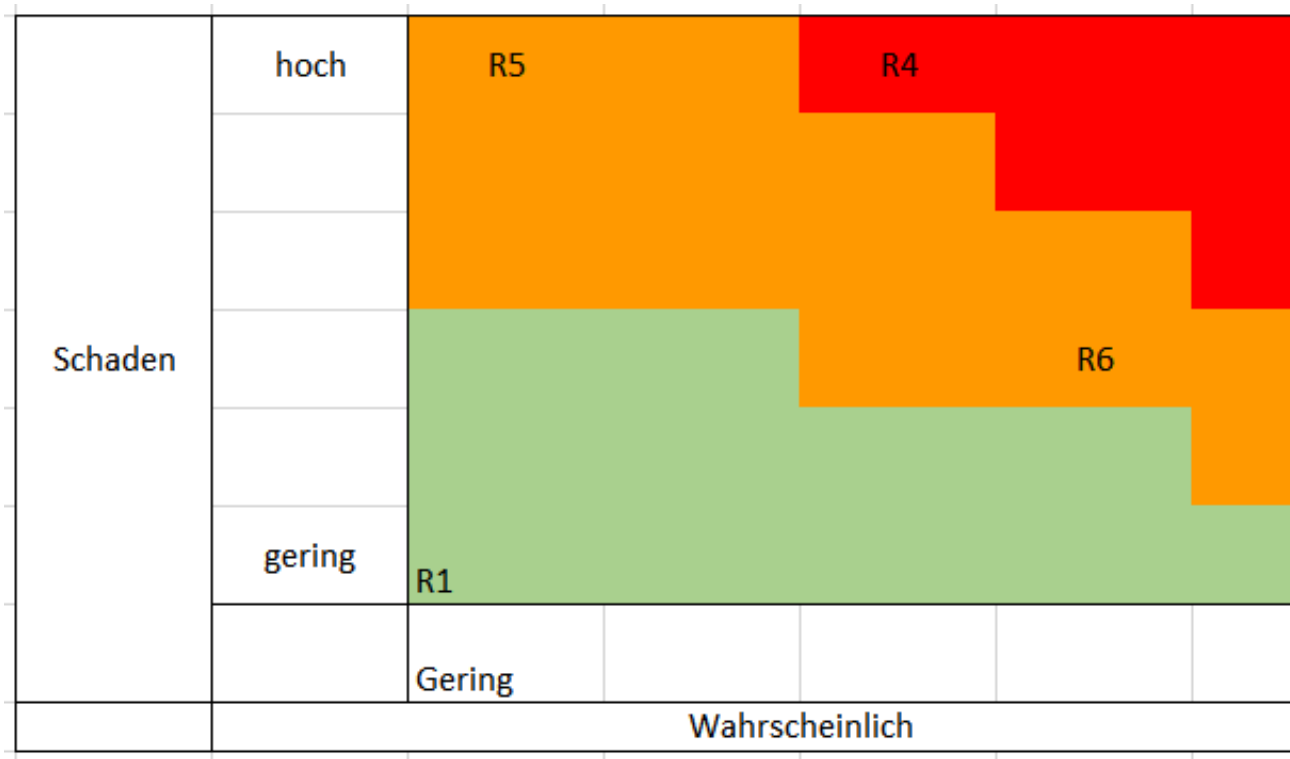


Abbildung 2.1: Risikomatrix von unserem Projekt

2.1.6 Risikoanalyse

Die Risikomatrix ist ein einfaches Tool die Wahrscheinlichkeit und Verheerung von verschiedenen Risiken einfach darzustellen. Je nach Seriosität wird einem Bereich die Farbe rot, gelb oder grün zugewiesen. Schwere und Wahrscheinliche Risiken sollten vermieden oder vermindert werden.

- Risikomatrix

2.2 Pflichtenheft

2.2.1 Zielbestimmung

- Projektbeschreibung
- IST-Zustand
- SOLL-Zustand
- NICHT-Ziele (Abgrenzungskriterien)

2.2.2 Produkteinsatz und Umgebung

- Anwendungsgebiet
- Zielgruppen
- Betriebsbedingungen
- Hard-/Softwareumgebung

2.2.3 Funktionalitäten

- MUSS-Anforderungen
  - Funktional
  - Nicht-funktional
- KANN-Anforderungen
  - Funktional
  - Nicht-funktional

2.2.4 Testszenarien und Testfälle

- Beschreibung der Testmethodik
- Testfall 1
- Testfall 2
- ...

2.2.5 Liefervereinbarung

- Lieferumfang
- Modus
- Verteilung(Deployment)

»»»> 3c0b844d31a00a1c748c7c305ec8b2786a3b01f7

2.3 Planung

2.3.1 Projektstrukturplan

Tabelle 2.1: Projektstrukturplan

P 1	Projektstart	
1	Planung	Alle
1.1	Besprechungstermine setzen	Alle
1.1.1	Vertrag erstellen	Alle
1.2	Verantwortungsmatrix	Alle
1.3	Budgetplan erstellen	Alle
2	Programmierung (Kevin + Flo)	Kevin Glatz, Florian Tiptsch
2.1	Programmiersprache	Kevin Glatz
2.1.1	Sprache Wählen Š	Kevin Glatz
2.1.2	Informationsammlung für Features	Kevin Glatz
2.1.3	Vertraut machen	Kevin Glatz
2.2	Arduino UNO kaufen (Flo)	Florian Tipotsch
2.2.1	Vertraut machen	Kevin Glatz
2.2.2	Test Programme	Kevin Glatz
2.3	Programmierung des Brutkasten	Kevin Glatz
3	Brutkasten (Leo)	Leonid Hammer
3.1	Equipment Liste erstellen	Leonid Hammer
3.1.1	Equipment überprüfen	Leonid Hammer
3.1.2	Equipment Kaufen	Alle
3.2	Konzeptzeichnung	Leonid Hammer
3.2.1	Zeichnen	Leonid Hammer
3.2.2	Bauen	Leonid Hammer
4	Datenbank	Kevin Glatz
4.1	ER-Diagramm	Florian Tipotsch + Kevin Glatz
4.2	Sprache Wählen	Florian Tipotsch
4.3	HTML Seite mit Daten	Florian Tipotsch
5	Nachforschung (Leo)	Leonid Hammer
5.1	Welches Tier	Leonid Hammer
5.2	Wie viele Tiere	Leonid Hammer
5.3	Futter	Leonid Hammer
6	Webapp	Florian Tipotsch
6.1	Prototyp	Florian Tipotsch
6.2	Login am Server	Florian Tipotsch
6.3	Registrierung Zuchkammer	Florian Tipotsch

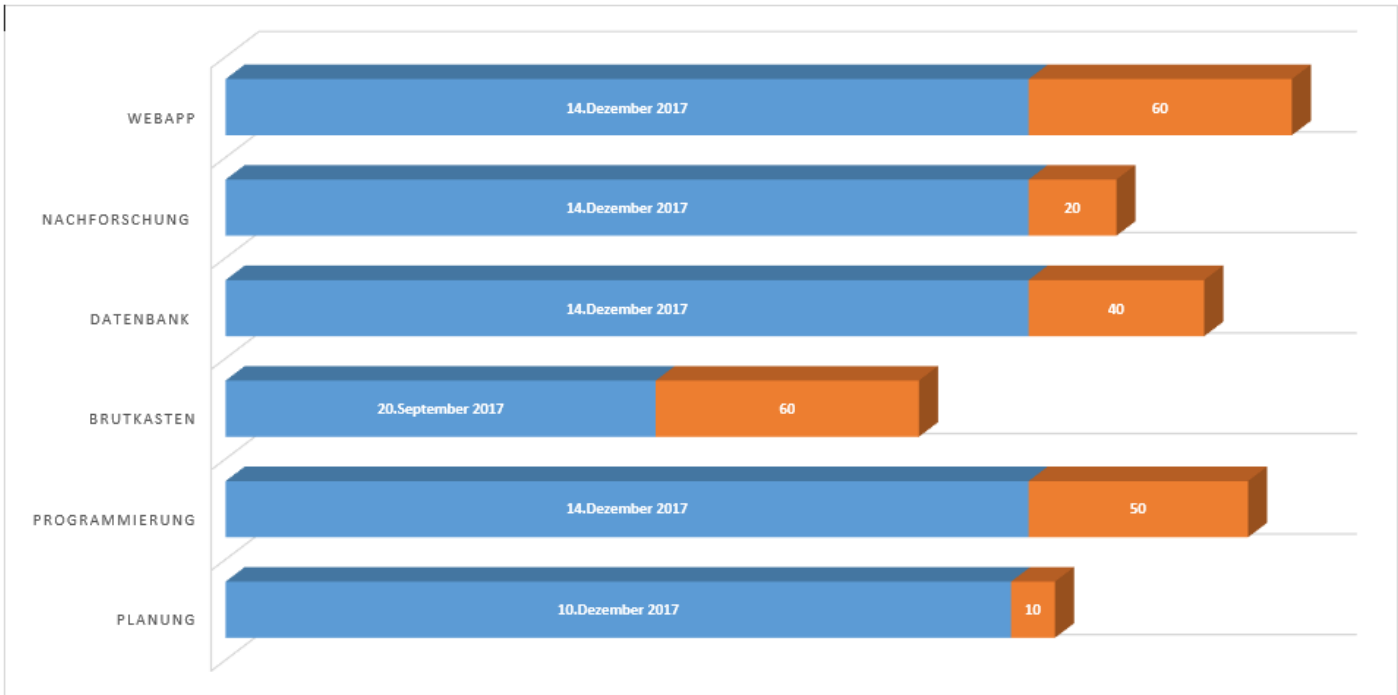


Abbildung 2.2: Unser Gant Chart für das Projekt

2.3.2 Meilensteine

Tabelle 2.2: Meilensteine

1	Planung	Alle
2	Programmierung	Kevin Glatz, Florian Tiptsch
3	Brutkasten	Leonid Hammer
4	Datenbank	Kevin Glatz
5	Nachforschung	Leonid Hammer
6	Webapp	Florian Tipotsch

2.3.3 Gant-Chart

2.3.4 Abnahmekriterien

Die Abnahmekriterien wurden von unserem Auftraggeber so gesetzt, dass es einen Prototypen geben soll, welcher Daten auslesen kann und darauf reagiert.

2.3.5 Pläne zur Evaluierung

Zuerst wollen wird die Daten auf unserem Arduino auslesen und diese dann mit Hilfe eines Wlan Moduls an unsere Website senden.

3 Brutkasten

Hier werden wir uns dem Aufbau, der Planung und Ausführung des Brutkastens widmen.

3.1 Equipment Liste

- Technisches Equipment
  - Arduino
  -
  -
  -
  -
  -
-
-
-
-
-

3.2 Überprüfen des Equipments

3.3 Equipment Kaufen

3.4 Konzeptzeichnung

3.5 Zeichnen

3.6 Bauen

4 Eingesetzte Technologien

- Kurzbeschreibung aller Technologien, die verwendet wurden.
- Technologien die aus dem Unterricht bekannt sind, nur nennen und deren Einsatzzweck im Projekt beschreiben, nicht die Technologien selbst.
- Technologien die aus dem Unterricht nicht bekannt sind, im Detail beschreiben incl. deren Einsatz im Projekt
- Fokus aus eingesetzten Frameworks

4.1 Technologie für Webapp

- PHP - Für Webapp
- Html - Für Webapp
- MySql - Für Datenbanken
- Yii2 - Für Webapp
- MVC - Model, View, Controller - Für Webapp
- CRUD - Create, Read, Update, Delete - Für Webapp
- REST - Für Datenbank, Webapp und Datenspeicherung

4.1.1 HTML - Hypertext Markup Language

HTML sind die Grundlagen bzw. die Grundstrukturen für jede Website und werden von Browsern dargestellt. HTML wird von dem World Wide Web Consortium (W3C) (3) und dem Web Hypertext Application Technology Working Group (WHATWG) (2) weiterentwickelt.

Gute Ressourcen zum lernen und schreiben von HTML findet man auf w3schools.com (1)

4.1.2 Was ist Yii

Yii ist ein high Performance PHP Framework, welches vor allem für die Entwicklung im Web 2.0 eingesetzt wird. Web 2.0 fördert das User aktiv im Web mitmachen. Diese können eigenen Beiträge erstellen und diese auf der Website anzeigen lassen. Mehr dazu im Kapitel Yii2 (16)

Alternative zu Frameworks

Yii kann sehr weitreichend eingesetzt werden. Mit dem richtigen Wissen und den richtigen Fähigkeiten kann man alles, was mit einer PHP Seite möglich ist, in Yii2 umsetzten.

Allerdings sind Frameworks nicht verwaltungsfreundlich, da sie sehr viel Vorwissen erfordern, um diese richtig zu implementieren und zu warten. Einfacher zu implementieren sind CMS Systeme. Es gibt sehr viele große CMS Systeme wie zum Beispiel:

- Joomla
- Wordpress
- Drupal
- Contao

Diese haben wir auch schon im Unterricht kennengelernt und damit Websites erstellt. Vorteile sind vor allem die einfache Installation und rasche Einrichtung einer Website. Auch SEO wird von den CMS Systemen vereinfacht. Nachteile sind oft eingeschränkte Möglichkeiten und Grenzen welche das CMS setzt.

Warum haben wir uns für Yii2 entschieden

Der Hauptgrund warum wir uns gegen CMS Systeme entschieden haben, sind die eingeschränkten Möglichkeiten die wir damit hätten. Bei Yii2 können wir die gesamte Website nach unseren Bedarf zusammenstellen und auch so bearbeiten wie wir es wollen. Es war uns auch wichtig, dass wir nach modernen Entwurfsmustern arbeiten. Bei Yii2 wird das MVC - Model, View, Controller Muster eingesetzt.

Wir hätten uns auch für andere Frameworks entscheiden können, allerdings war

uns Yii2 schon bekannt und wir haben damit schon einige Websites erstellt.

Alternativen für Yii2 sind:

- PureMVC
- Laravel

4.1.3 PureMVC

PureMVC ist seit dem Release in 2008 unverändert. Das hat den Vorteil, dass der administrative Aufwand aufgrund nicht vorhandener Updates sehr gering ist. Außerdem muss man das Framework nur einmal lernen und kann dieses dann ohne irgendwelche Änderungen zu befürchten meistern. Es gibt auch Best-Practicse Beispiele in vielen verschiedenen Sprachen. Siehe Website: (10)

4.1.4 Laravel

Laravel: PHP That Doesn't Hurt. Code Happy and Enjoy The Fresh Air. Laravel will PHP einfach und übersichtlich Programmierbar machen dazu verwendet es auch das MVC Muster und auch den PHPComposer um sehr einfach neue Erweiterungen zu installieren. Auch bei Laravel gibt es sehr gute Documentationen. Siehe Website: (**?**)

4.2 Technologien für die Datenauslesung

- Gas Sensor - Für Luftqualität
- Wärmesensor - Für Raumtemperatur
- Luftfeuchtigkeit - Für Luftqualität
- Schalter - Für Futtermenge
- kinetischer Motor - für geregelte Luftzufuhr
- WLAN - Für Datenübertragung

4.2.1 Gas Sensoren

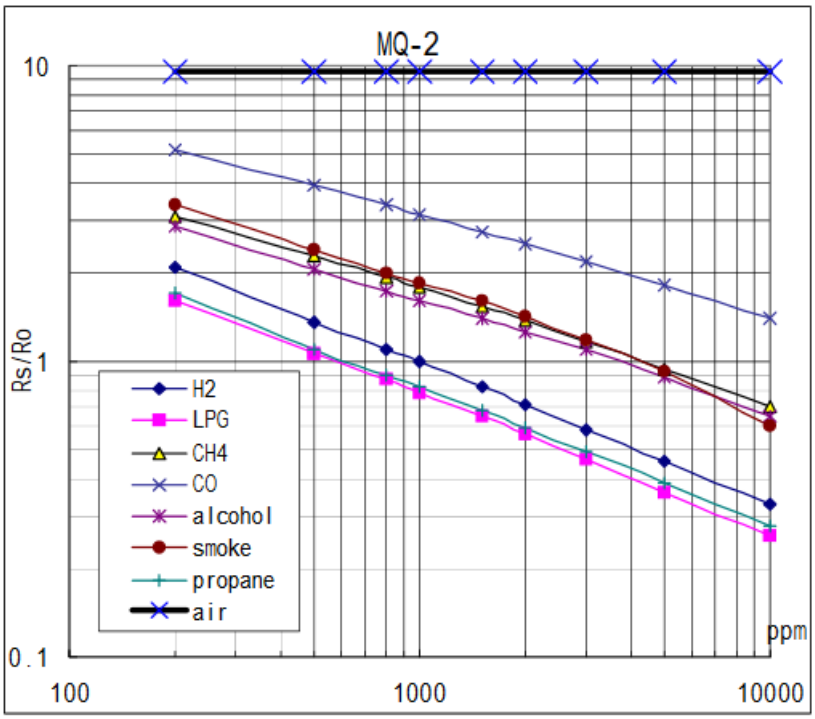
Für die Auswertung der Gaswerte stehen uns mehrere Module zur Verfügung. Zum einen stand uns die vielfältige MQ-Serie zur Verfügung oder der spezialisierter CCS811 von Adafruit.

In der Schule haben wir nicht nur den MQ2 Sensor zur Verfügung bereitgestellt bekommen, sondern auch den Adafruit CCS811. Wir bedanken uns dafür vielmals.

MQ Gas Sensoren

- MQ2 Methane, Butane, LPG, smoke
- MQ3 Alcohol, Ethanol, smoke
- MQ4 Methane, CNG Gas
- MQ5 Natural gas, LPG
- MQ6 LPG, butane gas

- MQ7 Carbon Monoxide
- MQ8 Hydrogen Gas
- MQ9 Carbon Monoxide, flammable gasses
- Mehr gibt es auf der Website: (5)



(7)

(5)

Im Datasheet (7) kann man herauslesen das der Sensor MQ2 (5) H2, LPG, CH4, CO, Alkohol, Rauch und Propan in einem Bereich von 200 bis 10000 Parts per million (Anteil pro Million) messen kann. Wie empfindlich der Sensor ist, hängt von den RS und RO werten ab.

- RS: Sensor Widerstand bei verschiedenen Konzentrationen von Gas
- RO: Sensor Widerstand bei 1000ppm von H2 bei sauberer Luft.

Da der MQ-2 auf so viele Gase außer Co2 reagiert haben wir uns entschlos-

sen eine Alternative zu suchen. Glücklicherweise hat uns die Schule auch den CCS811 von Adafruit zur Verfügung stellen können,

Adafruit CCS811

In unserem Projekt haben wir uns letztendlich für den CCS811 von Adafruit entschieden. Im Gegensatz zu dem MQ-Sensoren erlaubt der CCS811 eCO2 in einem Bereich von 400 bis 8192 ppm (parts per million) auszulesen. (9)

Dadurch können wir uns komplizierte mathematische Formelrechnungen sparen und direkt den gewünschten Datentypen verwenden.

4.2.2 Wärmesensor

DHT11

In dem Arduino Uno Set war der DHT 11 Wärme und Temperatur Sensor von Adafruit direkt mitgeliefert. Dieses Modul erlaubt es uns digitale Daten in einem zwei Sekunden Takt auszulesen. Es liest die Wärme in einem Bereich von 0 bis 50°C, die Daten können allerdings um etwa 2°C variieren.

CCS811

Der CCS811 besitzt auch einen Thermistor, mit welchem wir direkt die Raumtemperatur auslesen können. Da der DHT11 allerdings auch die Luftfeuchtigkeit anzeigen kann, verwendeten wir diesen anstatt den CCS811. (9)

4.2.3 Luftfeuchtigkeit

Da der DHT direkt bei dem Arduino Kit mit dabei war, gab es keinen Grund einen anderen Sensor zu suchen. Das Modul erlaubt uns die Luftfeuchtigkeit in einem Bereich von 20 bis 80 Prozent zu messen. Diese Daten können allerdings um beinahe 5% variieren.

4.2.4 kinetischer Motor

Für eine automatisierte Luftzufuhr verwenden wir den Servo-Mikrocontroller SG90. Sowie der DHT 11 war dieser Motor schon im Arduino Uno Set beigelegt. Der 3-polige Servo wird für die Luftzufuhr verwendet. Der Propeller am drehenden Motor öffnet und schließt die Luftklappe des Kastens. Das ermöglicht eine höhere Kontrolle und Regulierung des CO2 Wertes. (https://servodatabase.com/servo/towerpro

4.2.5 Hebel

Um zu sehen wie viel Futter für die Insekten vorhanden ist, verwendeten wir einen Joystick. Dieser Joystick war beim Arduino Uno Set mitgeliefert und kann die X- und Y-Achse ablesen sowie auf Druck zu reagieren. Anhand der Y-Achse kann man den Joystick als Hebel verwenden um den momentanen Futterstand abzulesen.

4.2.6 Datenübertragung

Damit der Nutzer die Daten letztendlich auf der Website abrufen kann, verwenden wir das WLAN-Modul ESP8266. Dieser sehr beliebter und billiger Mikrocontroller fungiert als eine voll Funktionsfähige Netzwerkschnittstelle.

Es erlaubt uns auch den Brutkasten kabellos und benutzerfreundlicher zu Gestalten

5 Problemanalyse

5.1 USE-Case-Analyse

- UseCases auf Basis von Benutzerzielen identifizieren:
  - Benutzer eines Systems identifizieren
  - Benutzerziele identifizieren (Interviews)
  - Use-Case-Liste pro Benutzer definieren
- UseCases auf Basis von Ereignissen identifizieren:
  - Externes Event triggert einen Prozess
  - zeitliches Event triggert einen Prozess (Zeitpunkt wird erreicht)
  - State-Event (Zustandsänderung im System triggert einen Prozess)
- Werkzeuge:
  - USE-Case-Beschreibungen (textuell, tabellarisch)
  - USE-Case-Diagramm
  - Aktivitätsdiagramm für den Use-Case (Interaktion zwischen Akteur und System abbilden)
  - System-Sequenzdiagramm (Spezialfall eines Sequenzdiagramms: Nur 1 Akteur und 1 Objekt, das Objekt ist das komplette System, es geht um die Input/Output Requirements, die abzubilden sind)

5.2 Domain-Class-Modelling

- "Dinge"(Rollen, Einheiten, Geräte, Events etc.) identifizieren, um die es im Projekt geht
- ER-Modellierung oder Klassendiagramme
- Zustandsdiagramme (zur Darstellung des Lebenszyklus von Domain-Klassen darstellen)

5.3 User-Interface-Design

- Mockups
- Wireframes

6 Systementwurf

6.1 Architektur

6.1.1 Design der Komponenten

Unsere Webapp ist in Yii2 programmiert und ist deshalb nach dem MVC Muster aufgebaut.

6.1.2 MVC - Model, View, Controller

Was ist MVC?

MVC, auch Model, View, Controller ist ein modernes Entwurfsmuster, welches meist für Anwendungen, die ein User Interface beinhalten, eingesetzt wird. Zum Beispiel für PHP, JAVA, C# und Ruby Anwendungen. (15)

Vor- und Nachteile

Vorteile:	Nachteile:
Gleichzeitiges Programmieren	Schlechte Übersicht
Hohe Kohäsion (13)	Konsistente Programmierung notwendig
Lose Kopplung (14)	Steile Lernkurve

Das MVC Entwurfsmuster ist sehr weit verbreitet und hoch angesehen. Es wird deshalb in den meisten Anwendungen mit User Interface angewandt. Der Grund für die Verwendung dieses Musters sind die vielen Vorteile die es bietet. Außerdem können viele Nachteile über Frameworks behoben werden.

Aufbau von MVC

Im Grunde arbeitet MVC mit 3 Klassen:

- Modell
- View
- Controller

Die verschiedenen Klassen haben unterschiedliche Aufgaben.

View

Die View Klasse ist nur zum Anzeigen der einzelnen Teile des User Interfaces verantwortlich. Bei Webseiten zum Beispiel die Index Seite oder das Login Formular. Außerdem nimmt es alle Benutzer/innen Eingaben entgegen.

Controller

Die Controller Klasse steuert die ganze Anwendung. Sie nimmt die User eingaben von der View Klasse entgegen und verarbeitet diese. Außerdem ändert sie auch die View Klasse um andere Seiten anzuzeigen. Sie nimmt auch die Daten von der Modell Klasse entgegen, verarbeitet diese und gibt sie an die View Klasse weiter.

Modell

Die Modell Klasse enthält die darzustellenden Daten. Sie ist unabhängig von der View und der Controller Klasse

6.1.3 CRUD - Create, Read, Update, Delete

Was ist CRUD?

CRUD sind die 4 grundlegenden Aufgaben einer Datenbankanbindung:

- Create - Erstellen neuer Datensätze
- Read - Auslesen der Datensätze
- Update - Aktualisieren von vorhandenen Datensätze
- Delete - Löschen von Datensätzen

(12)
CRUD ist für alle Datenbankzugriffe verantwortlich die getätigt werden.

6.1.4 Yii2

Was ist Yii

YII ist ein high Performance PHP Framework welches vor allem für die Entwicklung im Web2.0 eingesetzt wird. Web 2.0 fördert das User aktiv im Web mitmachen. Diese können eigene Beiträge erstellen und diese auf der Website anzeigen lassen.(16)

Gii

Gii ist der Yii eigene Model, Crud, Controller, Form, Module und Extension Generator. Mit Gii kann man sehr einfach eine Model Klasse mit einer unterliegenden Datenbanktabelle erstellen. Aus dieser Model Klasse kann man dann wiederum CRUD Befehle erzeugen. Mit Gii kann man die gesamte Grund MVC Struktur für Yii2 erzeugen und im weiterm Verlauf dann nach eigenen Wünschen verändern.

Vor- und Nachteile

Yii hat sehr viele Vorteile, allerdings auch einige Nachteile:
Vorteile sind:

- CRUD-Creator (Gii)
- Model Generator (Gii)
- Einfache Implementierung von HTML Formulare
- Einfach Datenbankzugriffe

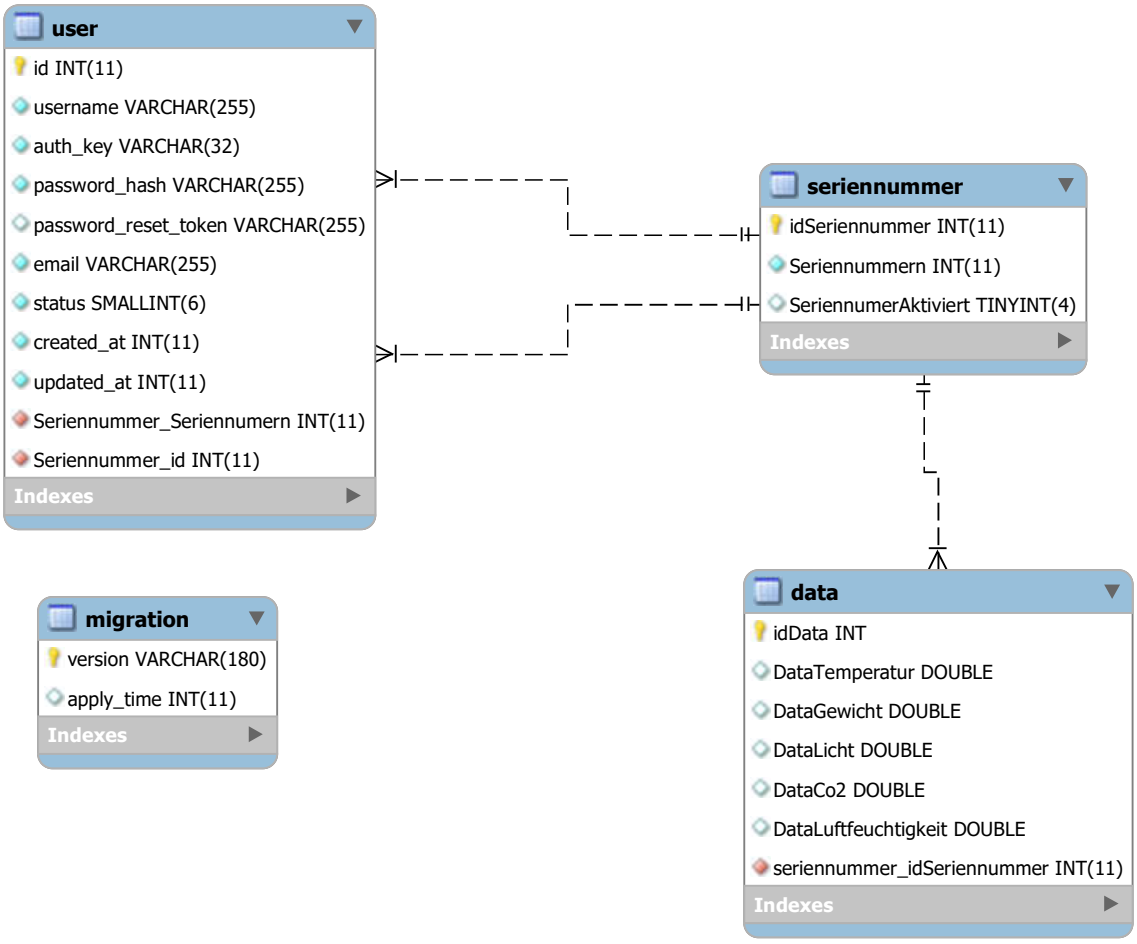
Nachteile ist der Hohe Setup-Aufwand und die benötigte Kenntnisse in PHP und SQL, welche allerdings bei fast allen Frameworks von Nöten sind.

6.1.5 Benutzerschnittstellen

Die Benutzerschnittstelle unseres Brutkasten besteht hauptsächlich über die Webapp. Dort kann man sich anmelden und die Daten für den eigenen Brutkasten auslesen. Um sicherzugehen, dass man nur die eigenen Daten sieht, muss man sich auf der Website mit der Seriennummer, die man beim Kauf eines Kasten erhält, registrieren. Dadurch kann man dann auf die Daten zugreifen, welche für die jeweilige Seriennummer gespeichert sind.
Wenn man noch nicht registriert wird kann man auf der Website einen Link zu unserem Webshop sehen, den wir allerdings nicht in diesem Projekt erstellen werden da es den Rahmen dieses Projektes Sprengen würde.

6.1.6 Datenhaltunskonzept

Hier ist unser ER-Diagramm


user
id INT(11)
username VARCHAR(255)
auth_key VARCHAR(32)
password_hash VARCHAR(255)
password_reset_token VARCHAR(255)
email VARCHAR(255)
status SMALLINT(6)
created_at INT(11)
updated_at INT(11)
Seriennummer_Seriennumern INT(11)
Seriennummer_id INT(11)
Indexes
seriennummer
idSeriennummer INT(11)
Seriennummern INT(11)
SeriennumerAktiviert TINYINT(4)
Indexes
migration
version VARCHAR(180)
apply_time INT(11)
Indexes
data
idData INT
DataTemperatur DOUBLE
DataGewicht DOUBLE
DataLicht DOUBLE
DataCo2 DOUBLE
DataLuftfeuchtigkeit DOUBLE
seriennummer_idSeriennummer INT(11)
Indexes

Die Datenbank werden wir bei unserem Web Hoster anlegen und dort alle Daten bis zu 7 Tage lang speichern. Daten sollen alle 5-10 Minuten von unserem Brutkasten übermittelt werden. Zur Übermittlung werden wir ein WLAN Modul an unserem Brutkasten anbringen. Mehr dazu im Kapitel Datenübermittlung. Für Prototyp Zwecken werden wir die Daten vorerst lokal über PHP-MyAdmin speichern.

6.1.7 Konzept für Ausnahmebehandlung

- Systemweite Festlegung, wie mit Exceptions umgegangen wird
- Exceptions sind primär aus den Bereichen UI, Persistenz, Workflow-Management

6.1.8 Sicherheitskonzept

Auf unsere Website kann man sich mit Hilfe von Username und Password anmelden. Im weiteren Verlauf des Projektes besteht noch die Möglichkeit für unsere Rest-Schnittelle eine Authentifizierung einzubauen. Da unser Prototyp allerdings nur lokal vorhanden ist, ist dies zum jetzigen Zeitpunk noch nicht notwendig.

6.1.9 Design der Testumgebung

Unser Protoyp wird getestet indem wir ihn einen Tag autonom laufen lassen und am Ende dieser 24 Stunden Periode alle Daten kontrollieren, ob diese Sinn machen und der Realität entsprechen. Um dies zu gewährleisten, bräuchten wir allerdings einen klimaregulierten Raum der CO2 messen kann.

6.1.10 Desing der Ausführungsumgebung

Das Endprodukt sollte im Freien autonom funktionieren können. Das Setup für den Endbenutzer/in besteht aus dem Anstecken an eine Stromversorgungquelle und dem Verbinden mit dem Internet, sowie das Einsetzen der Insekten. Im weiteren Verlauf der Brutzeit muss der Benutzer/die Benutzerin die Insekten füttern. Unser Produkt übernimmt das Regulieren von Temperatur sowie CO2 Werte, damit die Insekten überleben können.

6.2 Detailentwurf

USE-Cases Klassendiagramme vom Domain-Klassendiagramm ableiten (incl. detaillierter Darstellung und Verwendung von Vererbungshierarchichen, abstrakten Klassen, Interfaces)

- Sequenzdiagramme vom System-Sequenz-Diagramm ableiten
- Aktivitätsdiagramme
- Detaillierte Zustandsdiagramme für wichtige Klassen

Verwendung von CRC-Cards (Class, Responsibilities, Collaboration) für die Klassen

- um Verantwortlichkeiten und Zusammenarbeit zwischen Klassen zu definieren und
- um auf den Entwurf der Geschäftslogik zu fokussieren

Design-Klassen für jeden einzelnen USE-Case können z.B. sein:

- UI-Klassen
- Data-Access-Klassen
- Entity-Klassen (Domain-Klassen)
- Controller-Klassen
- Business-Logik-Klassen
- View-Klassen

Optimierung des Entwurfs (Modularisierung, Erweiterbarkeit, Lesbarkeit):

- Kopplung optimieren
- Kohäsion optimieren
- SOLID
- Entwurfsmuster einsetzen

6.3 Arduino

Die erfolgreiche Verwendung des Arduino besteht aus zwei Punkten:

- Verkablung der einzelnen Module
- Programmierung

Wir können vor allem dem Programmcode um einiges vereinfachen indem wir diesen in drei Unterbereiche aufteilen, wobei jedes Abteil eine fest zugeteilte Aufgabe bekommt. Dasselbe zählt für die Verkablung der einzelnen Module, da diese mit einer strukturierten Farbkodierung übersichtlicher werden.

6.3.1 Verkablung

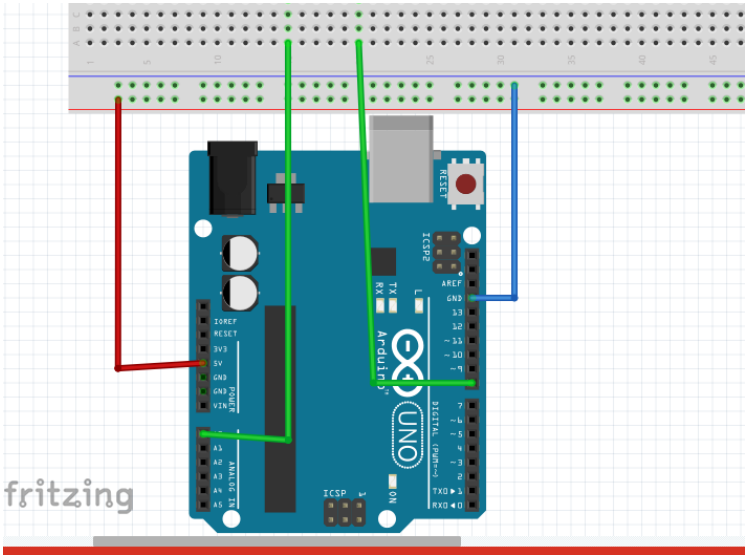


Abbildung 6.2: Arduino mit Breadboard

In Abbildung 6.2 können wir nun den Arduino und das Breadboard sehen. Für eine einfachere Unterscheidung der Kabel verwenden wir ein simples Farbschema. Rot steht für die Stromzufuhr, grün für die Datenauslesung und blau für die Erdung.

Die untersten und obersten zwei Zeilen, lassen den Strom bzw. die Erdung horizontal laufen. Die Steckplätze darüber leiten hingegen jegliche Information vertikal. Ein wichtiges Detail ist das man Stromzufuhr auf der horizontalen Plus reihe platziert (in der Grafik anhand der roten Linie zu sehen), da ansonsten keine Energie ankommt. Bei den vertikalen Steckplätzen gibt es allerdings keine negativ oder positiv gepolten Spalten.

Worauf man bei der Datenauslesung achten muss, ist das Arduino zwei Möglichkeiten dafür verwendet. Zum einen gibt es digitale Daten, welche einen Wert von 1 oder 0 besitzten können. Die Alternative dafür sind analoge Daten, die einen Wert von 0 bis 1023 Wahrnehmen können.

Vereinfacht erklärt werden bei digitalen Pins entweder 0 oder 5 Volt gelesen und bei analogen Pins 5 Volt durch 1024 Werte geteilt. Je nachdem wie viel Stromzufuhr das Modul hat, kann es somit einen Datenwert um einiges genauer zurückgeben (4)

Bibliotheken importieren und globale Variablen definieren

In diesem Schritt importieren wir die nötigen Bibliotheken welche aus GitHub oder direkt von Arduino zur verfügung gestellt bekommen. Auch werden alle Variablen die sowohl in den Methoden Setup und Loop verwendet werden erstellt.

Setup

Die Funktion Setup wird verwendet um einmalige Konfigurationen zu tätigen. Dazu zählen unter anderem Aufbau einer Verbindung, Kalibrierung sensibler Controller oder ähnlichen. Die hier getätigten Aktivitäten werden nur ausgeführt wenn der Arduino startet bzw. zurückgesetzt wird.

Loop

Diese Methode verarbeitet und gibt all unsere Werte aus. Besonders an der Klasse Loop ist, dass diese dauerhaft wiederholt wird. Es gibt keine maximale Durchlaufmenge und die Geschwindigkeit eines Durchlaufs wird von der Methode d̈elayïn Millisekunden definiert. Mit Hilfe dieser Eigenschaften, können wir zeitnahe alle Daten auslesen. Falls nun ein Wert eine Mindestgrenze kann sofort darauf reagiert werden.

Eine genauere Beschreibung der einzelnen Module und deren Programmierung erfolgt in den kommenden Seiten.

7 Implementierung

Detaillierte Beschreibung der Implementierung aller Teilkomponenten der Software entlang der zentralsten Use-Cases:

- GUI-Implementierung
- Controllerlogik
- Geschäftslogik
- Datenbankzugriffe

Detaillierte Beschreibung der Teststrategie (Testdriven Development):

- UNIT-Tests (Funktional)
- Integrationstests

Zu Codesequenzen:

- kurze Codesequenzen direkt im Text (mit Zeilnnummern auf die man in der Beschreibung verweisen kann)
- lange Codesequenzen in den Anhang (mit Zeilennummer) und darauf verweisen (wie z.B. hier)

7.1 Webapp

Für unser Projekt erstellen wir eine Webapp mit der man die Daten seiner eigenen Zuchtkammer anzeigen lassen kann. Wir haben geplant, dass man sich mit der Seriennummer des Brutkastens registrieren kann und dann am Handy oder Browser über eine Webapp alle Daten anzeigen lassen kann. Folgende Daten sollte man auslesen können:

- Sauerstoff
- Luftfeuchtigkeit
- Gewicht
- Temperatur
- Futtermenge
- ungefähre Zeit bis zu Reife

Als Grundlage für die Website haben wir das Framework Yii2 verwendet. Mehr dazu im Kapitel Yii2.

7.1.1 Mockups

Hier sieht man die Ansicht, wenn man auf unserer Website angemeldet ist. Man kann auf die Daten der Zuchtkammer zugreifen indem man auf den Button 'Meine Zuchtkammer' klickt.

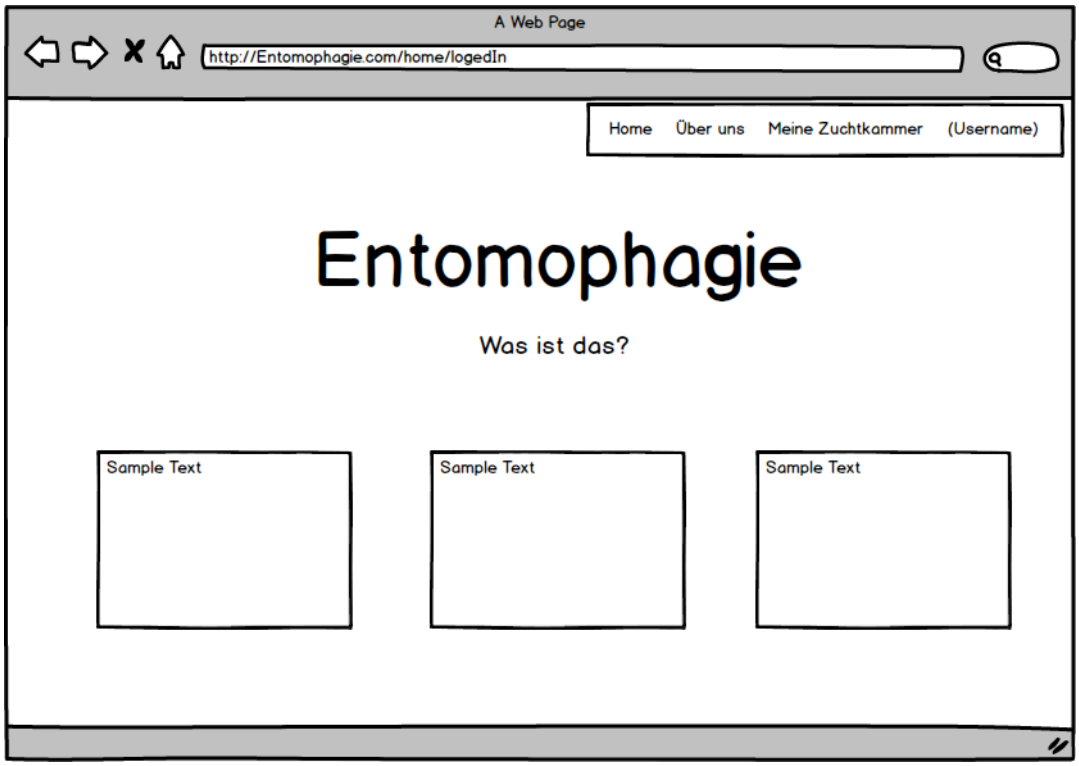


Abbildung 7.1: Mockup unserer Seite wenn man eingeloggt ist

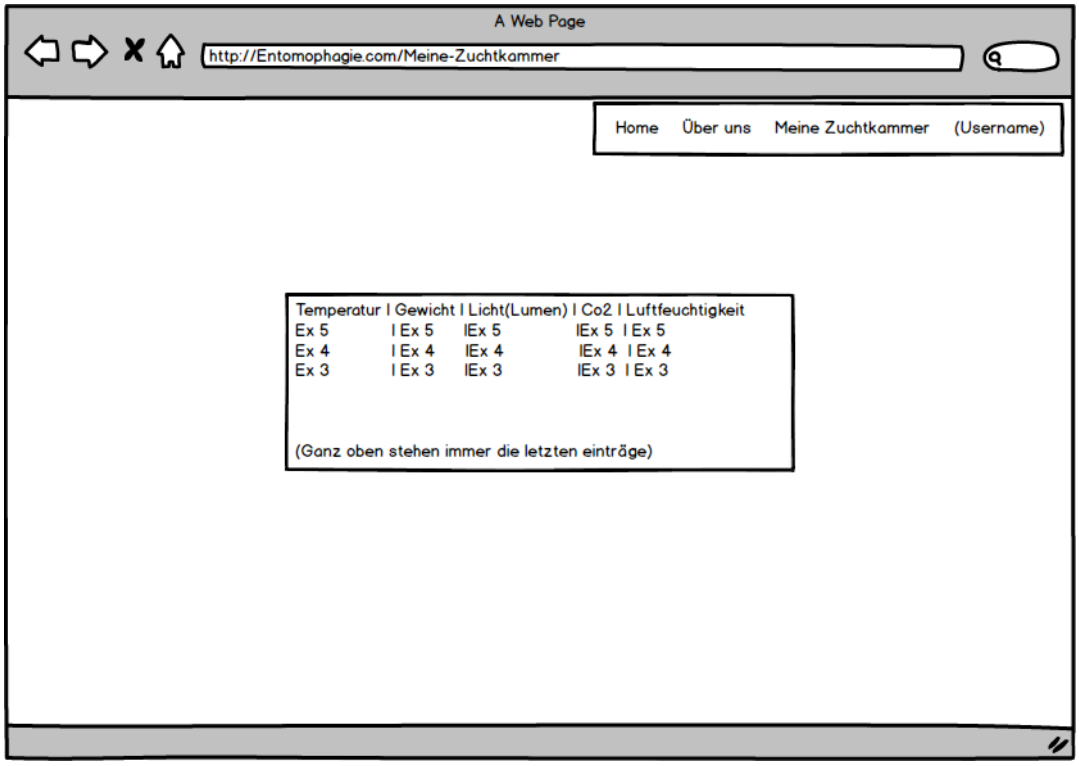


Abbildung 7.2: Mockup der Seite Meine-Zuchtkammer

Auf der Seite 'Meine Zuchtkammer' sieht man die letzten Daten, die die Zuchtkammer wiedergegeben hat. Diese sind in absteigender Reihenfolge geordnet, was bedeutet, dass der letzte Eintrag ganz oben steht.

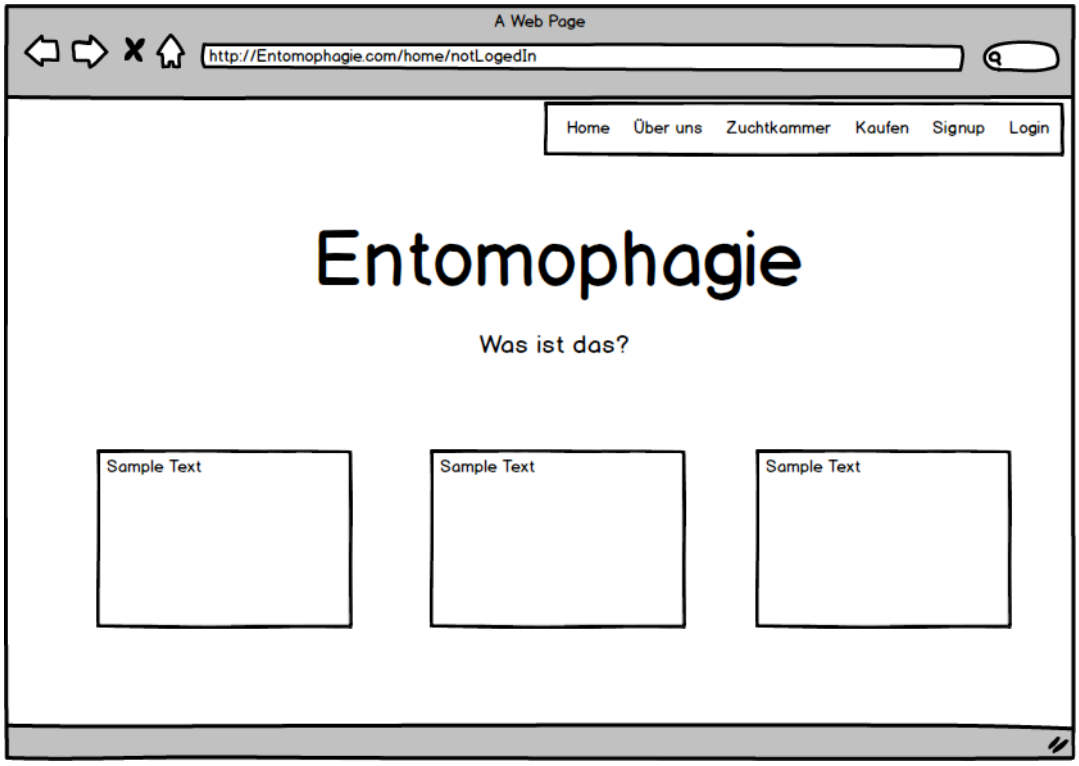


Abbildung 7.3: Mockup unserer Seite wenn man nicht eingeloggt ist

Hier sieht man die Ansicht, wenn man auf unserer Website nicht angemeldet ist. Man kann über den Button 'Zuchtkammer' mehr über unsere Zuchtkammern herausfinden. Weiters kann man über den Button 'Kaufen' seine eigene Zuchtkammer erwerben. Über den Button 'Signup' kann man sich als neuer Benutzer registrieren. Um sich registrieren zu können braucht man allerdings zuerst eine Seriennummer, die man beim Kauf einer Zuchtkammer erhält.

7.1.2 Datenbankzugriffe

Unserer Datenbanzugriffe werden von unserem Framework verarbeite. Dabei verwendet das Framework CRUD Befehle und arbeitet nach dem MVC Muster. Das heißt es gibt ein unterliegendes Modell welches Daten an unseren Controller mitgibt, welcher vom View angezeigt werden.

Der gesamte Datenbankzugriff kann mittels Yii2 sehr einfach erstellt werden. Mehr dazu siehe Gii.

7.1.3 Datenübermittlung

Um die Daten von unserem Arduino an unsere Datenbank zu senden werden wir in den Brutkasten ein WLAN Modul einbauen und die Daten alle 5-10 Minuten als JSON-Format an unsere REST-Schnittstelle unserer Webapp senden. Hierfür haben wir das WLAN Modul ESP8266 verwendet. Das hat den Vorteil, dass unsere Daten direkt von unserem Brutkasten an die Datenbank gegeben werden können.
Für diese Lösung brauchen wir eine REST-Schnittelle in unserer Webapp. Dies kann über Yii2 sehr einfach realisiert werden. Yii2 hat eine vor implementierte REST-Schnittelle, welche man nur noch aktivieren muss. Die Daten werden aus der Modell Klasse ausgelesen und über den Controller an die REST-View weitergeleitet. Die ganze Implementierung sind 5-10 Zeilen Code. Die Code Sequenz sehen Sie im Kapitel Rest Schnittelle.

7.1.4 Rest Schnittelle

Hier sieht man die Ordnerstruktur. Wir erstellen im obersten Verzeichnis einen neuen Ordner namens 'api' um im Web über folgende

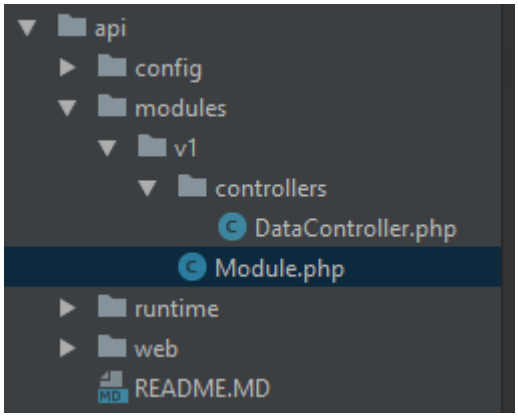


Abbildung 7.4: Abbildung des API Ordners

URL:'www.entomophagie/api/web/v1/datas' auf unsere REST-Schnittelle zugreifen können. Dabei müssen wir darauf achten, dass wir Groß- und Kleinschreibung beachten. Falls nicht, könnte es passieren, dass wir keine Anzeige bekommen. Um dieses Problem zu lösen, ist es am einfachsten alle Ordner, in unserem Verzeichnis immer klein zu schreiben.
Weiters Kopieren wir 'web', 'config' und den 'runtime' Ordner aus unserem Front- bzw. Backend Ordner.

Quelltext 7.1: Controller Klasse für REST

```
<?php

namespace api\modules\v1\controllers;
use yii\rest\ActiveController;

class DataController extends Active Controller
{
   public $modelClass = 'common\models\Data';
}
```

Hier sieht man die Controller Klasse. In der Controller Klasse müssen wir den Yii eigenen ActiveController einbinden und die Modell Klasse festlegen, welche die Daten ausliest.

Quelltext 7.2: Module Klasse für REST

```php
<?php

namespace api\modules\v1;

class Module extends \yii\base\Module
{
   public $controllerNamespace =
   'api\modules\v1\controllers';

   public FUnction init()
   {
      parent::init();
   }
}
```

Hier sieht man die Module Klasse. In der Module Klasse wird unser neuer Rest Controller initialisiert, damit unsere Anwendung diesen verwenden kann. Dort müssen wir unseren Controller, den wir zuvor erstellt, haben festlegen.
Wir haben den Rest Controller mit folgendem Online Tutorial erstellt (8).

7.2 Verkablung

Die tatsächliche Verkablung der einzelnen Module war am Anfang recht komplex. Trotz des richtigen Setups unterliefen Fehler die verhinderten das Daten sinnvoll ausgelesen werden konnten. Verwendung von Widerständen führte zu falschen oder unrealistischen Daten, da der Strom der darüber lief zu gering war bzw. der Widerstand zu hoch.

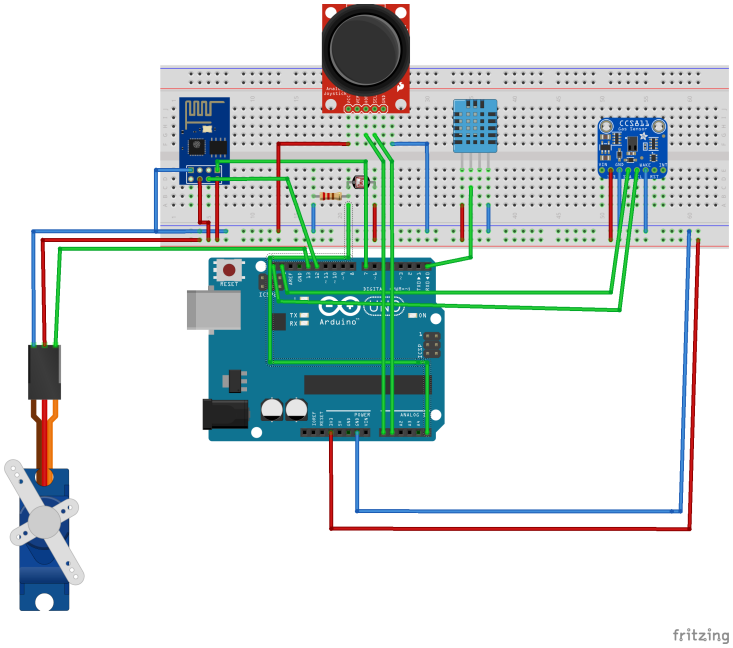


Abbildung 7.5: Verkablung aller Module

In dieser Grafik kann man die Verkablung aller Module sehen. Abgebildet von links anfangend:

- Servomotor SG0
- WLAN Modul ESP8266
- Joystick Modul (oben)
- Lichtempfindlichkeit (unten)
- Temperatur und Luftfeuchtigkeitssensor DHT11

- CO2 Sensor CCS811

Die größte Herausforderung bot der CCS811 sowie der ESP8266, da beide keine typische Verkabelung verwenden. Wie man an der Grafik erkennen kann werden mehrere Strom oder Datenquellen benötigt.

7.2.1 CCS811

=======

7.3 Arduino

Die Verkablung der einzelnen Module ist aus verschiedenen Gründen wichtig. Zum einen brauch jedes Modul genügend Strom um seine Aufgabe zu erfüllen, gleichzeitig würde es aber ohne die Erdung zu einem Kurzschluss kommen. Des weiteren ist es wichtig das die Module Daten auslesen und weitersenden können mit stromfluss einfügen) »»»> 3e06afbabfa374741d0d74fa5cbde2bc3cc60033

Der CCS811 benötigt weder digitale noch analoge Werte da er die Pins von der I2C Uhr und Datenlinien verwendet. Des weiteren ist es wichtig anzumerken das dieser Sensor nur mit Mikrocontroller funktioniert die intern I2C "clock strechingünterstützen. In unserem Projekt mussten wir nur die Steckplätze SCL und SDA für den CO2 Sensor verwenden und es funktionierte.

7.3.1 ESP8266

Da der ESP als eingestehende Netzwerkschnittelle zählt braucht diese besonders viel Energie. Daher muss man beim Verkabeln vor allem darauf achten

dass beide Stromanschlüsse richtig plaziert wurden. Ähnlich wie das CO2 Messmodul verwendet der WLAN Adapter 2 Datenpins. Diese sind allerdings beide digital und können an beliebigen stellen zwischen 0 und 13 gesetzt werden

7.3.2 Joystick Modul

Der Joystick verwendet ursprünglich zwei analoge Pins und einen Analogen. Damit benötigt dieser Mikrocontroller die meisten Pins. Damit eine variable X und Y Achse ausgelesen werden kann benötigen wir die zwei analogen Pins. Der digitale Wert fungiert als Knopfdruck, dieser kann in unserem Projekt allerdings ignoriert werden. Auf die X Ache können wir hingegen nicht verzichten, da beide Werte benötigt werden um die Position auslesen zu können.

«««< HEAD

7.4 Programmierung

=======

Bibliotheken importieren und globale Variablen definieren

itHub oder direkt von Arduino zur verfügung gestellt bekommen. Auch werden alle Variablen die sowohl in den Methoden Setup und Loop verwendet werden erstellt. äten werden nur ausgeführt wenn der Arduino startet bzw. zurückgesetzt wird. en definiert. Mit Hilfe dieser Eigenschaften, können wir zeitnahe alle Daten auslesen. Falls nun ein Wert eine Mindestgrenze kann sofort darauf reagiert werden. kt kann der Quellcode in drei Bereiche eingeteilt werden. »»»> 3e06afbabfa374741d0d74fa5cbde2bc3cc60033

7.4.1 DHT11 & CCS811

Für den DHT 11 sowie dem CCS811 gibt es eine von Adafruit frei verwendbare Bibliothek. Mithilfe von dieser kann man ein Objekt erstellen und es mithilfe diesem Objektes die Momentane Temperatur/Luftfeuchtigkeit auslesen.

Der Parameter #include fügt externe Klassen in unser Projekt ein und erlaubt es uns in das Objekt dht sowie ccs zu erstellen. Des weitere können wir mit dem Parameter #define relevante Daten festlegen. Ein solches Beispiel ist von welchem Pin Daten empfangen werden.

Quelltext 7.3: Datenauslesung CCS811

```
1
2  delay(2000);
3  if(ccs.available())
4  {
5     if(!ccs.readData())
6     {
7        Serial.print("CO2: ");
8        Serial.println(ccs.geteCO2());
9     }
10    else
11    {
12       Serial.println("ERROR!");
13       //while(1);
14    }
15 }
```

(9)

Die Loop Funktion, wiederholt sich im vorgegebenen Rhythmus immer wieder und gibt keine Variablen zurück. Dort werden alle Daten ausgelesen und an

das WLAN Modul weitergeschickt bzw. in diesem Fall wird es auf den Serial Monitor ausgegeben.

Die WENN abfrage prüft als erstes ob der CO2 Sensor kalibriert wurde oder nicht. Falls es gelungen sein sollte gibt es die Werte mit dem Befehl Serial.println(ccs.geteCO2()); im Serial Monitor aus.

7.4.2 Hebel

Das Joystick bzw. der Hebel konnte direkt abgelesen werden. Wichtig hierbei ist es das man nicht nur eine Achse im Setup verwendet sondern beide, ansonsten kommt es bei dem Loop zu einem Fehler und der Wert der benötigten Achse verändert sich nicht.

7.4.3 SG90

Die vom verwendeten Bibliotheken waren bei der Installierung der Arduino eigenen IDE direkt dabei. Die Servo kontrolliert die Luftzufuhr und muss daher Werte vom CCS811 wiederverwenden

Quelltext 7.4: Automatisierten Servobewegung bei zu niedrigen CO2 Werten

```
1
2 if(ccs.geteCO2() <= co2Min)
3 {
4    //move the micro servo from 0 degrees to 180
         degrees
5    for(; servoAngle < 180; servoAngle++)
6    {
7       servo.write(servoAngle);
8       delay(10);
```

```
9     }
10 }
11 if (ccs.geteCO2() > co2Min && servoAngle != 0)
12 {
13     servo.write(45);
14     servoAngle = 0;
15     Serial.println("RETURN");
16 }
```

(6)

Die If Abfrage prüft ob CO2 einen Mindestwert (co2Min) unterschreitet. Falls das passiert wird eine Schleife ausgeführt die den Servo 180° dreht. Diese 180° würde die Lüftungsklappe aufhalten. Falls der CO2 Wert wieder in einem akzeptablen Bereich liegt und die Position des Motors nicht null beträgt, wird die zweite Schleife aktiviert die den Motor zur Position 0 zurückbringt

7.4.4 Wlan

d g Für die Implementierung der Wlan-Anbindung haben wir ein ESP8266 verwendet. Das Modul ist sehr einfach einzurichten. Das Ganze haben wir mit den Beispielen von den ESP8266 in der Arduino IDE gemacht.

ESP8266

Das Wlan Modul ESP866 wird verwendet um die Verbindung mit dem Internet aufzubauen und die Daten mit Hilfe der REST-Schnittelle an unsere Website senden. Dafür haben wir die ESP8266 Library von Arduino verwendet.(**?**)

Quelltext 7.5: Verbindungsaufbau des ESP8266

```
1  #include <ESP8266.h>
2
3  const char* ssid = "SSID des Benutzers";
4  const char* password = "password des Benutzers";
5
6  const char* host = "Unsere Website";
7
8  void setup()
9  {
10    Serial.begin(115200);
11    dela(10);
12
13    Serial.println();
14    Serial.println();
15    Serial.println("Connecting to ");
16    Serial.println(ssid);
17
18
19    WiFi.mode(WIFI_STA);
```

```
20      WiFi.begin(ssid,password);
21  }
```

Hier sieht man den Verbindungsaufbau des ESP8266. In der Zeile 20 wird dann die Verbindung gestartet. Vorher übergeben wir die SSID und das Password des WLANs, womit das Modul sich verbinden soll.

Quelltext 7.6: Verbindungsaufbau Response

```
while(WiFi.status() != WL_CONNECTED)
{
   delay(500);
   Serial.print(".");
}

Serial.println("");
Serial.println("WiFi connected");
Serial.println("IP adress: ");
Serial.println(WiFi.localIP());
```

Bei erfolgreicher Verbindung wird die IP Adresse unseres Moduls ausgegeben.

Quelltext 7.7: Ausführung der REST Befehle

```
client.print(String("GET ) + url + " HTTP/1.1\r\n)" +
"Host: " + host + "\r\n" +
"Connection: close\r\n\r\n");
unsigned long timeout = millis();
while(millis() - timeout > 5000)
{
   Serial.println(">>> Client Timeout !");
   client.stop();
   return;
}
```

Anschließend können wir in unserer Loop REST Befehle wie Get, Post, Put, etc ausführen.

8 Deployment

- Umsetzung der Ausführungsumgebung
- Deployment
- DevOps-Thema

9 Tests

9.1 Systemtests

Systemtests aller implementierten Funktionalitäten lt. Pflichtenheft

- Beschreibung der Teststrategie
- Testfall 1
- Testfall 2
- Tesfall 3
- ...

9.2 Akzeptanztests

10 Projektevaluation

siehe Projektmanagement-Unterricht

11 Benutzerhandbuch

falls im Projekt gefordert

12 Betriebswirtschaftlicher Kontext

12.0.1 Arduino in der Wirtschaft

Mikrocontroller gewinnen vor allem beim Endverbraucher immer mehr an Qualität, allerdings wird auch im Landwirtschaftlichen Bereich von diesen Technologien Gebrauch gemacht. In dem Artikel Ärduino Projekte in der Landwirtschaft und Fischzuchtẅird beschrieben wie Privatpersonen ihre eigene Lösung zu Problemen wie Fischzucht, Lenksysteme für Erntemaschinen und weitere Fälle gestalten. Arduino Mikrocontroller werden im wirtschaftlichen Bereich nicht genutzt, da es ein Werkzeug zur Prototyp Erstellung ist. Die einzelnen Module werden hingegen auch im professionellen Raum verwendet. Der wirtschaftliche Vorteil von Mikrocontroller besteht nicht nur in der Größe, die es für Beinahe jeden Bereich verwendbar machen, sondern auch der geringe Preis und die Automatisierung, welche mit dieser Technik ermöglicht wird. Man kann damit nicht nur potentiell Personal sparen, sondern auch permanent Werte einlesen und überwachen. Das erlaubt es uns in der Medizin oder Meerestierforschung schneller einzugreifen falls die Daten einen Mindestwert unterschreiten ten Kosten unseres Projekt beträgt wie folgend beschrieben. Die tatsächlichen Ausgaben sind allerdings um einiges geringer, da wir ein Teil der Module schon besetzt habe oder von der Schule zur verfügung gestellt bekommen haben.

12.0.2 Kostenrechnung

Die Kostenrechnung ist ein mächtiges Werkzeug die mehrere wichtige Aufgabenbereiche deckt. Dazu zählen unter anderem Grundlagen der Preisbildung, Verwendung als Entscheidungsinstrument oder auch als Planungsinstrument. Grundlage für die gängigen Kostenrechnungsarten besteht aus der Istkostenrechnung, diese listet alle tatsächlich angefallenen Kosten eines Produktes oder einer Periode auf. (11)

Fichten Leimholzplatte	€ 8,97
Plexiglas	€ 5,49€
Arduino Uno	€ 25,94
Lochrasterleiterplatte	€ 2,65
40-Piece Jumper Wire	€ 6,49
Adafruit CCS811 Air Quality Sensor	€ 22,98
DHT11 Digital Humidity Temperature Sensor	€ 4,99
ESP8266 WLAN/WiFi Module	€ 6,29
Servo Micro 9g SG90	€ 11,70
Joystick Dual Axis Module	€ 3,99
Raspberry Pi 896 8860 3 Model B	€ 31,95
SanDisk Ultra 16 GB SDHC	€ 10,99
Summe	€ 142,44

12.1 Nutzen einer Website

Es ist für Unternehmer wichtiger eine Website zu besitzen, weil für viele Personen das Internet der erste Ansprechpartner, um Informationen zu erhalten, ist. Dabei ist es sinnvoll, dass man die gewünschte Website schnell und einfach findet. Dafür ist eine gute search engine optimation (SEO) wichtig, damit die Website in Google auf der Ersten Seite angezeigt wird, da die zweite Ergebnis Seite von Google oft ignoriert wird.

Auch in unserem Projekt haben wir eine Website erstellt. Diese ist allerdings eine reine Funktionswebsite. Auf unserer Website kann man vorerst nur Daten seines Brutkastens auslesen. Natürlich könnte die Website noch mit einem Webshop erweitert werden, da dies allerdings den Rahmen des Projektes sprengen würde, haben wir dies nicht gemacht.

Auf Websiten ist es auch wichtig, dass das Unternehmen widergespiegelt wird. Man sollte ein innovatives Corporate Design, falls eines vorhanden ist, anwenden. Denn in der heutigen modernen Zeit ist es für viele der Erste Eindruck vom Unternehmen.

13 Zusammenfassung

- Etwas längere Form des Abstracts
- Detaillierte Beschreibung des Outputs der Arbeit

Literaturverzeichnis

[1] W3schools. URL: https://www.w3schools.com/.

[2] Web hypertext application technology working group. URL: https://whatwg.org/.

[3] World wide web consortium. URL: https://www.w3.org/.

[4] Arduino. analogread(). URL: https://www.arduino.cc/reference/en/language/functions/analog-io/analogread/.

[5] Arduino. Mqgassensor. URL: https://playground.arduino.cc/Main/MQGasSensors.

[6] Calin Dragos. Tutorial: How to control the tower pro sg90 servo with arduino uno, October 2. URL: https://www.intorobotics.com/tutorial-how-to-control-the-tower-pro-sg90-servo-with-arduino-uno/.

[7] LTD HANWEI ELETRONICS CO. Technical data mq-2 gas sensor. URL: https://www.mouser.com/ds/2/321/605-00008-MQ-2-Datasheet-370464.pdf.

[8] Budi Irwan. Setup restful api in yii2, July 2014. URL: http://budiirawan.com/setup-restful-api-yii2/.

[9] Dean Miller. Adafruit ccs811 air quality sensor, January 2018. URL: https://cdn-learn.adafruit.com/downloads/pdf/adafruit-ccs811-air-quality-sensor.pdf.

[10] PureMVC. Implementation idioms and best practices. URL: http://puremvc.org/.

[11] OStR Prof. Mag. Klaus-Peter Haberl Prof. Mag. Rudolf Lechner Prof. Mag. Helmut Bauer Dir. Mag. Dr. Gerhard Veidl. *Rechnungswesen & Controlling*. MANZ Verlag Schulbuch GmbH, 2014.

[12] Wikipedia. Crud. URL: https://de.wikipedia.org/wiki/CRUD.

[13] Wikipedia. Kohäsion. URL: https://de.wikipedia.org/wiki/Koh%C3%A4sion_(Informatik).

[14] Wikipedia. Lose kopplung. URL: https://de.wikipedia.org/wiki/Lose_Kopplung.

[15] Wikipedia. Model view controllers. URL: https://de.wikipedia.org/wiki/Model_View_Controller.

[16] Wikipedia. Web 2.0. URL: https://en.wikipedia.org/wiki/Web_2.0.