






































## Hardware Reparatur

Löten  4  
SMD Löten  3  
Feines Handwerk  5  
Exakt  4  
Kabelbau  4  
Manuals verstehen  5  
Level 1 Reparaturen  5  
Level 2 Reparaturen  5  
Level 3 Reparaturen  4








## Hardware Kenntnisse

Laptop  4  
Tablet  4  
Smartphone  4  
Desktop PC  4  
Display v TV  4  
Hi-Fi Audio  3  
Video  4  
Eventtechnik  2  
Projektor  2  
iMac  4  
iPod  4  
Kopfhörer  3  
Lautsprecher  4  
Car-Hi-Fi  3  
Smartwatch  3  
DVB Receiver  4  
Elektrotechnik Bauteile  3  
Drucker  4














## Arbeitsabläufe

Ersatzteilverwaltung  5  
Kostenvoranschläge erstellen  5  
Kundensupport  5  
Zusammenarbeit mit Kunden  5  
Garantieabgeltungen  5  
Fehlersuche  4  
Fehlerbehebung  4  
Refurbishment  4  
Automatisierung von Arbeit  3  
Sicherheitsbewusstsein  5

## Software Systeme

Manjaro/Arch Linux  4  
Debian/Ubuntu/Kali/Fedora Linux  4  
Apple iOS  4  
Mac OS und Mac OS ARM  4  
Apple iPad OS  4  
Android  4  
Manjaro ARM  4  
Windows  3

## Software Kenntnisse

Microsoft Office  4  
Google Dienste  4  
Jira/Confluence  3  
Citrix  2  
Umgang mit KI  4  
GitHub  3  
Asana  3  
Notion  3  
NeoVim  3  
Terminal  4  
SAP  3  
Bildbearbeitung  4  
MS Teams  1

## Erklärung

Grundkenntnisse  
Gutes theoretisches Wissen  
Erfahrung in Praktik und Theorie  
Gute praktische und theoretische Kenntnisse  
Sehr gute praktische und theoretische Kenntnisse

```

import os
import csv
from reportlab.lib.colors import HexColor
from reportlab.pdfgen import canvas
from reportlab.lib.pagesizes import A4
from reportlab.platypus import Paragraph
from reportlab.lib.styles import ParagraphStyle

width=logo_width,
text_width = c.stringWidth(name, "Helvetica-Bold", height=logo_height)
def add_header(c, width, height, margin):
    mask="auto",
    # Icons zeichnen
    # Überschrift und die Freizeitaktivitäten und das Python-Logo
    filename=os.path.join(path_to_icons_folder, "freizeitaktivitaeten.png")
    try:
        print(f"Fehler bei Icon '{icon_path}': {e}")
    except:
        pass
    for i in range(num_icons):
        c.drawString(margin, height - margin - 15, on_x_position = x_position)
        "Aaron Feldmann Skill Auflistung"
        text_width + i * (icon_params["width"]
def read_data_from_csv(file_path):
    line(margin, height - margin - 10, c.setFillColor(HexColor("#ADD8E6"))
    width - margin, height - margin - 10, text = (
    Liest die Daten aus einer CSV-Datei und gibt sie als Dictionary zurück. icon_x_position = 17, "erstellt. Den Source
    """
    y_position - 17, "erstellt. Den Source
    data = {}
    def draw_categories(
        with open(file_path, newline='datacodings_folder') width=width, height=height, margin=margin, column_width=width)
        reader = csv.DictReader(csvfile)
        stroke=0, style = styles["Normal"]
        for row in reader:
            fill=1, style.fontName = "Helvetica"
            category = row['Kategorie']
            c.drawString(margin, height - margin - 10, c.drawString(margin, height - margin - 10, c.drawString(margin, height - margin - 10,
            icon = row['Icon'].strip() if row['Icon'] else None
            c.drawImage( style.leading = 12
            value = row['Wert'].strip()
            icon_path, style.textColor = HexColor(
            ) if 'Wert' in row and row['Wert'] else None
            icon_x_position,
            if category not in data:
                y_position - 15, paragraph = Paragraph(text
                data[category] = [Kategorien links und rechts definieren
                width=icon_params["width"]
                data[category].append(
                height=icon_params["height"]
                left_categories = ["Hardware Kenntnisse",
                height=icon_params["height"]
                "Name": row['Name'],
                "Arbeitsabläufe", "Hardware-Reparatur"]
                "Icon": icon, right_categories = ["Software Systeme", )
                "Value": value
                "Programiersprachen", "Exzessiv Software"
            })
            print(f"Fehler bei Icon '{icon_path}': {e}")
            icon_params = set_icon_parameters(height=13, width=13, y_offset=3)
            verteilt auf mehrere Spalten
            for category, items in data.items():
                def add_explanation(c, data, margin, column_width, bottom
                # Bestimme Position basierend auf der Kategorie
                font_size = 8 # Schriftgröße
def set_icon_parameters(height=13, width=13, y_offset=3):
    category = "Python"
    y_position = y_position
    column_width = (width - 2
    Definiert Standardparameter für Icons
    explanation_x = margin + column_width
    margin = 10 # Abstand
    elif category in right_categories:
        explanation_y = bottom_margin + 200
    return {"height": height, "width": width, "x_offset": x_offset, "y_offset": y_offset}
    c.showPage() # Neue Seite
    x_position = margin
    c.setFont("Helvetica-Bold", 12)
    c.setFont("Courier", font_
    else:
        c.setFillColor(HexColor("#000000"))
        c.drawString(explanation_x, explanation_y, explanation)
def get_color_for_value(value):
    continue
    c.drawString(explanation_x, explanation_y, explanation)
    c.line(
    Gibt die entsprechende Farbe für einen Wert zurück
    explanation_x,
    with open(code_file, "r")
    """
    c.setFont("Helvetica-Bold", 12)
    explanation_y + 18,
    for line in file:
    if value == "1":
        return HexColor("#ADD8E6")
        c.drawString(x_position, y_position, line)
        x_position = margin
    elif value == "2":
        return HexColor("#87CEEB")
        y_position -= 15
        )
        # Mittelblau
        # Wechsle zu nächster
    elif value == "3":
        return HexColor("#4682B4")
        # Einträge in der Kategorie
        c.setFont("Helvetica", 10)
        if y_position < margin:
            items = for item in data.get("Erklärung", []):
                column_index +
            y_position = h
        elif value == "4":
            draw_item(c, item, x_position, y_position, item["Name"])
            y_position = h
            icons_folder, color=category_color_for_value(item["Value"])
        elif value == "5":
            y_position -= 20
            c.setFillColor(color)
            # Wechsle zu nächster
            return HexColor("#2E8B57")
            # Dunkelgrün
            c.rect(explanation_x, explanation_y, 225, 15, fill=1)
            c.showPage
        else:
            # Aktualisiere die Y-Positionen für die nächsten
            c.setFont("Helvetica", 10)
            c.showPage
            return HexColor("#FFFFFF")
            # Weiß
            left_categories = ["Hardware Kenntnisse", "Arbeitsabläufe", "Hardware-Reparatur"]
            right_categories = ["Software Systeme", "Programiersprachen", "Exzessiv Software"]
            y_position_left = y_position
            explanation_y -= 20
            column_index
            elif category in right_categories:
                y_position
def create_pdf(filename, data, icons_folder):
    y_offset = y_offset
    def add_logo_and_description(c, icons_folder, logo_path, name, path_to_icons_folder, "pl.png")
    Erstellt ein PDF mit den Daten aus der CSV-Datei, Icons und Python-Code.
    c.drawString(x_pos
    """
    def draw_item(c, item, x_position, y_position, icons_folder, description):
        # Python-Code
        c.drawString(x_pos
    c = canvas.Canvas(filename, pagesize=A4)
    """
    width, height = A4
    Zeichnet einen einzelnen Eintrag
    logo_path = os.path.join(path_to_icons_folder, "pl.png")
    script_dir = os.path.dirname(os
    margin = 50
    """
    name = item["Name"]
    logo_height = 150
    csv_file_path = os.path.join(s
    # Überschrift und Hauptinhalt
    icon_path = os.path.join(
    logo_x = width - margin
    icon_width = 50
    path.join(sc
    add_header(c, width, height, margin, icons_folder, item["Icon"])
    if item["Icon"] in left_categories:
        icon_path = os.path.join(
        path.join(sc
    draw_categories(c, data, icons_folder, width, height, margin, column_width)
    value = item["Value"]
    margin, (width - margin) / 2)
    # Logo hinzufügen
    add_explanation(c, data, margin, item["Value"], margin)
    try:
        code_file_path = os.path.join(
        create_pdf(output_pdf_path, da
    # Python-Code hinzufügen
    # Namen zeichnen
    c.drawImage(
    add_code_with_columns(c, code_file_path, column_width, column_index, path,
    c.setFont("Helvetica", 12)
    logo_x,
    c.save()
    c.drawString(x_position + 5, y_position, logo_yname)

```