

Arbeitsumgebung (Arbeiten mit Jupyter Notebooks)

Jacques Mock Schindler

25.08.2025

Ausgangslage

In der Informatik geht es darum, wie Informationsverarbeitung mit Hilfe von Computern automatisiert werden kann.

Die Automatisierung der Informationsverarbeitung erfordert die Verwendung von Programmiersprachen. Im Informatikunterricht wird in erster Linie mit der Programmiersprache Python gearbeitet.

Im folgenden findet sich eine Anleitung für die Installation der für den Unterricht erforderlichen Programme.

Installation von Python

Dieser Abschnitt führt Sie Schritt für Schritt durch die Installation von Python auf einem Windows-Rechner.

Microsoft Store Falle

Achten Sie beim Herunterladen von Python darauf, dass Sie sich auf der offiziellen Seite von Python (<https://www.python.org>) und **nicht** im Microsoft Store befinden. Wenn Sie versehentlich die Python Version aus dem Microsoft Store installiert haben, kann das bei der Arbeit an den Schulprojekten zu Problemen führen.

Deinstallieren Sie die Microsoft Version von Python und installieren Sie die Version von der offiziellen Website.

1. Laden Sie die neueste Version von Python von der offiziellen Website herunter: [python.org](https://www.python.org).
2. Führen Sie das heruntergeladene Installationsprogramm durch Doppelklick auf die Datei aus. Stellen Sie sicher, dass Sie die Option "Add Python to PATH" aktivieren, bevor Sie auf "Install Now" klicken.

💡 Die 'PATH'-Umgebungsvariable

Stellen Sie sich die PATH-Variable wie ein Adressbuch für die Kommandozeile (Terminal) vor. Wenn Sie einen Befehl wie `python` eingeben, schaut der Computer in diesem Adressbuch nach, wo das entsprechende Programm zu finden ist. Indem Sie das Häkchen bei "Add Python to PATH" setzen, fügen Sie die Adresse des Python-Interpreters zu diesem Adressbuch hinzu. Ohne diesen Eintrag weiss der Computer nicht, wo er suchen soll, und meldet, dass er den Befehl nicht kennt.

- Überprüfen Sie die Installation, indem Sie die Eingabeaufforderung öffnen (Terminal → Windows-Taste + R, dann `cmd` eingeben) und den Befehl `python --version` eingeben. Dies sollte die installierte Python-Version anzeigen.

Hello World

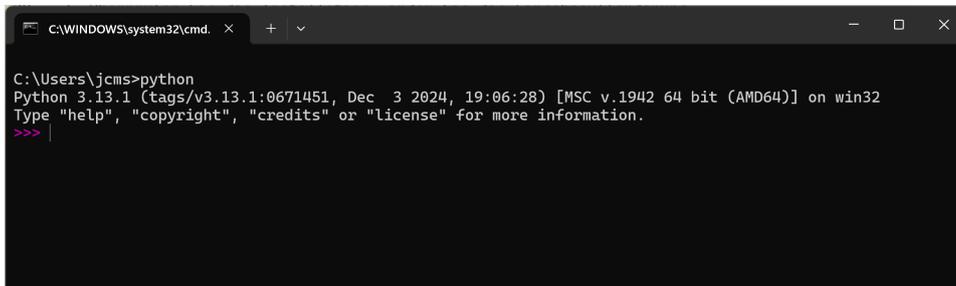
Es hat sich eingebürgert, dass das erste Programm, das ausgeführt wird, ein Programm ist, das den Text "Hello World" auf dem Bildschirm ausgibt. Um dieser Tradition zu folgen, führen Sie die folgenden Schritte aus:

- Öffnen Sie ein Terminal (Windows-Taste + R, dann `cmd` eingeben).

💡 Das Terminal

Unter dem Begriff "Terminal" versteht man ein Programm, das eine textbasierte Benutzeroberfläche bereitstellt, um mit dem Betriebssystem zu interagieren. In einem Terminal können Sie Befehle eingeben und erhalten die Ausgaben direkt im Fenster.

- Geben Sie den Befehl `python` ein, um die Python-Shell zu starten. Die Python Shell sollte ungefähr so, wie das folgende Bild aussehen.



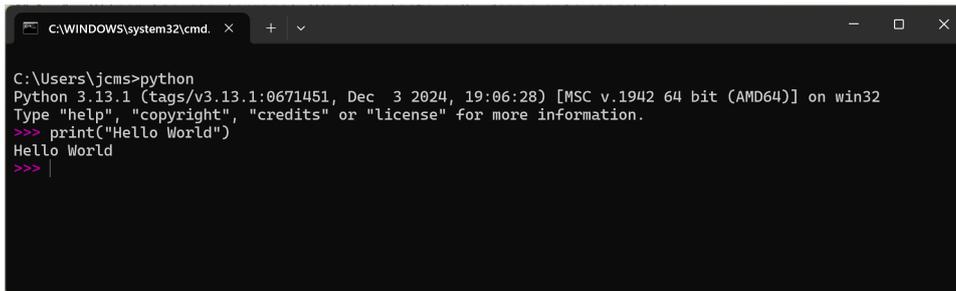
```
C:\WINDOWS\system32\cmd. x + v
C:\Users\jcms>python
Python 3.13.1 (tags/v3.13.1:0671451, Dec 3 2024, 19:06:28) [MSC v.1942 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

Abbildung 1: Python Shell

3. Geben Sie den folgenden Befehl ein und drücken Sie anschliessend die Eingabetaste:

```
print("Hello World")
```

Das Resultat sollte wie das folgende Bild aussehen.



```
C:\WINDOWS\system32\cmd. x + v
C:\Users\jcms>python
Python 3.13.1 (tags/v3.13.1:0671451, Dec 3 2024, 19:06:28) [MSC v.1942 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> print("Hello World")
Hello World
>>>
```

Abbildung 2: Python Shell

Gratuliere - Sie haben Ihr erstes Python-Programm erfolgreich ausgeführt!

Arbeitsumgebung

Im Unterricht wird nicht direkt in der Python-Shell gearbeitet, sondern mit sogenannten Jupyter Notebooks. Jupyter Notebooks ermöglichen es, in der gleichen Datei sowohl Code (Programm Teile) als auch formatierten Text (in Markdown) zu verarbeiten. Eine Jupyter Notebook Datei hat die Endung `.ipynb`. Vom Jupyter Notebook unterschieden werden muss die Arbeitsoberfläche in welcher die Jupyter Notebooks bearbeitet werden. Diese Oberfläche nennt sich JupyterLab und läuft in einem Webbrowser.

💡 Das Jupyter Ökosystem

Die im Unterricht verwendeten Jupyter Notebooks sind Teil eines ganzen Jupyter Ökosystems. Der Name Jupyter setzt sich aus den drei Programmiersprachen **J**ulia, **P**ython und **R** zusammen, die in diesem Ökosystem eine zentrale Rolle spielen. Zum Jupyter Ökosystem gehören auch zahlreiche Erweiterungen und Tools, die die Arbeit mit Notebooks und Daten erleichtern.

Der Unterricht beschränkt sich auf die Verwendung von Jupyter Notebooks mit der Programmiersprache Python sowie den Einsatz von JupyterLab als Arbeitsumgebung.

Damit dies alles funktioniert, braucht es ein paar weitere Vorbereitungsarbeiten.

1. Erstellen Sie im Ordner "Informatik" einen Unterordner mit dem heutigen Datum als Namen. Formatieren Sie das Datum nach dem Schema "YYMMDD", für den 1. August 2025 wäre das zum Beispiel "250801".
2. Öffnen Sie den soeben erstellten Ordner.
3. Geben Sie die Tastenfolge `Ctrl + L` ein, um die Adresszeile des Dateimanagers zu aktivieren.
4. Überschreiben Sie den Inhalt der Adresszeile mit dem Text `cmd` und drücken Sie die Eingabetaste. Dadurch wird ein Terminal geöffnet, das direkt im aktuellen Ordner arbeitet.
5. Geben Sie im neu geöffneten Terminal den folgenden Befehl ein und drücken Sie die Eingabetaste:

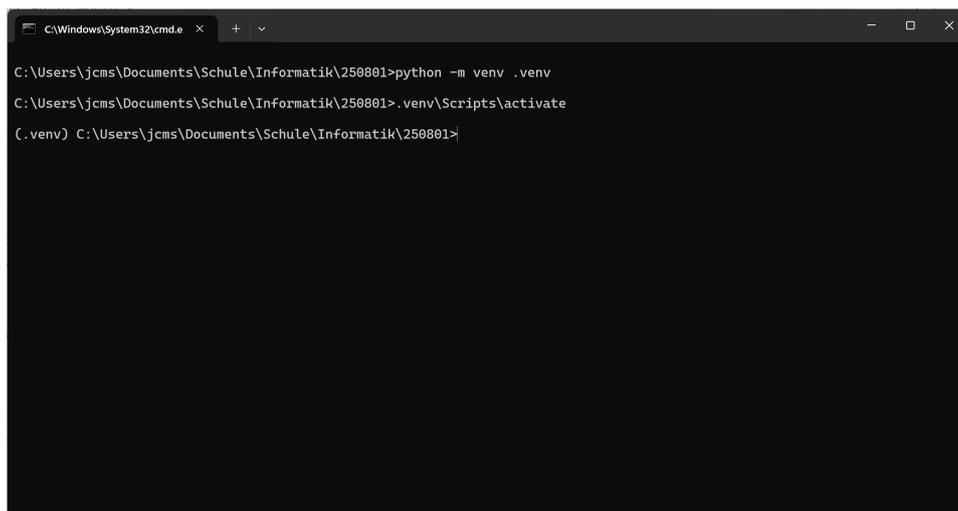
```
python -m venv .venv
```

Dadurch wird eine sogenannte Python Virtual Environment erstellt (venv wegen **V**irtual **E**nvironment). Dieses Python Virtual Environment dient dazu, die eigenen Programmierprojekte unabhängig voneinander gestalten zu können.

6. Aktivieren Sie das Python Virtual Environment mit dem folgenden Befehl:

```
.venv\Scripts\activate
```

Ihr Terminal sieht nach dem Erstellen und Aktivieren der Python Virtual Environment ungefähr so aus:



```
C:\Windows\System32\cmd.exe x + v - □ x
C:\Users\jcms\Documents\Schule\Informatik\250801>python -m venv .venv
C:\Users\jcms\Documents\Schule\Informatik\250801>.venv\Scripts\activate
(.venv) C:\Users\jcms\Documents\Schule\Informatik\250801>
```

Abbildung 3: Aktivierte Python Virtual Environment

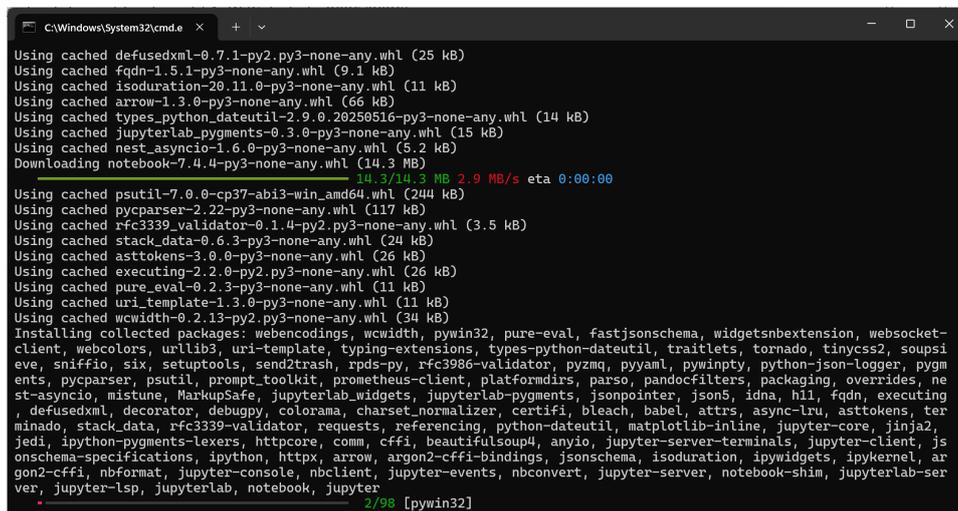
Das Wort in der Klammer am Anfang der Zeile zeigt den Namen der aktiven Python Virtual Environment an. Im vorliegenden Fall ist das `.venv`.

7. In der nun aktivierten Python Virtual Environment installieren Sie die benötigten Pakete mit dem folgenden Befehl:

```
pip install jupyter
```

Das dauert eine Weile.

Während der Installation werden die benötigten Pakete (Ergänzungen zur bestehenden Python Installation) heruntergeladen und in der Python Virtual Environment gespeichert. Das Terminal sieht dabei ungefähr so aus:



```
C:\Windows\System32\cmd.exe x + v
Using cached defusedxml-0.7.1-py2.py3-none-any.whl (25 kB)
Using cached fqdn-1.5.1-py3-none-any.whl (9.1 kB)
Using cached isoduration-20.11.0-py3-none-any.whl (11 kB)
Using cached arrow-1.3.0-py3-none-any.whl (66 kB)
Using cached types_python_dateutil-2.9.0.20250516-py3-none-any.whl (14 kB)
Using cached jupyterlab_pygments-0.3.0-py3-none-any.whl (15 kB)
Using cached nest_asyncio-1.6.0-py3-none-any.whl (5.2 kB)
Downloading notebook-7.4.4-py3-none-any.whl (14.3 MB)
14.3/14.3 MB 2.9 MB/s eta 0:00:00
Using cached psutil-7.0.0-cp37-abi3-win_amd64.whl (244 kB)
Using cached pyparsing-2.2.2-py3-none-any.whl (117 kB)
Using cached rfc3339_validator-0.1.4-py2.py3-none-any.whl (3.5 kB)
Using cached stack_data-0.6.3-py3-none-any.whl (24 kB)
Using cached asttokens-3.0.0-py3-none-any.whl (26 kB)
Using cached executing-2.2.0-py2.py3-none-any.whl (26 kB)
Using cached pure_eval-0.2.3-py3-none-any.whl (11 kB)
Using cached uri_template-1.3.0-py3-none-any.whl (11 kB)
Using cached wcwidth-0.2.13-py2.py3-none-any.whl (34 kB)
Installing collected packages: webencodings, wcwidth, pywin32, pure-eval, fastjsonschema, widgetsnbextension, websocket-client, webcolors, urllib3, uri-template, typing-extensions, types-python-dateutil, traitlets, tornado, tinycss2, soupsieve, sniffio, six, setuptools, send2trash, rpds-py, rfc3986-validator, pyzmq, pyyaml, pywinpty, python-json-logger, pygments, pyparsing, psutil, prompt_toolkit, prometheus-client, platformdirs, parso, pandocfilters, packaging, overrides, nest-asyncio, mistune, MarkupSafe, jupyterlab_widgets, jupyterlab-pygments, jsonpointer, json5, idna, h11, fqdn, executing, defusedxml, decorator, debugpy, colorama, charset_normalizer, certifi, bleach, babel, attrs, async-lru, asttokens, terminado, minio, stack_data, rfc3339-validator, requests, referencing, python-dateutil, matplotlib-inline, jupyter-core, jinja2, jedi, ipython-pygments-lexers, httpcore, comm, cffi, beautifulsoup4, anyio, jupyter-server-terminals, jupyter-client, jsonschema-specifications, ipython, httpx, arrow, argon2-cffi-bindings, jsonschema, isoduration, ipywidgets, ipykernel, argon2-cffi, nbformat, jupyter-console, nbclient, jupyter-events, nbconvert, jupyter-server, notebook-shim, jupyterlab-server, jupyter-lsp, jupyterlab, notebook, jupyter
```

Abbildung 4: Terminal während der Jupyter Installation

Alle in einer Python Virtual Environment installierten Pakete sind innerhalb dieser Umgebung dauerhaft verfügbar und müssen daher für das gleiche Projekt kein zweites Mal installiert werden.

8. Starten Sie den Jupyter Server mit dem folgenden Befehl:

```
jupyter-lab
```

Dies startet den Jupyter Notebook Server und öffnet automatisch ein Browserfenster mit der Jupyter Notebook Oberfläche.

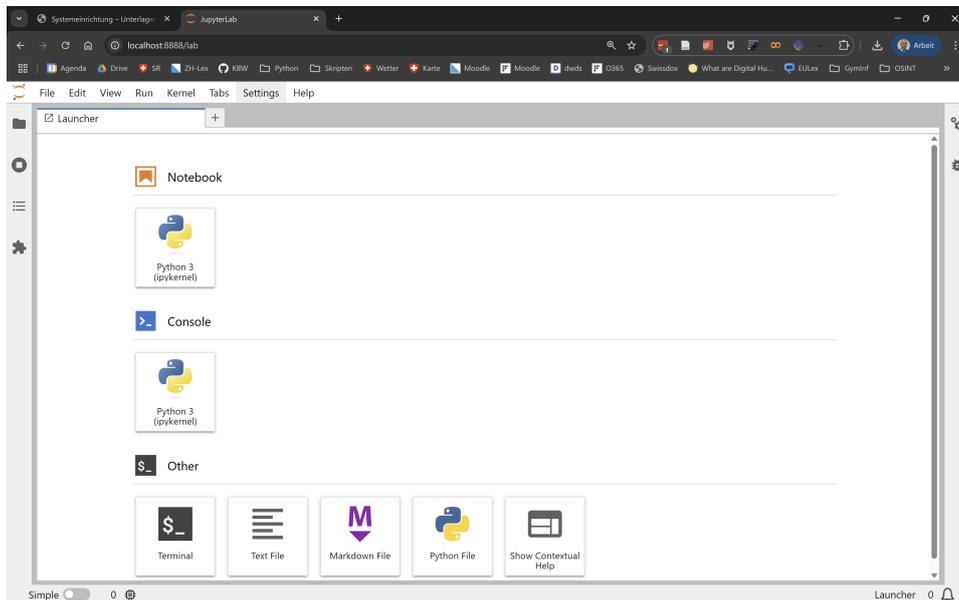


Abbildung 5: Startseite Jupyter Lab

9. Klicken Sie auf den Button "Python 3 (ipykernel)" unter dem Titel Notebook.

Damit starten Sie ein neues Jupyter Notebook. Der Cursor blinkt in einer leeren Zelle. Bei dieser Zelle handelt es sich um eine sogenannte Code-Zelle. In einer Code-Zelle können Sie Python Code eingeben und ausführen.

Überprüfen Sie das, indem Sie in der Zelle den Befehl `print("Hello World")` eingeben und anschliessend die Tastenfolge `Shift + Enter` drücken (alternativ können Sie auch auf den Button "Run" in der Werkzeugleiste klicken).

Das Resultat sollte wie das folgende Bild aussehen.



Abbildung 6: Hello World in einem Jupyter Notebook

In einem Jupyter Notebook können Sie nicht nur Python Code ausführen, sondern auch Text (formatiert in Markdown) darstellen.

Für die Darstellung von Text müssen Sie die Zelle als Text-Zelle markieren. Dazu klicken Sie auf den Button "Code" in der Werkzeugleiste

und wählen im Dropdown-Menü die Option "Markdown" aus.

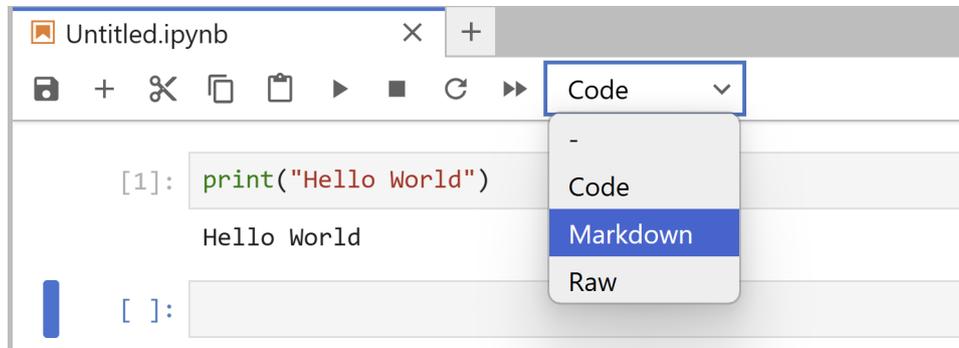


Abbildung 7: Umstellen der Zelle auf Markdown

Probieren Sie das aus. Schreiben Sie einen Titel und einen kurzen Text in die Zelle unterhalb der Code-Zelle mit `print("Hello World")`. Damit der Text in der Zelle formatiert angezeigt wird, müssen Sie die Zelle mit der Tastenfolge `Shift + Enter` ausführen (analog zum Ausführen von Code-Zellen).

Eine Zelle ist entweder eine Code-Zelle oder eine Text-Zelle. Für den Wechsel zwischen Code- und Text-Darstellung müssen Sie je eine neue Zelle anlegen. Das geht mit der Tastenfolge `Esc + B` (für "Below") oder `Esc + A` (für "Above") während Sie sich in einer Zelle befinden. Alternativ können Sie auch die Buttons "Insert Cell Below" oder "Insert Cell Above" aus den Werkzeugen der Zelle verwenden.

10. Das Jupyter Notebook ist eine Datei mit der Endung `.ipynb`. Neu erstellte Jupyter Notebooks erhalten den Namen "Untitled.ipynb". Um diesen Namen zu ändern, klicken Sie mit der rechten Maustaste auf den Titel "Untitled" in der oberen linken Ecke des Jupyter Notebooks und wählen Sie die Option "Rename" aus dem Kontextmenü. Anschliessend können Sie den Namen des Jupyter Notebooks eingeben.

Öffnen bestehender Jupyter Notebooks

Häufiger als das Erstellen eines neuen Jupyter Notebooks ist das Öffnen eines bereits bestehenden Jupyter Notebooks. Hier wird das entsprechende Vorgehen beschrieben.

1. Navigieren Sie in den Ordner in dem sich das Jupyter Notebook befindet.

2. Stellen Sie sicher, dass der Ordner über eine Python Virtual Environment mit installierten Jupyter Paketen verfügt.

Öffnen Sie dazu im ausgewählten Ordner das Terminal (`ctrl + L` anschließend `cmd` und Eingabetaste). Dann starten Sie die Python Virtual Environment und geben den Befehl `pip list` ein. Dieser Listet alle in der Python Virtual Environment installierten Pakete auf. Falls die Jupyter Pakete nicht aufgelistet werden, müssen Sie diese wie [oben](#) beschrieben installieren.

3. Starten Sie den Jupyter Server mit dem Befehl `jupyter-lab`.
4. Öffnen Sie das Dateiverzeichnis. Dazu müssen Sie auf dem linken Rand das Ordner-Symbol anklicken.
5. Wählen Sie das Jupyter Notebook aus, das Sie öffnen möchten. Mit einem Doppelklick auf das Jupyter Notebook wird dieses geöffnet.

Wenn Sie das Dateiverzeichnis wieder schliessen möchten, klicken Sie auf das Ordner-Symbol auf der linken Seite erneut.

Diese Schritte funktionieren auch, wenn Sie ein Jupyter Notebook öffnen möchten, das Sie von jemand anderem erhalten haben. Sie müssen dieses dazu lediglich in den Ordner kopieren, in dem sich die Python Virtual Environment mit den installierten Jupyter Paketen befindet.

Häufige Fehlermeldungen

Problem / Fehlermeldung (Was Sie sehen)	Mögliche Ursache (Warum es passiert)	Lösung (Was Sie tun können)
Der Befehl <code>python</code> ist entweder falsch geschrieben oder konnte nicht gefunden werden.	Python wurde bei der Installation nicht zur PATH-Variable hinzugefügt. Der Computer weiss nicht, wo er <code>python.exe</code> finden soll.	<ol style="list-style-type: none"> 1. Deinstallieren Sie Python über die Systemsteuerung. 2. Installieren Sie Python erneut. 3. Achten Sie diesmal unbedingt darauf, das Häkchen bei "Add Python to PATH" zu setzen.

Problem / Fehlermeldung (Was Sie sehen)	Mögliche Ursache (Warum es passiert)	Lösung (Was Sie tun können)
Der Befehl <code>jupyter-lab</code> ist entweder falsch geschrieben oder konnte nicht gefunden werden.	Sie haben vergessen, die virtuelle Umgebung zu aktivieren. Der Befehl <code>jupyter-lab</code> existiert nur innerhalb der aktivierten Umgebung.	<ol style="list-style-type: none"> Überprüfen Sie, ob <code>(.venv)</code> am Anfang der Kommandozeile steht. Falls nicht, führen Sie den Aktivierungsbefehl erneut aus: <code>.venv\Scripts\activate.</code>
<code>ImportError: DLL load failed...</code> oder ähnliche Fehler unter Windows	Ein häufiges Problem mit der Installation von <code>pywin32</code> , einer wichtigen Windows-Bibliothek, die von Jupyter benötigt wird.	<ol style="list-style-type: none"> Stellen Sie sicher, dass Ihre virtuelle Umgebung aktiv ist. Führen Sie den Befehl <code>pip install --upgrade pywin32</code> aus, um die Bibliothek zu reparieren.

Problem / Fehlermeldung (Was Sie sehen)	Mögliche Ursache (Warum es passiert)	Lösung (Was Sie tun können)
Kernel Error oder der Status "Kernel starting, please wait..." ändert sich nicht	Die Verbindung zwischen der Browser-Oberfläche und dem Python-"Gehirn" (dem Kernel) ist gestört. Dies kann viele Ursachen haben.	<ol style="list-style-type: none"> 1. Der einfachste erste Schritt: Klicken Sie im JupyterLab-Menü auf "Kernel" -> "Restart Kernel...". 2. Wenn das nicht hilft, schliessen Sie JupyterLab im Terminal (mit der Tastenkombination Strg + C) und starten Sie es mit <code>jupyter-lab neu</code>.
Permission denied (Zugriff verweigert) bei der Installation von Paketen	Sie versuchen, Pakete an einem systemweiten Ort zu installieren (z. B. in <code>C:\Program Files</code>), für den Sie keine Schreibrechte haben.	Dies ist genau das Problem, das virtuelle Umgebungen lösen! Stellen Sie sicher, dass Ihre <code>venv</code> aktiv ist (<code>(.venv)</code> muss sichtbar sein). Dadurch wird sichergestellt, dass alle Pakete lokal in Ihren Projektordner installiert werden, wo Sie die vollen Rechte haben.