

Pflichtenheft und technische Spezifikation im Programmierprojekt

Projekt Vogelscheuche/Vogelerkennung

Projektname: DigitalNest

Mitarbeiter: Fiona Büttner,

Laura Isler,

Hilal Kurtoglu,

Moritz Niemeyer



Inhaltsverzeichnis

1	Visionen und Ziele.....	1
2	Anforderungsanalyse	2
2.1	Funktionale Anforderungen / Use-Cases.....	2
2.2	Nicht-funktionale Anforderungen.....	3
2.3	Risiken.....	4
2.4	GUI	4
3	Realisierung.....	5
3.1	Komponenten	5
3.2	Interne Schnittstellen / Klassendiagramm	6
3.3	Externe Schnittstellen	7
4	Entwicklungs- und Teststrategie.....	7
5	Kooperationen und Verwertungsplan	7

1 Visionen und Ziele

Unser Projekt für dieses Semester ist es eine Vogelscheuche zu programmieren. Was das für uns im Detail bedeutet, wird im Folgendem erläutert:

Wir wollen in C# ein Programm entwickeln, dass mithilfe einer gegebenen Bilderkennung Vögel erkennen kann und in Verbindung mit einem weiteren Programm auch Vogelarten voneinander unterscheiden kann. Die Bilderkennungssoftware möchten wir mit Bildern, welche wir hochladen, trainieren, um bestmögliche Ergebnisse zu erzielen. Die Bilder werden vom User erstmal selber hochgeladen. Die hochgeladenen Bilder werden dann wie vorherig beschrieben von dem Programm verarbeitet. Die Software soll auf den Bildern die Vogelarten und die Anzahl dieser identifizieren. Nach dem Abschließen der Erkennung soll der Benutzer die Möglichkeit haben das Bild zu speichern, wobei er dann zusätzlich Datum und Ort des Fotos angeben kann. Die gespeicherten Bilder können dann nach den darauf erkennbaren Vogelarten gruppiert werden. Zudem soll sich der Benutzer auch eine Statistik ausgeben lassen können, die die Anzahl der fotografierten Vögel innerhalb einer Woche bzw. eines Monats darstellt.

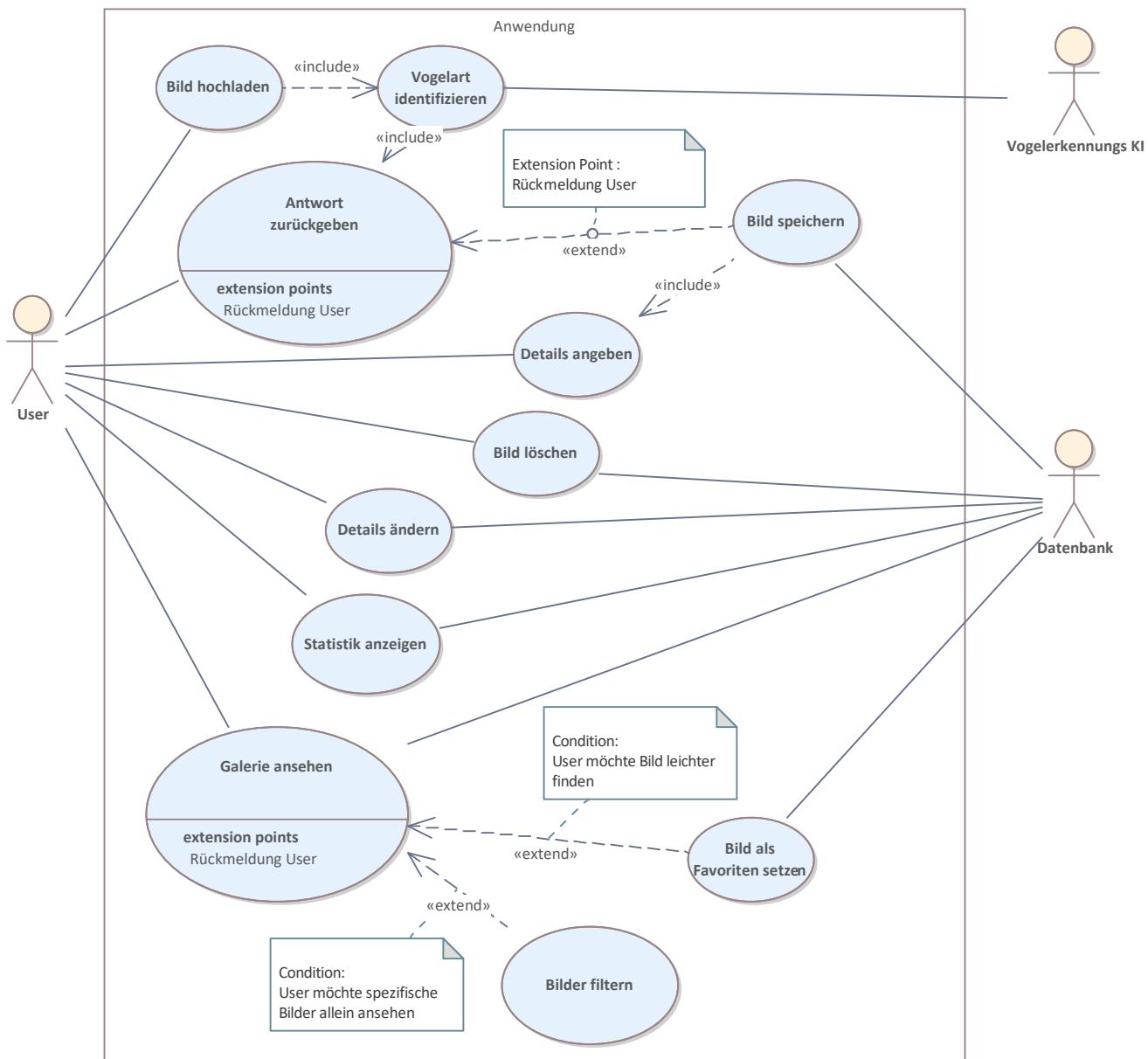
Wenn wir bis zu diesem Punkt im Projekt gekommen sind, haben wir unsere grundlegende Software fertig. Sollte noch Zeit zur Verfügung stehen, haben wir uns vorgenommen eine Kamera und einen Lautsprecher zu integrieren. Die Kamera soll automatisch ein Bild machen, wenn sie ein Vogel erkennt. Dieses Foto wird dann an das Programm weitergeleitet. Der Lautsprecher soll bei der Sichtung von bestimmten Vogelarten ein Geräusch machen, welches die Vögel verschrecken soll.

Da sich das Projekt dieses Semester auf die Software fokussiert, sind die Kamera und der Lautsprecher eher Vision als Ziel, welche wir verwirklichen werden, wenn wir es zeitlich schaffen.

2 Anforderungsanalyse

2.1 Funktionale Anforderungen / Use-Cases

UseCase	Beschreibung
Bild hochladen	User lädt ein Bild hoch
Vogelart identifizieren	Das Bild wird an die KI übergeben und die Vogelart wird bestimmt
Antwort zurückgeben	Die bestimmte Vogelart wird dem User angezeigt
Details angeben	Der User setzt Aufnahmeort und -Datum des jeweiligen Fotos
Bild speichern	Nach Identifizierung der Vogelart und Angabe der Details wird das Bild in der Datenbank gespeichert
Bild löschen	Der User löscht ein Bild
Statistik anzeigen	Der User kann sich wöchentliche und monatliche Statistiken ausgeben lassen, sowie die gesamte Anzahl an Bildern von Vögeln, die er bereits hochgeladen hat
Galerie ansehen	Der User lässt sich alle bereits hochgeladenen Bilder anzeigen
Bilder filtern	Die Bilder lassen sich nach bestimmten Gruppen filtern (z.B. Aufnahmeort/-datum, Vogelart)
Als Favorit setzen	Bilder können vom User als Favoriten gesetzt werden
Favoriten ansehen	Als Favorit gesetzte Bilder können separat angezeigt werden
Details ändern	Beim Speichern angegebene Details können vom User verändert werden



2.2 Nicht-funktionale Anforderungen

Für unser Projekt ist Visual Studio an erster Stelle unverzichtbar, da wir darauf sämtlichen Code entwickeln. Um unseren Code systematisch zu speichern und zu verwalten, nutzen wir GitLab.

Um sicherzustellen, dass der Nutzer alle Bilder hochladen kann, benötigen wir als nächstes eine leistungsstarke Datenbank, die diese Bilder speichert. Zusätzlich erhalten wir von Herrn Rodner einen Clip-Service zur Kategorisierung von Bildern. Dieser ermöglicht es, die Vögel aufzunehmen und zu erkennen.

Wenn wir bis hierhin erfolgreich sind würden wir im nächsten Schritt eine Kamera einbauen, damit der Benutzer keine Bilder mehr selber hochladen muss. Die Kamera erkennt mithilfe unseres Programms einen Vogel und macht automatisch ein Bild von diesem und speichert es in der Datenbank.

Sollten wir auch dies erfolgreich abgeschlossen haben und im Zeitplan liegen, planen wir die Integration eines Lautsprechers, um Vögel abzuschrecken. Hierfür ist jedoch auch ein Arduino oder Raspberry Pi erforderlich, auf dem das gesamte Programm ausgeführt wird. Diese Plattform wiederum benötigt ein Batterie-Pack für den Betrieb.

2.3 Risiken

Risiken	Maßnahmen
Schlechtes Zeitmanagement	Regelmäßige Absprachen (mind. Jeden Donnerstag) und kontinuierliches Arbeiten
Überforderung mit Aufgabenstellung, z.B. das Zusammenführen von unseren zwei Schnittstellen	Probleme runterbrechen, das Internet, Chat GPT und Lehrkräfte als Hilfestellung nutzen und sich gegenseitig unterstützen
Nicht aufeinander abgestimmte Teilprogramme/Klassen	Präsentieren der gegenseitigen Arbeit
Streit über bestimmte Umsetzungen	Lose ziehen, debattieren, abstimmen
Datenset der Vogelarten ist auf Englisch	Erstmal Teilübersetzung der wichtigsten Vogelarten per Hand oder auf Englisch lassen
Zeitplan wird nicht eingehalten, durch z.B. krankes Mitglied	Rücksprachen halten und evtl. Ziele und Anforderungen anpassen
Git macht Probleme (merge Problem)	Hilfe von Betreuern/Google
Kamera falsch aufgestellt (Bsp. falscher Winkel oder Lichtverhältnisse)	Kamera neu aufstellen bzw. keine Funktion bei Dunkelheit
Kamera / KI erkennt Vogel nicht	Erraten der Vogelart durch Zufallsprinzip
Technik funktioniert nicht	Kamera und Lautsprecher werden dann weggelassen

2.4 GUI



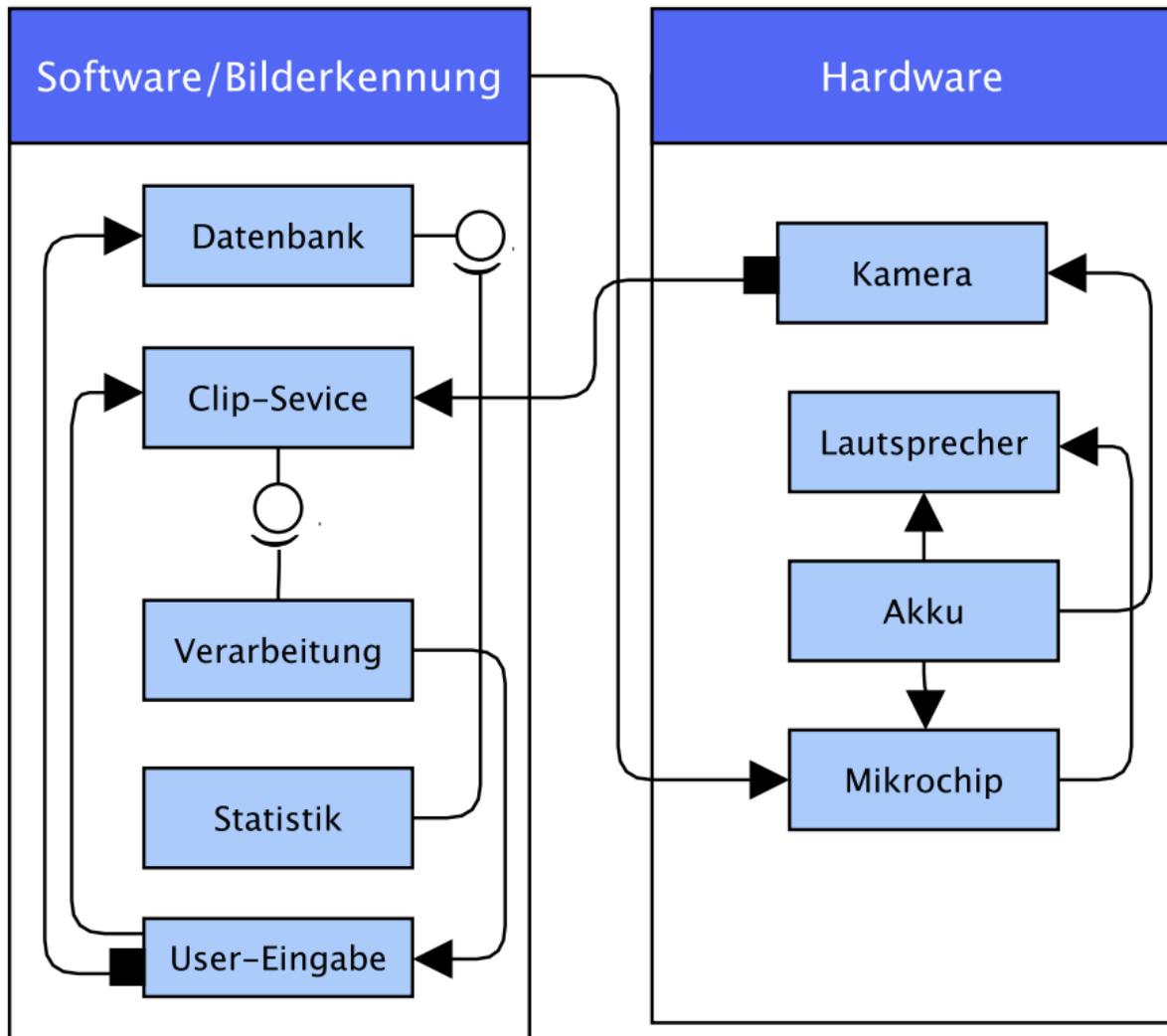






3 Realisierung

3.1 Komponenten



Unsere Endkomponenten gliedern sich in zwei Hauptgruppen – Software und Hardware. Die Hardware-Komponenten umfassen einen Akku, eine Kamera, einen Mikrochip und einen Lautsprecher. Der Akku versorgt die anderen Komponenten mit Strom, während der Mikrocontroller ein Signal von der Kamera empfängt. Das aufgenommene Bild wird von den Software-Komponenten verarbeitet. Zusätzlich sendet der Mikrocontroller ein Signal an den Lautsprecher, der daraufhin ein Geräusch erzeugt.

Die Software-Komponenten setzen sich aus dem User Interface, der Statistik, der Verarbeitung, der Bilderkennungssoftware und einer Datenbank zusammen. Die Bilder werden entweder von der Kamera über den Mikrocontroller an die

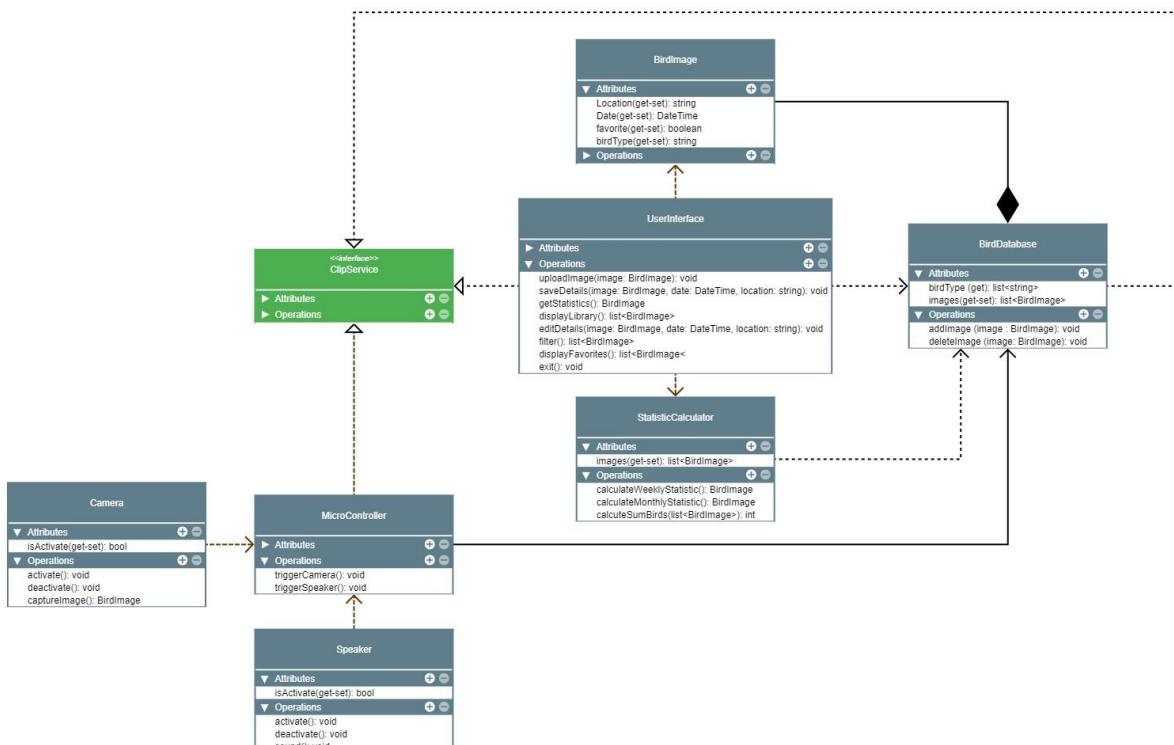
Bilderkennungssoftware weitergeleitet oder der Benutzer lädt selbstständig Bilder hoch. Die hochgeladenen Bilder werden dann in der Datenbank gespeichert. Durch die Datenbank kann sich der Benutzer dann eine Statistik über die erkannten Vögel ausgeben lassen.

Die Zuständigkeiten in unserem Team sind klar definiert:

Laura übernimmt die Verantwortung für die Benutzeroberfläche (User Interface) und die dazugehörige Verarbeitung. Moritz ist für die Gestaltung und Implementierung der Datenbankkomponente verantwortlich. Hilal kümmert sich um die Funktionalitäten zur Erstellung und Anzeige von statistischen Informationen über die erkannten Vögel und Fiona ist für die Integration mit dem Clip-Service verantwortlich und gewährleistet eine reibungslose Kommunikation zwischen unserer Anwendung und dem Clip-Service.

Die Hardware-Komponenten, werden derzeit als gemeinsame Aufgabe betrachtet, da die genaue Aufteilung dieser Verantwortlichkeiten noch abhängig von der zeitlichen Machbarkeit ist und später im Projektverlauf festgelegt wird.

3.2 Interne Schnittstellen / Klassendiagramm



3.3 Externe Schnittstellen

Gemäß unserem aktuellen erfordert unser System derzeit keine externen Schnittstellen. Alle erforderlichen Komponenten, einschließlich der Benutzeroberfläche, der Bilderkennungssoftware, der Datenbank und möglicher zukünftiger Funktionen wie der Kamera und des Lautsprechers, sind intern miteinander verbunden.

Es ist jedoch wichtig zu beachten, dass sich die Anforderungen im Verlauf des Projekts ändern können.

4 Entwicklungs- und Teststrategie

Für die verschiedenen Haupt-Use-Cases haben wir uns Testmöglichkeiten überlegt.

Eine wichtige Funktion ist das Hochladen und Speichern von Bildern. Das wird überprüft, indem wir eine bestimmte Anzahl, von zum Beispiel 5 Bildern hochladen und speichern. Anschließend wird kontrolliert, ob die Bilder mit den Details in der Datenbank vorhanden sind.

Diese Bilder werden dann über die Galerie aufgerufen und die Details werden stichprobenweise bearbeitet. Daraufhin muss wieder kontrolliert werden, ob eine Veränderung der Details in der Datenbank erkennbar ist. Das gleiche Prinzip wird auch bei dem Testen der Löschfunktion durchgeführt.

Um zu kontrollieren, inwiefern der Vogel richtig erkannt wurde, werden Bilder hochgeladen, auf denen unter anderem verschiedene Vogelarten abgebildet sind, aber auch welche auf denen kein Vogel darauf zu sehen ist. So wird nicht nur überprüft, ob die Vogelerkennung, sondern auch die Erkennung der Vogelart funktioniert.

Eine weitere wichtige Funktion ist die Statistikausgabe. Hier werden ebenfalls Vogelbilder hochgeladen mit verschiedenen Daten und Orten. Es muss sich gemerkt werden wie viele Bilder hochgeladen wurden und mit welchen Angaben, um das richtige Ergebnis mit dem ausgegebenen Ergebnis zu vergleichen.

5 Kooperationen und Verwertungsplan

Unsere Software soll als Open Source Tool unter der MIT-Lizenz auf GitHub veröffentlicht werden. Das erlaubt der Community Verbesserungen vorzunehmen oder auch das Produkt auf die eigenen Anforderungen weiter zu individualisieren. Es wird ebenfalls ein Docker-Image sowie eine Docker Compose Datei veröffentlicht, damit Anwender mit geringem Aufwand die Software lokal einsetzen können. In anschließenden Arbeiten ist ebenfalls eine dedizierte App-Version geplant sowie ein Community-Tool um eine globale Vogeldatenbank aufzubauen.

Unsere Anwendung soll speziell in Gärten zur Verschreckung von Vögeln oder für Ornithologen/ Vogelfreunde zum Erstellen einer Digitalen Bibliothek angewendet werden.