

EIN WEBBASIERTES MULTIPLE-KERNEL-SYSTEM FÜR DIE AUTOMATISIERTE ERSTELLUNG VON ANALYTISCHEN BERICHTEN.

Dr. Alexander Wagner

AGENDA

- **Einführung**
- **Das Ziel des Projektes**
- **Das Konzept, die Struktur und die Funktionalitäten des Systems**
- **Analytischer Bericht**
- **Die System-Funktionen**
- **Prozessablauf der automatisierten Erstellung des analytischen Berichts**
- **Das System GUI mit Steuerelementen**
- **GitHub Repository des Systems**
- **Content des GitHub-Repositorys des Systems**
- **Web-App des Systems**
- **Bereitstellen (Deploy) der Web-App in Streamlit Community Cloud**
- **Weiterentwicklung des Systems**
- **Zusammenfassung**
- **Folgerung**

Einführung

Diese Arbeit gehört zum Forschungsgebiet „Automatisierte Erstellung von analytischen Berichten als webbasierte Applikationen“. Das Ziel des Projektes besteht in der Automatisierung von Routineoperationen zur Erstellung von analytischen Berichten über die Standardisierung der Prozessabläufe des Datenmanagements bis hin zur Ablage des Berichtes, der Software und Daten in der Web-Cloud und der Verwaltung des Client-Cloud Prozesses. Zur Anwendung kommt ein breites Spektrum an Werkzeugen der Datenverarbeitung, der Softwareintegration und der Steuerung von IT-Prozessen.

Schlüsselwörter: Automatisierte Erstellung von analytischen Berichten, Python, Streamlit, Streamlit.io-Cloud, WebApp, SAS, Autoit, MS OFFICE.



Das Ziel des Projektes

Das übergreifende Ziel dieses Projektes ist es, über die Implementierung einer automatisierten Erstellung von analytischen Berichten eine effiziente Steuerung der Kaskade aller Datenverarbeitungsschritte, die abschließende Ablage des Berichtes in der Web-Cloud und die Verwaltung des Client-Cloud Prozesses zu ermöglichen.

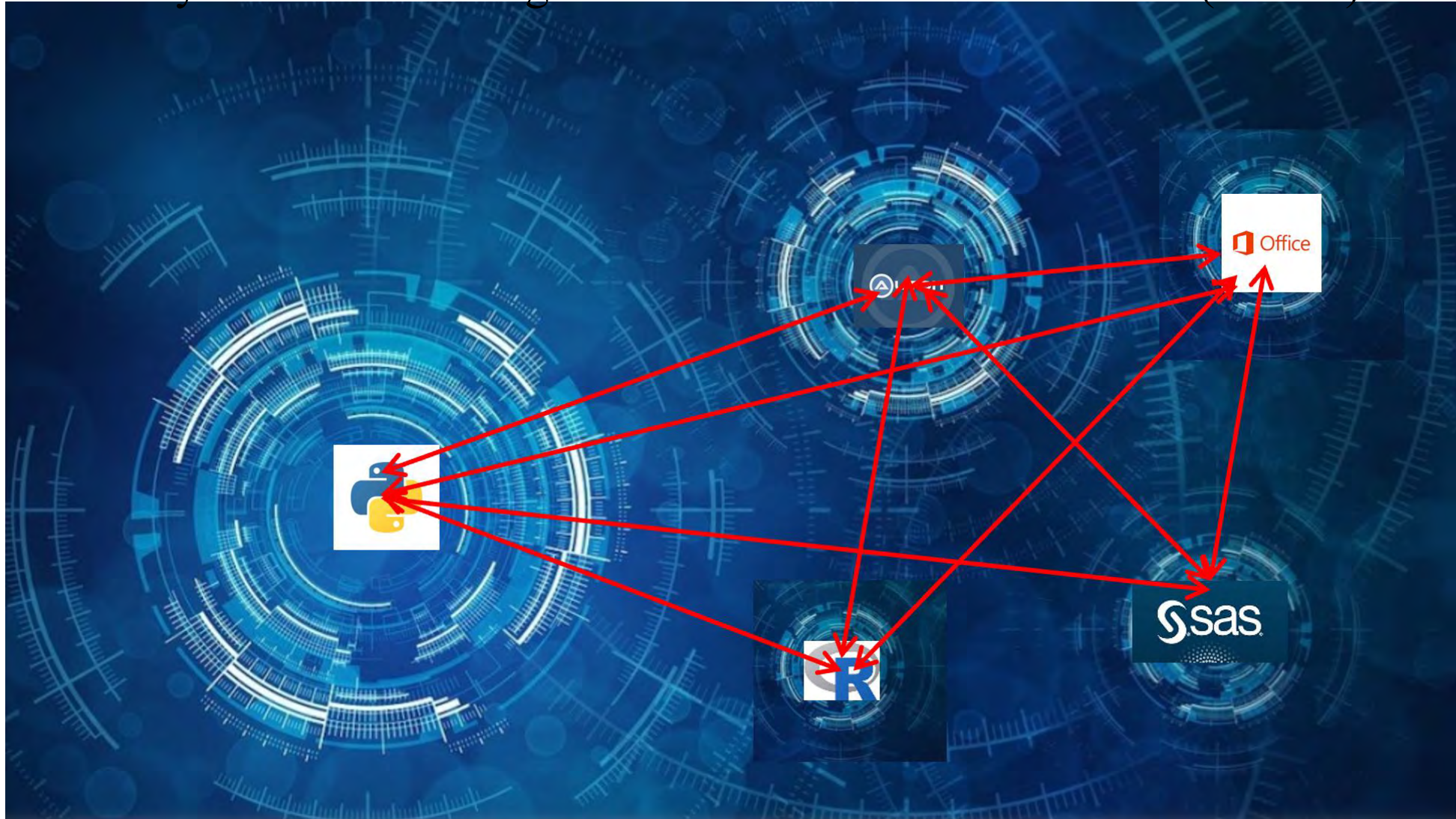
Die erste Aufgabe ist daher, alle Berichtsinformationen (Texte, Tabellen, Grafiken) in einem flexiblen Informationssystem/GitHub Repository zu organisieren. Die zweite Aufgabe besteht darin, alle notwendigen analytischen und Daten-Management-Verfahren und Methoden in einem speziellen Programm-Tool zu integrieren in Form einer Projekt-GUI. In der dritten Phase wird der Bericht in der Web-Cloud abgelegt und die Verwaltung des Client-Cloud Prozesses durchgeführt.

Dieses Projekt ist eine Fortsetzung verschiedener früherer Projekte.



Das Konzept, die Struktur und die Funktionalitäten des Systems

Das System ist eine Integration unterschiedlicher Software (Kernel)



Analytischer Bericht


Der analytische Bericht besteht aus den folgenden Standardelementen, und zwar:

- Frontseite
- Inhaltsverzeichnis
- Überschriften und Fußnoten
- Texte
- Tabellen
- Grafiken
- Listen

Das System entwickelt alle diese Komponenten mit Hilfe der Python DOCx-Bibliothek, SAS ODS und speziellen Autoit-Funktionen. Der Bericht wird in Form eines MS-WORD- und PDF-Dokumentes erstellt und im Repositoryum GitHub abgelegt

Tabellen und Graphiken. Beispiel: DOCx-Tabelle

Die Tabellen werden im Regelfall mit Hilfe von Python DOCx-Bibliothek erstellt; Tabellen, die mit SAS Proc Report erzeugt werden, können ebenfalls problemlos in den Bericht integriert werden. Die Erstellung von Graphiken wiederum erfolgt mit Hilfe von Python-Bibliotheken, SAS/Graph und SAS® 9.4 ODS Graphics Prozeduren, Funktionen R-System, etc.

	<i>Gesamt Population</i>	<i>Altersgruppen</i>				
	Median	0-14	15-24	25-54	55-64	65+
Aruba	39.3	17.64	12.78	41.72	14.28	13.59
Australia	38.7	17.8	12.79	41.45	11.83	16.14
Austria	44.0	14.01	11.07	42.42	13.23	19.26
Azerbaijan	31.3	22.95	14.84	45.39	10.17	6.64
Bahamas	32.0	22.55	16.4	44.14	9.16	7.75
The Bahrain	32.3	19.08	15.65	56.04	6.28	2.95
Bangladesh	26.7	27.76	19.36	39.73	6.93	6.23
Barbados	38.6	17.97	12.74	44.06	13.43	11.81
Belarus	40.0	15.78	10.29	44.76	14.21	14.95
Belgium	41.4	17.16	11.34	40.05	12.86	18.58

© Dr. Alexander Wagner. All Rights Reserved

DOCx-Tabelle, Python-script (Fragment)

```
doc.add_heading("Tabelle №"+str(k+1), 2)
doc.add_heading("Altersverteilung für ausgewählte Länder nach
WHO: " + str(country.values[k, 1]), 3)

t = doc.add_table(n_rows+2, n_cols, style="Table Grid")
set_repeat_table_header(t.rows[0])
for j in range(n_cols):
    if include_index:
        ha=1
    else:
        t.cell(0,j).text = df3.columns[j]
for i in range(2, n_rows+2):
    for j in range(n_cols):
        if include_index:
            t.cell(i, j).text = str(df3.values[i-2,j])
            t.cell(i, j).paragraphs[0].paragraph_format.alignment =
                WD_TABLE_ALIGNMENT.RIGHT
        else:
            t.cell(i, j).text = str(df3.values[i,j])
t.cell(i, 0).paragraphs[0].paragraph_format.alignment =
    WD_TABLE_ALIGNMENT.LEFT
```

```
t.allow_autofit = True

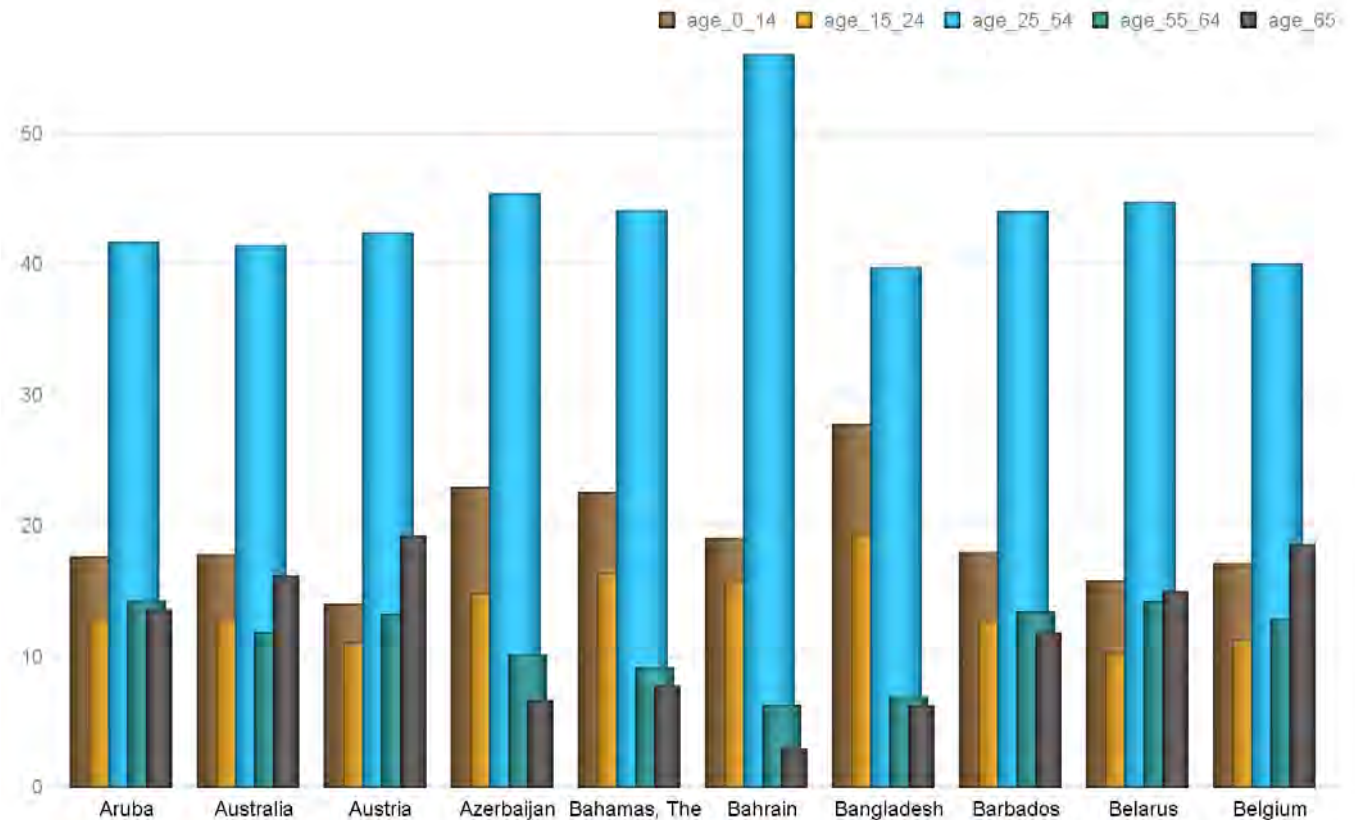
t.style = 'Colorful Shading Accent 1'
column_names0 = ["Country", "Median"]
column_names = ["0-14", "15-24", "25-54", "55-64", "65+"]
heading_cells = t.rows[0].cells
for j in range(0, 1):
    heading_cells[j].text = column_names0[j]
    heading_cells[j].width = widths[j]
heading_cells = t.rows[1].cells
for j in range(0, len(column_names)):
    heading_cells[j+2].text = column_names[j]
    heading_cells[j].width = widths[j]
cell = t.cell(1, j+2)
cell.paragraphs[0].alignment =
    WD_PARAGRAPH_ALIGNMENT.CENTER
cell.paragraphs[0].alignment =
    WD_PARAGRAPH_ALIGNMENT.CENTER
paragraph = cell.paragraphs[0]
run.add_picture("..\INDATEN\who-logo-png.png", Cm(1), Gm(1))
```


Beispiel: SAS-Graphik

Ein SAS-Macro für die Erstellung der Grafiken

```
%MACRO GRAPX(G=);  
  Title;  
  %COLOR(K=&G);  
  ods graphics / reset width=8in height=5in noscale noborder  
  imagename="GRAPH&G";  
  proc sgplot data=DATEN.DEMOGRA(where=(GR=&G)) nowall  
  noborder;  
    styleattrs datacolors=(&STRING);  
    vbar country / response=age_0_14 nostatlabel barwidth=0.6  
    nooutline dataskin=pressed  
    discreteoffset=-0.15 baselineattrs=(thickness=0);  
    vbar country / response=age_15_24 nostatlabel barwidth=0.5  
    nooutline dataskin=pressed  
    discreteoffset=-0.05 baselineattrs=(thickness=0);  
    vbar country / response=age_25_54 nostatlabel barwidth=0.4  
    nooutline dataskin=pressed  
    discreteoffset=0.05 baselineattrs=(thickness=0);  
    vbar country / response=age_55_64 nostatlabel barwidth=0.3  
    nooutline dataskin=pressed  
    discreteoffset=0.15 baselineattrs=(thickness=0);  
    vbar country / response=age_65 nostatlabel barwidth=0.2  
    nooutline dataskin=pressed  
    discreteoffset=0.25 baselineattrs=(thickness=0);  
    keylegend / location=outside position=topright noborder  
    valueattrs=(size=8 color=gray);  
    xaxis fitpolicy=thin display=(nolabel noticks noline)  
    valueattrs=(size=8 color=BLACK);  
    yaxis grid display=(noline noticks nolabel noline)  
    valueattrs=(size=8 color=gray);  
  run;
```

```
%MEND GRAPX;
```



Die System-Funktionen

Das System realisiert die folgenden Funktionen:

- Mit Hilfe von Python, SAS, R, etc. werden Tabellen und Grafiken für die Integration in den Bericht erstellt. Alle diese Berichtselemente werden lokal in einem speziellem Projekt-Directory abgelegt.
- Die vorgefertigten Text-Abschnitte für die Integration in den Bericht werden in einem speziellen Projekt-Directory als Word-Dokumente abgelegt.
- Mit Hilfe der Python DOCx-Bibliothek wird der Bericht erstellt.
- Der Bericht wird zunächst lokal auf einem speziellem Projekt-Directory abgelegt, danach in das ein GitHub Repository gepusht.
- Die Korrektur bzw. Nachbesserung der Programm-Codes und Berichts wird in der Regel lokal im GUI-Fenster vorgenommen und danach nach GitHub gepusht.
- Die spezielle Autoit UDF und Python Def realisieren mit Hilfe von speziellen Autoit Steuerelementen (Buttons, Pop- und Contents-Menu) die Download- bzw. Push/Upload Daten Transfer-Prozesse und Update des Systems.

Prozessablauf der automatisierten Erstellung des analytischen Berichts

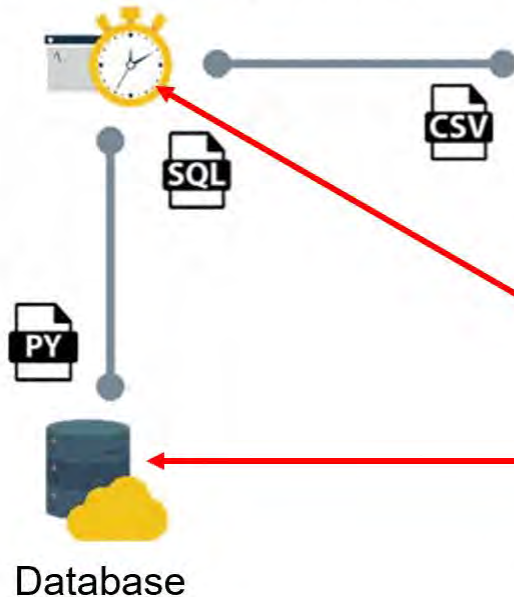
Der Prozessablauf für die Erstellung des analytischen Berichts erfolgt vollautomatisch. Durch Standard Point-and-Click Manipulationen wird der gesamte Prozess mit Hilfe von GUI-Elementen gesteuert. Die nachfolgende Folie stellt den Prozessablauf⁺ der automatisierten Erstellung des analytischen Berichts dar.

Der Prozessablauf besteht aus den folgenden Etappen bzw. Elementen:

- System-GUI als Kommando-Zentrale
- Data-Pushing
- Deployment der Web-App
- GitHub-Repositorium
- Streamlit.io-Cloud
- Produktion des Systems seitens der Endanwender

Schematische Darstellung des Anwendungsbetriebs in Web-Plattformen

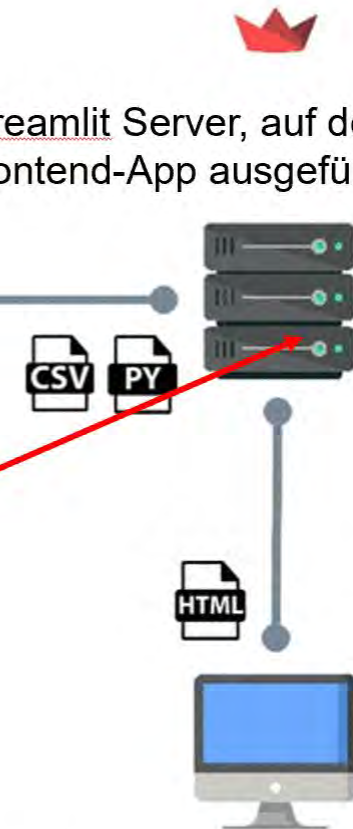
Hosten Streamlit-App auf Streamlit.io . Es wird den Code sowie den Datensatz aus dem Github Repository abrufen



GitHub Hostet den Datensatz und Python-Code für Streamlit Frontend



Streamlit Server, auf dem die Frontend-App ausgeführt wird



Erstellung des Front-End-Dashboard auf Python mit der Streamlit-Bibliothek

Der Endbenutzer greift über einen vom Streamlit-Server generierten Weblink auf das Dashboard zu


Das System GUI mit Steuerelementen

Ein webbasiertes Multiple-Kernel-System für die automatisierte Erstellung von analytischen Berichten

System-Konzept Start Bericht Open Bericht WEB APP-Bericht GitHubRepo

PARADOX GmbH Künstliche Intelligenz Dashboard. Demo-Version 3.0, Stand: 30. März 2023

Script WebAPP Script Bericht GitHubContent



Dateien in Github Repository

Datei	Download Url	Size	Date
<input type="checkbox"/> A%2B%2BNostalgio8...	https://raw.githubusercontent.com/av3wagner...	2984008	
<input type="checkbox"/> AWagner.JPG	https://raw.githubusercontent.com/av3wagner...	19453	
<input type="checkbox"/> AWbrigade.jpg	https://raw.githubusercontent.com/av3wagner...	23684	
<input type="checkbox"/> AWprofil.jpg	https://raw.githubusercontent.com/av3wagner...	206998	
<input type="checkbox"/> AutoitGui2.jpg	https://raw.githubusercontent.com/av3wagner...	121729	
<input type="checkbox"/> AutoitGui3.jpg	https://raw.githubusercontent.com/av3wagner...	86371	
<input type="checkbox"/> AutoitGui4.jpg	https://raw.githubusercontent.com/av3wagner...	87631	
<input type="checkbox"/> AutoitGui5.jpg	https://raw.githubusercontent.com/av3wagner...	74766	
<input type="checkbox"/> AutoitGuiBericht.jpg	https://raw.githubusercontent.com/av3wagner...	78369	
<input type="checkbox"/> Demogra.csv	https://raw.githubusercontent.com/av3wagner...	11060	
<input type="checkbox"/> DrWagner.md	https://raw.githubusercontent.com/av3wagner...	1784	
<input type="checkbox"/> KI.jpg	https://raw.githubusercontent.com/av3wagner...	27884	
<input type="checkbox"/> KI3.jpg	https://raw.githubusercontent.com/av3wagner...	7291	
<input type="checkbox"/> KSFE2023.docx	https://raw.githubusercontent.com/av3wagner...	1489116	
<input type="checkbox"/> KSFE2023.pdf	https://raw.githubusercontent.com/av3wagner...	2345941	
<input type="checkbox"/> KSFE2023F.docx	https://raw.githubusercontent.com/av3wagner...	1489116	
<input type="checkbox"/> KSFE2023F.pdf	https://raw.githubusercontent.com/av3wagner...	2345941	
<input type="checkbox"/> PostFooter.docx	https://raw.githubusercontent.com/av3wagner...	1491411	
<input type="checkbox"/> PostFooter.pdf	https://raw.githubusercontent.com/av3wagner...	2515708	
<input type="checkbox"/> README.md	https://raw.githubusercontent.com/av3wagner...	269	
<input type="checkbox"/> Welcome.jpg	https://raw.githubusercontent.com/av3wagner...	6602	
<input type="checkbox"/> requirements.txt	https://raw.githubusercontent.com/av3wagner...	24	
<input type="checkbox"/> streamlitV01.py	https://raw.githubusercontent.com/av3wagner...	14442	
<input type="checkbox"/> streamlit_app.py	https://raw.githubusercontent.com/av3wagner...	1223	
<input type="checkbox"/> streamlit_appV01.py	https://raw.githubusercontent.com/av3wagner...	7655	

Dateien auf lokaler Disk

Datei	Download Url	Size	Date
<input type="checkbox"/> KSFE2023.docx	C:\PROJEKTE2023\ALMATY2023\PushData\KSFE...	1454 KB	06:48:03 07.03.2023
<input type="checkbox"/> KSFE2023.pdf	C:\PROJEKTE2023\ALMATY2023\PushData\KSFE...	2290 KB	06:48:03 07.03.2023
<input type="checkbox"/> KSFE2023F.docx	C:\PROJEKTE2023\ALMATY2023\PushData\KSFE...	1454 KB	15:52:19 21.02.2023
<input type="checkbox"/> PostFooter.docx	C:\PROJEKTE2023\ALMATY2023\PushData\Post...	1456 KB	12:01:24 22.02.2023
<input type="checkbox"/> PostFooter.pdf	C:\PROJEKTE2023\ALMATY2023\PushData\Post...	2457 KB	12:01:23 22.02.2023

© PARADOX GmbH. Ein webbasiertes Multiple-Kernel-System für die automatisierte Erstellung von analytischen Berichten V3.0 (November 2022/März 2023), entwickelt von Dr. Alexander Wagner

+

●

GitHub Repositorium des Systems

Alle Programme und Daten des Systems liegen im GitHub-Repositorium. Mit Hilfe von GUI-Buttons und **Context Menus** kann man die Dateien hoch- und herunterladen, die WebApp starten, etc.

The screenshot displays a GitHub repository page for 'av3wagner/streamlit-example'. The repository is a fork of 'streamlit/streamlit-example'. The main content area shows a list of files and folders, including 'data', 'A++Nostalgi08.html', 'AWagner.JPG', 'AWbrigade.jpg', 'AWprofil.jpg', 'AppTest', 'AppTest2', 'AutoitGui2.jpg', 'AutoitGui3.jpg', 'AutoitGui4.jpg', 'AutoitGui5.jpg', 'AutoitGuiBericht.jpg', 'Demogra.csv', 'DrWagner.md', 'K1.jpg', 'K13.jpg', 'KSFE2023', and 'KSFE2023.docx'. A red arrow points to the 'GitHubRepo' button in the top navigation bar. The right sidebar shows repository statistics: 0 stars, 0 watching, 11.7k forks, and a language distribution chart showing HTML (98.7%), Python (1.1%), and Jupyter Notebook (0.2%).

File/Folder	Action	Time
data	Add files via upload	3 weeks ago
A++Nostalgi08.html	Add files via upload	last month
AWagner.JPG	Add files via upload	2 months ago
AWbrigade.jpg	Add files via upload	last month
AWprofil.jpg	Add files via upload	last month
AppTest	Update AppTest	2 months ago
AppTest2	Update AppTest2	2 months ago
AutoitGui2.jpg	Add files via upload	last month
AutoitGui3.jpg	Add files via upload	last month
AutoitGui4.jpg	Add files via upload	last month
AutoitGui5.jpg	Add files via upload	last month
AutoitGuiBericht.jpg	Add files via upload	last month
Demogra.csv	Add files via upload	3 weeks ago
DrWagner.md	Add files via upload	last month
K1.jpg	Add files via upload	2 months ago
K13.jpg	Add files via upload	2 months ago
KSFE2023	Update KSFE2023	2 months ago
KSFE2023.docx	Add files via upload	17 hours ago

+

●

Content des GitHub-Repositorys des Systems

Der Button „GitHubContent“ öffnet das System des Repository-Content auf GUI.

Ein webbasiertes Multiple-Kernel-System für die automatisierte Erstellung von analytischen Berichten

System-Konzept Start Bericht Open Bericht WEB APP-Bericht GitHubRepo PARADOX GmbH Künstliche Intelligenz Dashboard, Demo-Version 3.0, Stand: 30. März 2023 Script WebAPP Script Bericht **GitHubContent**

api.github.com/repos/av3wagner/st X

https://api.github.com/repos/av3wagner/streamlit-example/content 80%

JSON Rohdaten Kopfzeilen

Speichern Kopieren Alle einklappen Alle ausklappen JSON durchsuchen

```
{
  "html": "https://github.com/av3wagner/streamlit-example/blob/master/KSFE2023"
}
16:
{
  "name": "KSFE2023.docx",
  "path": "KSFE2023.docx",
  "sha": "a8d13903bb62667b6e80d0026f55c8c826c758f5",
  "size": 1489221,
  "url": "https://api.github.com/repos/av3wagner/streamlit-example/contents/KSFE2023.docx?ref=master",
  "html_url": "https://github.com/av3wagner/streamlit-example/blob/master/KSFE2023.docx",
  "git_url": "https://api.github.com/repos/av3wagner/streamlit-example/git/blobs/a8d13903bb62667b6e80d0026f55c8c826c758f5",
  "download_url": "https://raw.githubusercontent.com/av3wagner/streamlit-example/master/KSFE2023.docx",
  "type": "file",
  "_links": {
    "self": "https://api.github.com/repos/av3wagner/streamlit-example/contents/KSFE2023.docx?ref=master",
    "git": "https://api.github.com/repos/av3wagner/streamlit-example/git/blobs/a8d13903bb62667b6e80d0026f55c8c826c758f5",
    "html": "https://github.com/av3wagner/streamlit-example/blob/master/KSFE2023.docx"
  }
}
17:
{
  "name": "KSFE2023.pdf",
  "path": "KSFE2023.pdf",
  "sha": "bac71b95c1b587e38ad9c79ff541991b7a124c74",
  "size": 2345370,
  "url": "https://api.github.com/repos/av3wagner/streamlit-example/contents/KSFE2023.pdf?ref=master",
  "html_url": "https://github.com/av3wagner/streamlit-example/blob/master/KSFE2023.pdf",
  "git_url": "https://api.github.com/repos/av3wagner/streamlit-example/git/blobs/bac71b95c1b587e38ad9c79ff541991b7a124c74"
}
```

© PARADOX GmbH. Ein webbasiertes Multiple-Kernel-System für die automatisierte Erstellung von analytischen Berichten V3.0 (November 2022/März 2023), entwickelt von Dr. Alexander Wagner

Herunterladen des Berichts vom GitHub-Repo ins GUI-Fenster

Ein webbasiertes Multiple-Kernel-System für die automatisierte Erstellung von analytischen Berichten

PARADOX GmbH Künstliche Intelligenz Dashboard, Demo-Version 3.0, Stand: 31. Januar 2023

Projektinitiative
Ein Web- und Cloudbasiertes Multiple-Kernel
Eco-System für die automatisierte Erstellung
von analytischen Berichten

BERLIN
2023

Daten in GitHub Repository

Datei	Download URL	Size	Date
A%2B%2BNostalgi08...	https://raw.githubusercontent.com/av3wagner...	2984008	
AWagner.JPG	https://raw.githubusercontent.com/av3wagner...	19457	
AWbrigde.jpg	https://raw.githubusercontent.com/av3wagner...	23684	
AWprofil.jpg	https://raw.githubusercontent.com/av3wagner...	206998	
AutoiGui2.jpg	https://raw.githubusercontent.com/av3wagner...	121729	
AutoiGui3.jpg	https://raw.githubusercontent.com/av3wagner...	86371	
AutoiGui4.jpg	https://raw.githubusercontent.com/av3wagner...	87631	
AutoiGui5.jpg	https://raw.githubusercontent.com/av3wagner...	74766	
AutoiGuiBericht.jpg	https://raw.githubusercontent.com/av3wagner...	78369	
Demogra.csv	https://raw.githubusercontent.com/av3wagner...	11060	
DWagner.md	https://raw.githubusercontent.com/av3wagner...	1784	
KI.jpg	https://raw.githubusercontent.com/av3wagner...	27884	
KI3.jpg	https://raw.githubusercontent.com/av3wagner...	7291	
KSE2023.docx	https://raw.githubusercontent.com/av3wagner...	1489116	160810-21.02.2023
KSE2023.pdf	https://raw.githubusercontent.com/av3wagner...	2345941	
KSE2023F.pdf	https://raw.githubusercontent.com/av3wagner...	1489116	
KSE2023F.pdf	https://raw.githubusercontent.com/av3wagner...	2345941	
README.md	https://raw.githubusercontent.com/av3wagner...	269	
Welcome.jpg	https://raw.githubusercontent.com/av3wagner...	6602	
requirements.txt	https://raw.githubusercontent.com/av3wagner...	24	
streamlit_v01.py	https://raw.githubusercontent.com/av3wagner...	14442	
streamlit_app.py	https://raw.githubusercontent.com/av3wagner...	1223	
streamlit_app_v0.py	https://raw.githubusercontent.com/av3wagner...	7865	
streamlit_app_v02.py	https://raw.githubusercontent.com/av3wagner...	10424	

Daten auf lokaler Disk

Datei	Download URL	Size	Date
A%2B%2BNostalgi08...	https://raw.githubusercontent.com/av3wagner...	2984008	
AWagner.JPG	https://raw.githubusercontent.com/av3wagner...	19457	
AWbrigde.jpg	https://raw.githubusercontent.com/av3wagner...	23684	
AWprofil.jpg	https://raw.githubusercontent.com/av3wagner...	206998	
AutoiGui2.jpg	https://raw.githubusercontent.com/av3wagner...	121729	
AutoiGui3.jpg	https://raw.githubusercontent.com/av3wagner...	86371	
AutoiGui4.jpg	https://raw.githubusercontent.com/av3wagner...	87631	
AutoiGui5.jpg	https://raw.githubusercontent.com/av3wagner...	74766	
AutoiGuiBericht.jpg	https://raw.githubusercontent.com/av3wagner...	78369	
Demogra.csv	https://raw.githubusercontent.com/av3wagner...	11060	
DWagner.md	https://raw.githubusercontent.com/av3wagner...	1784	
KI.jpg	https://raw.githubusercontent.com/av3wagner...	27884	
KI3.jpg	https://raw.githubusercontent.com/av3wagner...	7291	
KSE2023.docx	https://raw.githubusercontent.com/av3wagner...	1489116	
KSE2023.pdf	https://raw.githubusercontent.com/av3wagner...	2345941	
KSE2023F.pdf	https://raw.githubusercontent.com/av3wagner...	1489116	
KSE2023F.pdf	https://raw.githubusercontent.com/av3wagner...	2345941	
README.md	https://raw.githubusercontent.com/av3wagner...	269	
Welcome.jpg	https://raw.githubusercontent.com/av3wagner...	6602	
requirements.txt	https://raw.githubusercontent.com/av3wagner...	24	
streamlit_v01.py	https://raw.githubusercontent.com/av3wagner...	14442	
streamlit_app.py	https://raw.githubusercontent.com/av3wagner...	1223	
streamlit_app_v0.py	https://raw.githubusercontent.com/av3wagner...	7865	
streamlit_app_v02.py	https://raw.githubusercontent.com/av3wagner...	10424	

Daten in GitHub Repository

Datei	Download URL	Size	Date
A%2B%2BNostalgi08...	https://raw.githubusercontent.com/av3wagner...	2984008	
AWagner.JPG	https://raw.githubusercontent.com/av3wagner...	19457	
AWbrigde.jpg	https://raw.githubusercontent.com/av3wagner...	23684	
AWprofil.jpg	https://raw.githubusercontent.com/av3wagner...	206998	
AutoiGui2.jpg	https://raw.githubusercontent.com/av3wagner...	121729	
AutoiGui3.jpg	https://raw.githubusercontent.com/av3wagner...	86371	
AutoiGui4.jpg	https://raw.githubusercontent.com/av3wagner...	87631	
AutoiGui5.jpg	https://raw.githubusercontent.com/av3wagner...	74766	
AutoiGuiBericht.jpg	https://raw.githubusercontent.com/av3wagner...	78369	
Demogra.csv	https://raw.githubusercontent.com/av3wagner...	11060	
DWagner.md	https://raw.githubusercontent.com/av3wagner...	1784	
KI.jpg	https://raw.githubusercontent.com/av3wagner...	27884	
KI3.jpg	https://raw.githubusercontent.com/av3wagner...	7291	
KSE2023.docx	https://raw.githubusercontent.com/av3wagner...	1489116	
KSE2023.pdf	https://raw.githubusercontent.com/av3wagner...	2345941	
KSE2023F.pdf	https://raw.githubusercontent.com/av3wagner...	1489116	
KSE2023F.pdf	https://raw.githubusercontent.com/av3wagner...	2345941	
README.md	https://raw.githubusercontent.com/av3wagner...	269	
Welcome.jpg	https://raw.githubusercontent.com/av3wagner...	6602	
requirements.txt	https://raw.githubusercontent.com/av3wagner...	24	

- 1 – Selektion des Dokuments
- 2 – Mausklick mit rechte Taste auf Menu-Option „Upload“
- 3 – Ausgewähltes Word-Dokument wurde runtergeladen, geöffnet und auf GUI platziert

Web-App des Systems

Das System verwendet die Streamlit Community Cloud, eine Open-Source-Plattform für die Community, um Streamlit-Apps bereitzustellen und freizugeben. Der Prozess zum Bereitstellen der Streamlit-App in Streamlit Cloud funktioniert in drei einfachen Schritten:

1. Generierung der requirements.txt, die alle Abhängigkeiten für das Projekt auflistet. Dies wird benötigt, um Streamlit mitzuteilen, welche Bibliotheken installiert werden müssen, um den Code auszuführen. Hierzu kann man Pipregs verwenden, um die Datei basierend auf den Importen im Programm Code automatisch zu generieren.
2. Das Hosting des Codes erfolgt in einem öffentlichen GitHub-Repository. Hierbei ist sicherzustellen, dass das Repository die requirements.txt Datei enthält.
3. Registrieren an der share.streamlit.io.

Die nachfolgende Folie stellt das Bereitstellen (Deployment) der WebApp in der Streamlit Community Cloud dar.

Bereitstellen (Deployment) der Web-App in Streamlit Community Cloud

New app

Deploy a new app...

From existing repo

From sample app template

Deploy an app

Repository

[Paste GitHub URL](#)

lanchuong/repo

Branch

master

Main file path

app.py

[Advanced settings...](#)

Deploy!

Streamlit Web-App auf der GUI des Systems

Beim Klicken auf Button „WEB APP-Bericht“ öffnet das System Web-Applikation auf GUI.

The screenshot displays a web application interface with a dark header. The header text reads: "Ein webbasiertes Multiple-Kernel-System für die automatisierte Erstellung von analytischen Berichten" in yellow, and "PARADOX GmbH Künstliche Intelligenz Dashboard. Demo-Version 3.0, Stand: 31. Januar 2023" in green. A navigation bar below the header contains buttons for "System-Konzept", "Start Bericht", "Open Bericht", "WEB APP-Bericht" (highlighted with a red box and a red arrow), and "GitHub".

The main content area is divided into a left sidebar and a central panel. The sidebar, titled "Navigation", includes a list of menu items: "Einführung", "Bericht Visualisierung" (selected with a red dot), "Hochladen von Dateien", "Herunterladen von Dateien", "Beispiel 18 Modellen", and "Über uns". Below this is a "Pick a dataset" section with a dropdown menu showing "KSFE2023F.pdf".

The central panel features the Paradox GmbH logo and the title "Visualisierung des Berichts in PDF-Form" with document icons. Below the title is a PDF viewer showing a page with the text: "Ein Web- und Cloudbasiertes Multiple-Kernel Eco-System für die automatisierte Erstellung von analytischen Berichten". The viewer includes a search bar, a page indicator "1 von 50", and a zoom level of "90%". At the bottom of the page, the text "BERLIN" and "2023" is visible. A red crown icon is located in the bottom right corner of the page.

At the bottom of the screenshot, a copyright notice reads: "© PARADOX GmbH, RealWordBericht2-System Version 1.0 (Mai 2022/Oktober 2022), entwickelt von Dr. Alexander Wagner".

Weiterentwicklung des Systems

Das System funktioniert als Streamlit.io Web-Applikation und konnte mit seiner Effizienz vielfach überzeugen. Dennoch bestehen Ansatzpunkte für Nachbesserung und Weiterentwicklung. Die folgenden Erweiterungs- und Optimierungspotentiale werden priorisiert: •

- Verwendung des funktional und technologisch überlegenen Django-System (<https://www.djangoproject.com/>) statt Streamlit.
- Verwendung von funktional und technologisch überlegenen Cloud-Systemen (Amazon EC2, Microsoft Azure Cloud, etc.) statt Streamlit.io
- Korrektur bzw. Nachbesserung des Berichts in Cloud von mehreren Anwendern gleichzeitig
- Verwendung der JSON Steuer/Config-Datei für voll automatisierte Berichtserstellung
- Automatisierung des Uploads/Download Prozesses z.B. mit Hilfe von Windows-Task

ZUSAMMENFASSUNG



Es wurde ein webbasiertes Multiple-Kernel-System für die automatisierte Erstellung von analytischen Berichten entwickelt und in die Produktion gebracht.

Das System hat seine programm-technische und technologische Effizienz gezeigt, funktioniert robust und kann für die Erstellung der analytischen Berichten verschiedener Art verwendet werden.

Weiterentwicklung und Nachbesserung des Systems folgt.



FOLGERUNG



In Kürze startet ein neues nichtkommerzielles Projekt „Automatisierung der Wissenschaftliche Forschung im Gesundheitswesen, Medizin und Pharmabranche mit Hilfe von KI-Methoden“.

Das Projekt wird schätzungsweise 6-8 Monate dauern.

Wer jemand Interesse an einer spannenden hochintelligenten Arbeit mit abschließender wissenschaftlicher Publikation hat, dann herzlich Willkommen ins Team!



TEAM



**Dr. Alexander Wagner,
Teilnehmer an KSFE-2015**



+



o



.



VIELEN DANK!

Dr. Alexander Wagner
av3.wagner@gmail.com

Literatur

- [1] Autolt website. <https://www.autoitscript.com/site/>
- [2] User Defined Function. https://de.wikipedia.org/wiki/User_Defined_Function
- [3] Base SAS. SAS Notes and Concepts for ODS, The RTF Destination. +
https://support.sas.com/rnd/base/ods/templateFAQ/Template_rtf.html •
- [4] Lauren Haworth, Applying Microsoft Word Styles to ODS RTF Output.
<http://www2.sas.com/proceedings/sugi30/043-30.pdf>
- [5] Kirk Paul Lafler Output Delivery System (ODS) – Simply the Basics. Software Intelligence Corporation, Spring Valley, California.
<http://www.scsug.org/wp-content/uploads/2012/11/Output-Delivery-System-ODS-%E2%80%93-Simply-the-Basics-SCSUG-2012.pdf>
- [6] Carol Matthews, Elena Kalchenko, Pretty Please?! Making RTF Output “Pretty” with SAS. United Biosource Corporation, Blue Bell, Pennsylvania.
<https://www.pharmasug.org/proceedings/2013/IB/PharmaSUG-2013-IB08.pdf>
- [7] SAS® 9.4 ODS Graphics: Procedures Guide, Sixth Edition.
<http://support.sas.com/documentation/cdl/en/grstatproc/69716/HTML/default/viewer.htm#titlepage.htm>
- [8] python-docx 0.8.11 documentation. <https://python-docx.readthedocs.io/en/latest/>
- [9] python-docx-template. <https://docxtpl.readthedocs.io/en/latest/>

Literatur (Fortsetzung 1)

- [9] python-docx-template. <https://docxtpl.readthedocs.io/en/latest/>
- [10] Automate Word Document (.docx) With Python-docx And pywin32
- [11] Jay, Automate Word Document (.docx) With Python-docx And pywin32. <https://pythoninoffice.com/automate-docx-with-python/> + ●
- [12] Dr. Rik Voorhaar, How to edit Microsoft Word documents in Python. <https://www.rikvoorhaar.com/python-docx/>
- [13] PyGithub- documentation <https://pygithub.readthedocs.io/en/latest/introduction.html>
- [14] Abdou -Rockikz, Python- script- to -create - GitHub- repository. <https://iq.opengenus.org/create-github-repository-python/>
- [15] Code- for- How- to- Use- Github- API- in- Python- Tutorial. <https://www.thepythoncode.com/article/using-github-api-in-python>
- [16] Streamlit-Dokumentation: <https://docs.streamlit.io/>
- [17] Marc, Introduction to Streamlit: Building Interactive Web Apps using Python. <https://marccodess.medium.com/introduction-to-streamlit-building-interactive-web-apps-using-python-62c250ba858d>
- [19] Alan Jones, Streamlit from Scratch: Embedding Images, Video and Audio. <https://towardsdatascience.com/streamlit-from-scratch-embedding-images-video-and-audio-8b2e8b98fad4>

Literatur (Fortsetzung 2)

- [20] Navid Mashinchi, The Current State of COVID-19 From 3 Different Perspectives. <https://towardsdatascience.com/the-current-state-of-covid-19-from-3-different-perspectives-3fbaabcd0348>
- [21] Ponsriharini, Deploy a Machine Learning model using streamlit. <https://medium.com/featurepreneur/deploy-a-machine-learning-model-using-streamlit-a7ef163c19b0>
- [22] Sharad Mittal, Creating a Web app of a ML model using Streamlit. <https://sharadmittal.hashnode.dev/creating-a-web-app-of-a-ml-model-using-streamlit>.
- [23] Deliver Your Customer Behavior Analysis Report with Streamlit. <https://medium.com/octopus-id-data/deliver-your-customer-behavior-analysis-report-with-streamlit-36f052d0a43c>
- [24] A. Wagner, K.Krobot. Aktuariell diskontiertes Überleben aus webbasierten Rohdatenbanken, KSFE-2009. https://saswiki.de/display/KONFERENZEN/KSFE+2009?preview=/21267185/21267221/13.KSFE-2009-Wagner-Webbasierte_Rohdatenbanken.pdf#KSFE2009-DataMining/WebMining/Datenmanagement
- [25] A. Wagner, J. Holstiege. Ein SAS basiertes System zur automatisierten Auswertung und Berichterstellung von klinischen Studien, KDFE-2019. https://saswiki.de/display/KONFERENZEN/KSFE+2019?preview=/19726371/19726410/23_KSFE_2019_Wagner_-_Ein_SAS_basiertes_System_zur_automatisierten_Auswertung_und_Berichterstellung_von_klinischen_Studien.pdf