Instructions for data staging

Step 1 – Need to install python and flask

- Go to https://www.python.org/downloads/
- Download **Python 3.10+ (recommended)**
- **IMPORTANT**:
  - ✔ Check **"Add Python to PATH"**
  - ✔ Click **Install Now**
- **Verify installation**:
  python --version
  pip –version

Step 2 - Create a Project Folder

mkdir app

cd app

Step 3 - Install Flask

pip install flask

Step 4 – Create Flask App with below code [Resume Analysis]

```
from flask import Flask, render_template, request, jsonify import requests

app = Flask(name)

AUTH_URL = "https://aa-community.cloud.automationanywhere.digital/v2/authentication" DEPLOY_URL = "https://aa-community.cloud.automationanywhere.digital/v3/automations/deploy"

USERNAME = "username" PASSWORD = "password"

FILE_ID = 100026284 RUN_AS_USER_ID = 100010000

def get_auth_token(): payload = { "username": USERNAME, "password": PASSWORD, "multipleLogin": False }

headers = {
  "Content-Type": "application/json"
}


response = requests.post(AUTH_URL, json=payload, headers=headers)


if response.status_code == 200:
  return response.json().get("token")
else:
  print("Auth Failed:", response.text)
  return None


def deploy_bot(token, job_desc, folder_path, email): headers = { "Content-Type": "application/json", "X-Authorization": token }

payload = {
  "fileId": FILE_ID,
  "runAsUserIds": [RUN_AS_USER_ID],
  "poolIds": [],
  "overrideDefaultDevice": False,
  "botInput": {
    "iStrJobDescription": {
      "type": "STRING",
      "string": job_desc
    },
    "iStrResumeFolderPath": {
      "type": "STRING",
      "string": folder_path
    },
    "iStrRequestorMailID": {
      "type": "STRING",
      "string": email
    }
  }
}
```

```
response = requests.post(DEPLOY_URL, json=payload, headers=headers)

return response.json(), response.status_code


@app.route("/", methods=["GET", "POST"]) def index(): if request.method == "POST": folder_path = request.form.get("folderPath")
job_desc_type = request.form.get("jobDescType") email = request.form.get("email") # ✅ NEW

  if job_desc_type == "text":
      job_desc = request.form.get("jobDescText")
  else:
      job_desc = request.form.get("jobDescFile")


  # Step 1: Authenticate
  token = get_auth_token()
  if not token:
      return jsonify({"error": "Authentication failed"}), 401


  # Step 2: Deploy Bot (email passed)
  deploy_response, status_code = deploy_bot(
      token,
      job_desc,
      folder_path,
      email
  )


  return jsonify({
      "status": "Bot deployment triggered",
      "response": deploy_response
  })

return render_template("index.html")


if name == "main": app.run(debug=True)
```

Step 4 – Create Index.html with below code

```
<!-- Bootstrap 5 CDN -->
<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.2/dist/css/bootstrap.min.css"
rel="stylesheet">

<style>
  body {
    background: linear-gradient(135deg, #e3f2fd, #f8f9fa);
    min-height: 100vh;
    display: flex;
```

```
        align-items: center;
        justify-content: center;
    }
    .card {
        border-radius: 15px;
        box-shadow: 0 10px 25px rgba(0,0,0,0.1);
    }
    .btn-primary {
        background: linear-gradient(90deg, #0078d4, #005a9e);
        border: none;
    }
    .status-message {
        margin-top: 15px;
        padding: 12px;
        border-radius: 8px;
        font-weight: 600;
        text-align: center;
    }
    .success {
        background: #e6fffa;
        color: #065f46;
    }
    .error {
        background: #fee2e2;
        color: #991b1b;
    }
</style>

<div class="row justify-content-center">
    <div class="col-md-7">
        <div class="card p-4">

            <h3 class="text-center mb-4">🤖 Smart Hire</h3>

            <form id="botForm" novalidate>

                <!-- Email -->
                <div class="mb-3">
                    <label class="form-label fw-bold">Email</label>
                    <input type="email"
```

```
          class="form-control"
          name="email"
          id="email"
          placeholder="Hr@example.com"
          required>
    <div class="invalid-feedback">
       Please enter a valid email address.
    </div>
</div>

<!-- Resume Folder -->
<div class="mb-3">
    <label class="form-label fw-bold">Resume Folder Path</label>
    <input type="text"
          class="form-control"
          name="folderPath"
          placeholder="C:\Resumes"
          required>
</div>



<!-- Job Description Type -->
<div class="mb-3">
    <label class="form-label fw-bold">Job Description Type</label>
    <div class="d-flex gap-4 mt-2">
       <div class="form-check">
          <input class="form-check-input" type="radio"
                name="jobDescType" value="text" id="jdText" checked>
          <label class="form-check-label" for="jdText">Text</label>
       </div>
       <div class="form-check">
          <input class="form-check-input" type="radio"
                name="jobDescType" value="file" id="jdFile">
          <label class="form-check-label" for="jdFile">File Path</label>
       </div>
    </div>
</div>

<!-- Job Description Text -->
<div class="mb-3" id="textDiv">
```

```
            <label class="form-label fw-bold">Job Description (Text)</label>
            <textarea class="form-control"
                      name="jobDescText"
                      rows="5"
                      placeholder="Paste job description here"></textarea>
          </div>

          <!-- Job Description File -->
          <div class="mb-3 d-none" id="fileDiv">
            <label class="form-label fw-bold">Job Description File Path</label>
            <input type="text"
                   class="form-control"
                   name="jobDescFile"
                   placeholder="C:\JD\job_description.txt">
          </div>

          <!-- Submit -->
          <div class="d-grid">
            <button type="submit" class="btn btn-primary btn-lg">
              🚀 Start Analysis
            </button>
          </div>
        </form>

        <div id="messageBox"></div>

      </div>
    </div>
  </div>
```

Step 5 – Python code for Interview

from flask import Flask, render_template, request, jsonify import pyodbc import requests import os

app = Flask(**name**)

def get_connection(): return pyodbc.connect(r"DRIVER={ODBC Driver 17 for SQL Server};" r"SERVER=IPADDCA1\SQLEXPRESS;" r"DATABASE=AA_Bounty;" r"Trusted_Connection=yes;" r"TrustServerCertificate=yes;")

CONTROL_ROOM_URL = "https://aa-community.cloud.automationanywhere.digital" AUTH_URL = f"{CONTROL_ROOM_URL}/v2/authentication" DEPLOY_URL = f"{CONTROL_ROOM_URL}/v3/automations/deploy"

AA_USERNAME = os.getenv("AA_USERNAME", "UserName") AA_PASSWORD = os.getenv("AA_PASSWORD", "Password")

FILE_ID = 100027336 RUN_AS_USER_ID = 100010000

def get_auth_token(): payload = { "username": AA_USERNAME, "password": AA_PASSWORD, "multipleLogin": False } headers = {"Content-Type": "application/json"} response = requests.post(AUTH_URL, json=payload, headers=headers)

```
if response.status_code == 200:
    return response.json().get("token")
else:
    raise Exception(f"AA Authentication Failed: {response.text}")


def get_questions(job_id): conn = get_connection() cursor = conn.cursor()

cursor.execute("""
    SELECT [Q1],[Q2],[Q3],[Q4],[Q5],
        [Q6],[Q7],[Q8],[Q9],[Q10]
    FROM [dbo].[QuestionTable]
    WHERE Job_ID = ?
""", job_id)

row = cursor.fetchone()
conn.close()

questions = []
if row:
    for i in range(1, 11):
```

```
        q = getattr(row, f"Q{i}")
        if q:
            questions.append({"no": f"Q{i}", "text": q})

return questions


def save_answer(candidate_id, job_id, qno, question, answer): conn = get_connection() cursor =
conn.cursor()

cursor.execute("""
    INSERT INTO CandidateAnswers
    (Candidate_ID, Job_ID, Question_No, Question, Answer)
    VALUES (?, ?, ?, ?, ?)
""", candidate_id, job_id, qno, question, answer)

conn.commit()
conn.close()


@app.route("/interview/int:job_id/int:candidate_id", methods=["GET", "POST"]) def interview(job_id,
candidate_id):

if request.method == "POST":
    data = request.json
    save_answer(
        candidate_id=candidate_id,
        job_id=job_id,
        qno=data["qno"],
        question=data["question"],
        answer=data["answer"]
    )
    return jsonify({"status": "saved"})

questions = get_questions(job_id)
return render_template(
    "interview.html",
    questions=questions,
    job_id=job_id,
    candidate_id=candidate_id
)
```

```
@app.route("/submit", methods=["POST"]) def submit(): data = request.json job_id = data["job_id"]
candidate_id = data["candidate_id"]

token = get_auth_token()

headers = {
   "X-Authorization": token,
   "Content-Type": "application/json"
}

payload = {
   "fileId": FILE_ID,
   "runAsUserIds": [RUN_AS_USER_ID],
   "poolIds": [],
   "overrideDefaultDevice": False,
   "botInput": {
      "iNumCandidateID": {
         "type": "NUMBER",
         "number": candidate_id
      },
      "iNumJobID": {
         "type": "NUMBER",
         "number": job_id
      }
   }
}

response = requests.post(DEPLOY_URL, json=payload, headers=headers)

if response.status_code not in [200, 201]:
   raise Exception(f"Bot Deploy Failed: {response.text}")

return jsonify({
   "status": "Bot Triggered Successfully",
   "aa_response": response.json()
})


if name == "main": app.run(debug=True)
```

Step 6 – Interview.html

```
<!-- AVATAR -->
<div class="avatar-zone">
  <img
    src="{{ url_for('static', filename='images/HR_Avatat.jpg') }}"
    alt="AI Interviewer"
    class="avatar-img"
  >
</div>

<!-- INTERVIEW CONTENT -->
<div class="content">
  <h2>AI Interviewer</h2>

  <div class="permission-warning" id="permissionWarning">
    Microphone access blocked! Please allow access.
  </div>

  <div class="status" id="status"></div>
  <div class="question" id="questionText"></div>

  <textarea id="answer" placeholder="Type or speak your answer..."></textarea>

  <div class="controls">
    <button class="btn-speak" onclick="speakQuestion()">🔊 Speak</button>
    <button class="btn-voice" onclick="startVoice()">🎙 Voice</button>
    <button class="btn-next" id="nextBtn" onclick="nextQuestion()">➡ Next</button>
    <button class="btn-submit" id="submitBtn" onclick="submitInterview()" style="display:none">
      ✔ Submit
    </button>
  </div>
</div>
```

Step 7 – Create SQL database with below table structure

USE [master] GO /****** Object: Database [AA_Bounty] Script Date: 1/10/2026 3:23:17 PM **/ CREATE DATABASE [AA_Bounty] CONTAINMENT = NONE ON PRIMARY (NAME = N'AA_Bounty', FILENAME = N'C:\Program Files\Microsoft SQL Server\MSSQL17.SQLEXPRESS\MSSQL\DATA\AA_Bounty.mdf' , SIZE = 8192KB , MAXSIZE = UNLIMITED, FILEGROWTH = 65536KB) LOG ON (NAME = N'AA_Bounty_log', FILENAME = N'C:\Program Files\Microsoft SQL**

Server\MSSQL17.SQLEXPRESS\MSSQL\DATA\AA_Bounty_log.ldf' , SIZE = 8192KB , MAXSIZE = 2048GB , FILEGROWTH = 65536KB) WITH CATALOG_COLLATION = DATABASE_DEFAULT, LEDGER = OFF GO ALTER DATABASE [AA_Bounty] SET COMPATIBILITY_LEVEL = 170 GO IF (1 = FULLTEXTSERVICEPROPERTY('IsFullTextInstalled')) begin EXEC [AA_Bounty].[dbo].[sp_fulltext_database] @action = 'enable' end GO ALTER DATABASE [AA_Bounty] SET ANSI_NULL_DEFAULT OFF GO ALTER DATABASE [AA_Bounty] SET ANSI_NULLS OFF GO ALTER DATABASE [AA_Bounty] SET ANSI_PADDING OFF GO ALTER DATABASE [AA_Bounty] SET ANSI_WARNINGS OFF GO ALTER DATABASE [AA_Bounty] SET ARITHABORT OFF GO ALTER DATABASE [AA_Bounty] SET AUTO_CLOSE OFF GO ALTER DATABASE [AA_Bounty] SET AUTO_SHRINK OFF GO ALTER DATABASE [AA_Bounty] SET AUTO_UPDATE_STATISTICS ON GO ALTER DATABASE [AA_Bounty] SET CURSOR_CLOSE_ON_COMMIT OFF GO ALTER DATABASE [AA_Bounty] SET CURSOR_DEFAULT GLOBAL GO ALTER DATABASE [AA_Bounty] SET CONCAT_NULL_YIELDS_NULL OFF GO ALTER DATABASE [AA_Bounty] SET NUMERIC_ROUNDABORT OFF GO ALTER DATABASE [AA_Bounty] SET QUOTED_IDENTIFIER OFF GO ALTER DATABASE [AA_Bounty] SET RECURSIVE_TRIGGERS OFF GO ALTER DATABASE [AA_Bounty] SET DISABLE_BROKER GO ALTER DATABASE [AA_Bounty] SET AUTO_UPDATE_STATISTICS_ASYNC OFF GO ALTER DATABASE [AA_Bounty] SET DATE_CORRELATION_OPTIMIZATION OFF GO ALTER DATABASE [AA_Bounty] SET TRUSTWORTHY OFF GO ALTER DATABASE [AA_Bounty] SET ALLOW_SNAPSHOT_ISOLATION OFF GO ALTER DATABASE [AA_Bounty] SET PARAMETERIZATION SIMPLE GO ALTER DATABASE [AA_Bounty] SET READ_COMMITTED_SNAPSHOT OFF GO ALTER DATABASE [AA_Bounty] SET HONOR_BROKER_PRIORITY OFF GO ALTER DATABASE [AA_Bounty] SET RECOVERY SIMPLE GO ALTER DATABASE [AA_Bounty] SET MULTI_USER GO ALTER DATABASE [AA_Bounty] SET PAGE_VERIFY CHECKSUM

GO ALTER DATABASE [AA_Bounty] SET DB_CHAINING OFF GO ALTER DATABASE [AA_Bounty] SET FILESTREAM(NON_TRANSACTED_ACCESS = OFF) GO ALTER DATABASE [AA_Bounty] SET TARGET_RECOVERY_TIME = 60 SECONDS GO ALTER DATABASE [AA_Bounty] SET DELAYED_DURABILITY = DISABLED GO ALTER DATABASE [AA_Bounty] SET ACCELERATED_DATABASE_RECOVERY = OFF

GO ALTER DATABASE [AA_Bounty] SET OPTIMIZED_LOCKING = OFF GO ALTER DATABASE [AA_Bounty] SET QUERY_STORE = ON GO ALTER DATABASE [AA_Bounty] SET QUERY_STORE (OPERATION_MODE = READ_WRITE, CLEANUP_POLICY = (STALE_QUERY_THRESHOLD_DAYS = 30), DATA_FLUSH_INTERVAL_SECONDS = 900, INTERVAL_LENGTH_MINUTES = 60, MAX_STORAGE_SIZE_MB = 1000, QUERY_CAPTURE_MODE = AUTO, SIZE_BASED_CLEANUP_MODE = AUTO, MAX_PLANS_PER_QUERY = 200, WAIT_STATS_CAPTURE_MODE = ON) GO USE [AA_Bounty] GO / Object: User [DDCAUser1] Script Date: 1/10/2026 3:23:17 PM **/ CREATE USER [DDCAUser1] FOR LOGIN [DDCAUser1] WITH DEFAULT_SCHEMA=[dbo] GO ALTER ROLE [db_owner] ADD MEMBER [DDCAUser1] GO /** Object: Table [dbo].[CandidateAnswers] Script Date: 1/10/2026 3:23:17 PM **/ SET ANSI_NULLS ON GO SET QUOTED_IDENTIFIER ON GO CREATE TABLE [dbo].[CandidateAnswers]([Candidate_ID] [int] NULL, [Job_ID] [int] NULL, [Question_No] varchar NULL, [Question] nvarchar NULL, [Answer] nvarchar NULL, [Answered_On] [datetime] NULL) ON [PRIMARY] TEXTIMAGE_ON [PRIMARY] GO /** Object: Table [dbo].[CandidateTable] Script Date: 1/10/2026 3:23:17 PM **/ SET ANSI_NULLS ON GO SET QUOTED_IDENTIFIER ON GO CREATE TABLE [dbo].[CandidateTable]([Candidate_ID] [int] NULL, [Candidate_Name] nchar NULL, [Candidate_Email] nchar NULL, [Candidate_PhoneNumber]**

nchar NULL, [Category] nchar NULL, [Job_ID] [int] NULL, [MatchScore] nchar NULL, [MatchedSkills] [text] NULL, [MissingSkills] [text] NULL, [ExperienceSummary] [text] NULL, [ActionRecommendation] [text] NULL, [Password] nchar NULL, [ExamLink] [text] NULL) ON [PRIMARY] TEXTIMAGE_ON [PRIMARY] GO / Object: Table [dbo].[JobDescriptionTable] Script Date: 1/10/2026 3:23:17 PM **/ SET ANSI_NULLS ON GO SET QUOTED_IDENTIFIER ON GO CREATE TABLE [dbo].[JobDescriptionTable]([Job_ID] [int] NULL, [Job_Description] [text] NULL, [Date] nchar NULL) ON [PRIMARY] TEXTIMAGE_ON [PRIMARY] GO /** Object: Table [dbo].[QuestionTable] Script Date: 1/10/2026 3:23:17 PM ******/ SET ANSI_NULLS ON GO SET QUOTED_IDENTIFIER ON GO CREATE TABLE [dbo].[QuestionTable]([Job_ID] nchar NULL, [Q1] [text] NULL, [Q2] [text] NULL, [Q3] [text] NULL, [Q4] [text] NULL, [Q5] [text] NULL, [Q6] [text] NULL, [Q7] [text] NULL, [Q8] [text] NULL, [Q9] [text] NULL, [Q10] [text] NULL, [Q11] [text] NULL, [Q12] [text] NULL, [Q13] [text] NULL, [Q14] [text] NULL, [Q15] [text] NULL) ON [PRIMARY] TEXTIMAGE_ON [PRIMARY] GO ALTER TABLE [dbo].[CandidateAnswers] ADD DEFAULT (getdate()) FOR [Answered_On] GO USE [master] GO ALTER DATABASE [AA_Bounty] SET READ_WRITE GO**