

# Entwicklerdokumentation

## Einführung

Diese Dokumentation bietet einen Einblick in das Landwirtschafts-Informationssystem. Die Anwendung ist auf Flask aufgebaut und ermöglicht es Benutzern, verschiedene Aspekte der Landwirtschaft zu verwalten, darunter Fahrzeuge, Viehbestand, Futterbestand, Flächen und Ernteertrag.

## Anforderungen

Um die Anwendung ausführen zu können, benötigen Sie:

- Python 3.6 oder höher
- Flask
- PostgreSQL-Datenbank
- Docker und Docker Compose (um die Anwendung per One-Click zu starten)

## Dateistruktur

Die Hauptdateien und Verzeichnisse in der Anwendung sind:

- **app.py:** Diese Hauptanwendungsdatei startet den Flask-Server und definiert Routen für verschiedene Funktionen. Die Flask-Routen werden mittels `@app.route('/...', methods=...)` definiert, wobei die gewünschte Operation durch die Angabe von `methods=['POST/GET/...']` entsprechend festgelegt wird.
- **models.py:** Diese Datei enthält die Datenbankmodelle, die mit SQLAlchemy definiert sind. Sie bildet die Struktur der Datenbanktabellen ab.
- **db\_operation\_\*.py:** In diesen Dateien sind CRUD-Operationen für die Datenbanktabellen implementiert. Sie ermöglichen die Manipulation der Daten mithilfe von SQLAlchemy. Jede Datei deckt CRUD-Operationen für einen spezifischen Datenbankbereich ab.
- **static/:** Dieses Verzeichnis enthält statische Dateien wie Bilder und CSS, die von den HTML-Vorlagen verwendet werden.
- **templates/:** Dieses Verzeichnis enthält Jinja2-HTML-Vorlagen, die vom Flask-Framework gerendert werden, um dynamische Inhalte zu generieren und an den Benutzer zu übermitteln.

Um auf Daten von der Flask-App im Frontend zuzugreifen, werden Jinja2-Templates genutzt. Diese ermöglichen es, dynamische Inhalte in das HTML einzubinden. Dabei werden spezielle Jinja2-Tags wie `{{ variable }}` verwendet, wobei „variable“ den Namen der von der Flask-Route übergebenen Variablen darstellt. Beim Rendern der Seite werden diese Tags durch die entsprechenden Daten ersetzt, die von der Flask-App bereitgestellt wurden.

## Installation und Ausführung

Der Befehl `docker-compose up -d` kann verwendet werden, um die Container gemäß der Konfiguration in der `docker-compose.yml`-Datei zu starten. Das `-d` Flag startet die Container im „detached“ Modus, was bedeutet, dass sie im Hintergrund ausgeführt werden, ohne die aktuelle Shell-Sitzung zu blockieren.

## Erläuterung des docker-compose.yml Files

### 1. docker-compose.yml Konfiguration:

- **version:** Diese Angabe definiert die Version der Docker-Compose-Syntax, die verwendet wird.
- **services:** Hier werden die verschiedenen Container definiert, die gestartet werden sollen.

## 2. PostgreSQL-Datenbank-Service:

- **image:** Das ist das Docker-Image, das für den PostgreSQL-Datenbankcontainer verwendet wird. Hier wird die Version 13 von PostgreSQL verwendet.
- **environment:** Hier werden Umgebungsvariablen definiert, die von PostgreSQL während der Containerausführung genutzt werden. In diesem Fall werden der Name der Datenbank, der Benutzername und das Passwort festgelegt.

## 3. Webanwendungs-Service:

- **build:** Dieser Befehl sagt Docker-Compose, dass es das Dockerfile im aktuellen Verzeichnis verwenden soll, um das Image für diesen Service zu bauen.
- **ports:** Hier wird Port-Mapping definiert, um den Port 5000 des Hostsystems mit dem Port 5000 des Containers zu verbinden. Dies bedeutet, dass die Webanwendung über den Port 5000 des Hostsystems erreichbar ist.
- **depends\_on:** Diese Anweisung stellt sicher, dass der Webanwendungs-Service erst gestartet wird, nachdem der PostgreSQL-Service gestartet wurde.
- **volumes:** Hier wird ein sogenanntes „Volume“ definiert, dass das Verzeichnis **./static** auf dem Hostsystem mit dem Verzeichnis **/app/static** im Container verbindet. Dies ermöglicht es, Dateien (wie z.B. Bilder) zwischen dem Host und dem Container auszutauschen.

Es wird ein Dockerfile verwendet, um die Umgebung für eine Anwendung innerhalb eines Docker-Containers zu definieren und einzurichten.

**FROM:** Festlegung des Python 3.6 Basisimages.

**WORKDIR:** Setzt das Arbeitsverzeichnis im Container auf `/app`.

**COPY requirements.txt requirements.txt**

**RUN pip install -r requirements.txt:** Kopiert die requirements.txt Datei ins Arbeitsverzeichnis und installiert die darin aufgeführten Pakete.

**COPY:** Kopiert alle Dateien und Verzeichnisse aus dem aktuellen Verzeichnis (des Hostsystems) in das Arbeitsverzeichnis des Containers.

**COPY static static:** Kopiert den Ordner „static“ aus dem aktuellen Verzeichnis des Hostsystems in das Arbeitsverzeichnis des Containers.

**CMD [“python“, “app.py“]:** Legt den Befehl fest, der ausgeführt wird, wenn der Container gestartet wird, um die Python-Anwendung app.py auszuführen.

## Funktionalitäten

Die Anwendung bietet folgende Funktionalitäten:

- Verwaltung von Fahrzeugen
- Verwaltung des Viehbestands
- Verwaltung des Futterbestands
- Verwaltung von Flächen
- Verwaltung des Ernteertrags

Jede dieser Funktionen umfasst Hinzufügen, Bearbeiten, Löschen und Anzeigen von entsprechenden Einträgen. Um den Ernteertrag graphisch zu visualisieren, wurde matplotlib verwendet.

## Tests

Die Anwendung enthält Unit-Tests, die mit Pytest geschrieben wurden. Diese Tests überprüfen das Verhalten der CRUD-Operationen für verschiedene Datenbanktabellen. Die Tests können mittels `pytest modulname.py` ausgeführt werden. Damit die „tatsächliche“ Datenbank nicht angegriffen wird, wurde eine `in:memory` Datenbank verwendet.

## Benutzerdokumentation


### Landing-Page

Nach starten der Anwendung gibt es die Möglichkeit das Wetter der nächsten 3 Tage abzufragen. Dafür ist die Eingabe der Stadt, des Landes und der Postleitzahl erforderlich. Für diese Integration wird mit einer externen API von `weatherbit.io` kommuniziert.


### Wettervorhersage

Wetter abrufen


#### Hartberg, AT




**2024-04-15**  
Max Temperatur: 22 °C  
Min Temperatur: 13.9 °C  
Wetter: Mäßiger Regen



**2024-04-16**  
Max Temperatur: 13.3 °C  
Min Temperatur: 4.3 °C  
Wetter: Starker Regen



**2024-04-17**  
Max Temperatur: 10.2 °C  
Min Temperatur: 3.5 °C  
Wetter: Bewölkt



**2024-04-18**  
Max Temperatur: 10.7 °C  
Min Temperatur: 3.9 °C  
Wetter: Aufgelockert Bewölkt

## Tierbestand

Es besteht die Möglichkeit Daten der Tiere zu speichern. Nach Klick auf den Button Tiere anzeigen werden vorhandene Tiere in der Datenbank aufgelistet. Hier sind das Updaten und Löschen von einzelnen Tieren auch möglich. Mittels der Suchleiste kann nach Ohrmarke oder Art des Tieres gesucht werden.

[Home](#) [Tierbestand](#) [Futterbestand](#) [Fahrzeuge](#) [Flächen](#) [Ernteertrag](#)

Suche nach Ohrmarke oder Art:  
  
Suchen

Tier hinzufügen Tiere anzeigen

**Tier hinzufügen** ✕

Ohrmarke:

Art:

Geburtstag:

Geschlecht:  

Männlich

Bild hochladen:  

Durchsuchen...

 Keine Datei ausgewählt.

Tier hinzufügen

[Home](#) [Tierbestand](#) [Futterbestand](#) [Fahrzeuge](#) [Flächen](#) [Ernteertrag](#)

Suche nach Ohrmarke oder Art:  
  
Suchen

Tier hinzufügen Tiere anzeigen

	Ohrmarke	Art	Geburtstag	Geschlecht	Aktionen
	x123	Geflügel	2024-04-01	weiblich	<button>Löschen</button> <button>Edit</button>

Tier hinzufügen

Tiere anzeigen

Tier bearbeiten

X

Ohrmarke:

x123

Art:

Geflügel

Geburtstag:

01 . 04 . 2024

Geschlecht:

Weiblich

Bild ändern:

Durchsuchen...

Keine Datei ausgewählt.

Tier aktualisieren

## Fahrzeugbestand

Dem Benutzer wird es ermöglicht Fahrzeuge hinzuzufügen. Dabei werden beim Jahr nur vierstellige Zahlen in einem festgelegten Definitionsbereich erlaubt. Löschen, Updaten und Anzeigen der Fahrzeuge wird ermöglicht. Gesucht kann nach der Marke oder dem Modell werden.

Home

Tierbestand

Futterbestand

Flächen

Ernteertrag

Suche nach Fahrzeugen:

Marke oder Modell eingeben

Suchen

Fahrzeug hinzufügen

Fahrzeug anzeigen

Fahrzeug hinzufügen

X

Marke:

Massey Ferguson

Modell:

x230

Jahr:

1990

Bild hochladen:

Durchsuchen...

ferguson.png

Fahrzeug hinzufügen

HomeTierbestandFutterbestandFlächenErnteertrag



Suche nach Fahrzeugen:

Marke oder Modell eingeben

Suchen

Fahrzeug hinzufügen

Fahrzeug anzeigen

	Marke	Modell	Jahr	Aktionen
	Valtra	x129	2024	<div>LöschenEdit</div>
	Massey Ferguson	x230	1990	<div>LöschenEdit</div>

Fahrzeug hinzufügen

Fahrzeug anzeigen

Fahrzeug bearbeiten

X

Marke:

Massey Ferguson

Modell:

x230

Jahr:

1990

Bild ändern:

Durchsuchen...

Keine Datei ausgewählt.

Fahrzeug aktualisieren

## Futterbestand

Um einen Überblick des aktuellen Futterbestands zu haben ist es möglich Futterartikel hinzuzufügen. Hierbei ist die Auswahl der Einheit kg/t möglich. Das Futter kann angezeigt, nach Kategorie gesucht, upgedatet und gelöscht werden.

HomeTierbestandFahrzeugeFlächenErnteertrag

Suche nach Futter:

Kategorie eingeben

Suchen

Futter hinzufügen

Futter anzeigen

Futter hinzufügen

Kategorie:Menge:Einheit:

kg

Futter hinzufügen

HomeTierbestandFahrzeugeFlächenErnteertrag

Suche nach Futter:

Kategorie eingeben

Suchen

Futter hinzufügen

Futter anzeigen

Kategorie	Menge	Einheit	Aktionen
Weizen	20	kg	<div>LöschenEdit</div>

Futter hinzufügen

Futter anzeigen

Futter bearbeiten

Kategorie:

Weizen

Menge:

20

Einheit:

kg

Futter aktualisieren

## Flächenbestand

Um einen Überblick über eigene Flächen zu haben, wird zwischen Eigenbesitz und Gepachtet differenziert. Die Flächen können upgedatet, angezeigt, gelöscht und nach Eigenbesitz/Gepachtet gesucht werden.

Suche nach: Eigenbesitz Suchen

Fläche hinzufügenFlächen anzeigen

### Fläche hinzufügen

X

Typ:

Größe:

Einheit:  
m<sup>2</sup> ▼

Besitz:  
Eigenbesitz ▼

Fläche hinzufügen

Fläche hinzufügenFlächen anzeigen

### Fläche bearbeiten

X

Typ:

Größe:

Einheit:  
m<sup>2</sup> ▼

Besitz:  
Gepachtet ▼

Vorname:

Nachname:

Telefonnummer:

Straße:



## Ernteertrag

Ernteerträge können gespeichert werden. Bei Bedarf können diese auch gelöscht, upgedatet oder gesucht werden. Mithilfe eines mitgespeicherten Zeitstempels können die Ernteerträge graphisch visualisiert werden.

Ernte hinzufügen

Ernte anzeigen

Ernteertrag Chart anzeigen

Ernte hinzufügen

Ernte:

Menge:

Einheit:

kg

Datum:

TT . MM . JJJJ

Fläche:

Weizenfeld

Ernte hinzufügen

Home

Tierbestand

Futterbestand

Fahrzeuge

Flächen

Ernte hinzufügen

Ernte anzeigen

Ernteertrag Chart anzeigen

Ernte	Menge	Einheit	Datum	Fläche	Aktionen
Weizen	20	kg	2024-04-01	Weizenfeld	<div>LöschenEdit</div>
Roggen	200	kg	2024-04-10	Weizenfeld	<div>LöschenEdit</div>
Gerste	2000	kg	2024-04-15	Weizenfeld	<div>LöschenEdit</div>

Ernte hinzufügen

Ernte anzeigen

Ernteertrag Chart anzeigen

## Ernte bearbeiten

Ernte:

Weizen

Menge:

20

Einheit:

kg

Datum:

01.04.2024

Fläche:

Weizenfeld

Ernte aktualisieren

