ScaleDux

Artificial Intelligence Intern

Task Sheet

Company Name	ScaleDux Software Innovation Pvt Ltd
Position	Artificial Intelligence Intern
Job ID	SDAII101
Task submission due date	Thursday, 24th July, 2025 (Mid night 11 pm)

ScaleDux

Preface:

Hi there,

Welcome, and thank you for showing interest in this opportunity with us at ScaleDux. We've put together this small challenge to help you explore how we think and work on real-world problems. The tasks here are not just for testing skills, they're a way for us to see how you solve things, how you think, and how you bring your ideas to life.

You don't need to be an expert in everything. What matters most is your effort, your curiosity, and how clearly you explain your approach. Whether you're good at coding, design, research, or problem-solving, there's room here for your strengths.

So pick a task you feel most excited about. Take your time. Do your best. And enjoy the process. 🙂

We're not looking for perfect answers, we're looking for honest, well-thought-out attempts.

All the best!

Team ScaleDux

ScaleDux

Rules & Guidelines

Dear candidate - these aren't strict rules. Think of them like helpful tips to guide you:

Pick only one task out of the three

- Choose the one that interests you the most. Don't try to do everything. Just one task, done well, is more than enough.

Use tools you're comfortable with

- We recommend Python because it's easy to review and works well for these tasks, but you're free to use what you know best. If you apply ML/NLP — great. If not, that's also okay, as long as your logic is clear.

Explain your thinking

- Your thought process matters more than fancy code. Help us understand why you did what you did. A clean notebook or a short summary will go a long way.

Keep your work original

- Please don't copy from the internet or AI tools blindly. You can take help, but make it your own. That's what we really care about.

Structure and clarity > complexity

- Even a simple solution, if clearly written, is better than something too complex that's hard to follow. Keep things neat and understandable.

Most importantly — enjoy building it

- This is your space to explore and learn. You're not being judged — we're just trying to know how you approach problems.

Submission Details

Deadline to submit: Thursday, 24th July 2025 — by midnight (11 pm)

How to submit:

- Upload your task to **GitHub** (preferred) or **Google Drive**
- Make sure the link is public or viewable without permission request
- Include:
    - Your code / notebook

- A short write-up (can be in README or a PDF/Word doc)
- Any visuals or charts you've made

If you face any issues or need help, feel free to reach out. (sdash@scaledux.com or dm me on LinkedIn)

All the best! We're really looking forward to seeing how *you* think, solve, and create. Just be yourself — that's more than enough.

ScaleDux

Table of Contents

ScaleDux

TASK 1: Startup Health Scoring Model

Objective:

Your goal is to simulate a **Startup Evaluation Engine** — similar to a credit score but for startups. Using the dataset provided, you will design a method to generate a composite score (out of 100) for each startup based on key business indicators.

Dataset Provided:

You are given a dataset of **100 fictional startups** (CSV file: Startup_Scoring_Dataset.csv), each with the following features:

Column Name	Description
startup_id	Unique ID of the startup
team_experience	Average years of relevant experience among core team (scale of 1 to 10)
market_size_million_usd	Total addressable market (TAM) they claim, in million USD
monthly_active_users	Current number of monthly active users
monthly_burn_rate_inr	Average monthly expenses (INR)
funds_raised_inr	Total funding raised so far (INR)
valuation_inr	Current company valuation (INR)

You can download and load the dataset to start your analysis.

What You Need to Do:

1. Data Preprocessing

- Normalize each numeric column to a **0 to 1 range**
    - Use Min-Max normalization
    - Treat each column appropriately (e.g., higher burn = worse, higher users = better)

2. Define a Custom Scoring Formula

- Assign **weights** to each feature based on your logic (e.g., Team 15%, Traction 25%, etc.)
- Create a final **composite score out of 100**
- The higher the score, the stronger the startup's potential

3. Ranking & Interpretation

- Rank all 100 startups based on your final score
- Identify the **Top 10** and **Bottom 10** performers
- Provide a brief explanation for why one startup scored very high or very low

4. Visualization (Optional but Recommended)

- Create a **bar chart** of scores (sorted)
- Show **correlation heatmap** between input features
- Create a **score distribution histogram**

5. Documentation

- Include a short write-up explaining:
    - Why you chose certain weights
    - How you handled negatively correlated metrics like burn rate
    - Any surprises or insights from your ranking

Notes & Guidelines:

- You're free to use tools and languages you're most comfortable with — but we **highly recommend using Python** along with **basic machine learning libraries** like **scikit-learn**, **XGBoost**, or **TensorFlow/Keras** where relevant. *We'd love to see how you think about applying ML models, even in simple ways — especially in scoring, ranking, or prediction.*
- A good structure would be:
    - Data preprocessing and normalization
    - Feature weighting or selection

  - Optional ML model (if appropriate: regression, classification, clustering)
  - Output visualization and explanation
- **Jupyter Notebook** or **Google Colab** is the preferred environment — it keeps your code, output, and reasoning in one place. *Notebooks with explanations and comments help us understand how you think.*
- Maintain a clean folder layout:
  - **notebooks/** — your working file (.ipynb)
  - **outputs/** — visualizations or charts
  - **README.md** or **summary.txt** — a short write-up explaining your approach
- If you're sharing via GitHub, make sure the repository is public (or access is granted). If using Google Drive, please enable viewing access.
- Add comments to your code — we care more about **clarity and structure** than fancy models or complex formulas.
- **Bonus (not required):**
  - Try model evaluation metrics (e.g., RMSE, accuracy)
  - Use cross-validation or visual tools (e.g., feature importance plots)
  - Suggest improvements or future steps

Reminder:

This task isn't just about the final number — it's about how you approach a real-world business problem using AI/ML thinking.

Do what you can. Document it well. And show us how you solve, not just how you code.

Evaluation Criteria:

Area	Weight
Code quality & logic	30%
Scoring methodology (reasoning)	30%
Visualizations and insights	20%
Documentation and clarity	20%

ScaleDux

TASK 2: Startup Pitch Text Evaluation with NLP

Objective:

You will analyze **multiple startup pitch decks** (5—10) and build an AI/NLP model to extract meaningful signals — such as business clarity, team strength indicators, traction confidence, or market understanding — and use those to generate a **composite quality score** for each deck.

Dataset Provided:

You will be given **5—10 real-world sample pitch decks** in PDF or slide format (to be shared via Google Drive or Notion link). These decks may vary in clarity, structure, and quality.

Each deck may include:

- Problem Statement
- Solution
- Market Size (TAM/SAM/SOM)
- Traction Graphs or Metrics
- Business Model
- Team Overview
- Ask (funding ask + use of funds)

What You Need to Do:

1. Extract and Parse Deck Content

- Use a tool like **PyMuPDF**, **pdfminer**, or **pdfplumber** to extract text
- Clean and preprocess the content (remove junk, separate slides/sections)
- Convert text into structured categories like: Problem, Solution, Traction, etc. (*If decks don't follow a standard format, infer based on keywords/phrases.*)

2. Score Each Deck Using NLP/ML

You may define your own scoring strategy, but here are some ideas:

Dimension	Example Signals
Problem Clarity	Strong if clearly defined and specific
Market Potential	Mentions of large or growing TAM, quantified data

Traction Strength	Metrics like “20% MoM growth” or user numbers
Team Experience	Background in domain, ex-founders, technical skill
Business Model	Subscription, SaaS, transaction-based, clear monetization
Vision / Moat	Mentions of IP, defensibility, data advantage
Overall Confidence	Tone of voice, use of decisive and investor-friendly language

Each dimension can be scored from **0—10** → total score out of 60 or normalized to **100**

Use any NLP tools you prefer: **TextBlob, spaCy, transformers**, or even **zero-shot classification** (e.g., Hugging Face pipelines).

3. Create an Output Dashboard or Summary Table

- One table with: Deck Name, Problem Score, Market Score, ..., Final Score
- Rank decks based on final score
- Identify Top 3 and Bottom 3 decks
- Bonus: Add your own “Investability Insight” — one-liner comment for each deck

4. (Optional Bonus) Generate Summary Reports

- Try summarizing each deck into 4 bullet points using LLMs (OpenAI, Cohere, Hugging Face)
- Or classify decks into categories (e.g., Fintech, HealthTech, SaaS, B2C, etc.)

Output Deliverables:

- Your code notebook (Python + NLP/ML logic)
- Final report (Excel/CSV + PDF/README explaining your scores)
- At least 2—3 visualizations (e.g., scoring radar chart, histogram, correlation heatmap)

Tools & Guidelines

We want you to **work in the environment you're comfortable with**, while also encouraging the use of tools that reflect current industry practices. You’re not restricted to any one stack — but **whichever tools you choose, ensure they are used meaningfully, responsibly, and with clear reasoning.**

Here's what we recommend (not mandatory):

For Coding & Execution

- **Python** is preferred — especially with **Jupyter Notebook** or **Google Colab** — because it allows you to clearly present code, outputs, and explanations together.
- If you're more comfortable with **R**, **JavaScript**, or other environments, you're welcome to use them — just make sure your documentation is easy to follow.

For NLP & ML Tasks

- You may explore libraries like:
    - **TextBlob**, **spaCy**, **VADER**, **NLTK** for NLP
    - **scikit-learn**, **XGBoost**, **Keras**, or **transformers (Hugging Face)** for machine learning
- If you want to use tools like **OpenAI (GPT)**, **Cohere**, or **zero-shot classification APIs**, that's welcome too — just clarify what model/tool you used and how.

For PDF Parsing

- Try tools like **PyMuPDF (fitz)**, **pdfminer.six**, or **pdfplumber** for text extraction.
- If needed, you can also use **OCR tools** like **Tesseract** (for scanned decks), or even manually extract a few slides for demonstration.

For Visualizations

- We love clean, thoughtful visuals — feel free to use:
    - **Matplotlib**, **Seaborn**, **Plotly**, or even **Excel/Tableau** if that's your strength.

Just Make Sure:

- Your tools are genuine, legal, and not auto-generated fluff.
- You include **explanations or comments** that help us understand *why* you used certain methods.
- Your **project structure is clean and well-organized** — think of it as something you'd present to a real hiring panel or VC team.
- If you're trying something new or experimental — **even better**! Just walk us through your logic.

What We're Looking For

Area	What We Evaluate
Extraction Quality	Can you accurately parse and clean the content?
Scoring Logic	Is your evaluation rubric sound and explainable?
NLP/ML Application	Do you apply relevant tools thoughtfully?
Visual Insight	Do your charts help us understand the decks better?
Documentation	Clear README or in-notebook explanations
Innovation	Any bonus logic, summaries, clustering, etc.

Evaluation Criteria:

Evaluation Area	Max Score	What We're Looking For
Extraction & Parsing Quality	20	Was the text extracted cleanly from PDFs? Are slides separated, noise removed, and content usable?
Scoring Logic Design	20	Is the scoring system thoughtful? Are weights justified? Does it reflect real-world investor thinking?
NLP/ML Application	20	Are appropriate tools used? Is there smart use of NLP, sentiment, keyword models, or transformer-based scoring?
Output Dashboard / Summary	15	Is there a final table or report showing scores per deck? Is it easy to interpret, ranked, or summarized well?
Visualizations	10	Are visuals meaningful (bar charts, word clouds, distribution graphs)? Are they labeled and insightful?
Documentation & Structure	10	Is the notebook/commenting clear? Is there a README or in-line explanation of how it works?
Bonus Creativity (Optional)	5	Did the intern go beyond? (e.g., generate summary bullets, rewrite a poor pitch, run clustering or zero-shot classification)

Total Score: 100 Points

We'll be flexible with which tools are used — but we'll score higher for:

- Strong **realistic scoring logic**
- Thoughtful NLP application (even if simple tools)

- Clear reasoning and smart observations

Reminder:

This task mirrors a real-world problem: How to evaluate 100+ pitch decks quickly and meaningfully using AI/NLP.

There's no perfect answer — just thoughtful systems, smart assumptions, and data-backed logic.

ScaleDux

TASK 3: Recommendation Engine for Startup Matching

Objective:

You are tasked with building a **Recommendation Engine** that simulates intelligent matchmaking between users on a platform like ScaleDux — connecting **Founders** with the most relevant **Service Providers and Mentors**, and vice versa.

This engine will power real-time decision-making:

- For founders: *"Who's the best person to help me at my current startup stage?"*
- For service providers: *"Which startup projects match my skills, experience, and timeline?"*

Dataset Provided:

The file User_Matching_Dataset.csv includes **100 users** — 50 Founders and 50 Service Providers/Mentors.

Each user includes:

Founders (Demand Side)

Column	Description
user_id	Unique ID for founder (e.g., F001)
startup_stage	Startup stage: Ideation, Validation, etc.
startup_industry	Sector (e.g., HealthTech, FinTech)
project_need	Specific need (e.g., UI/UX Revamp, Fundraising Support)
tech_requirement	Skill required (e.g., Python, SEO, GST Filing)
project_deadline	Preferred delivery timeframe

Service Providers / Mentors (Supply Side)

Column	Description
user_id	Unique ID (e.g., S001)
user_type	Either "Service Provider" or "Mentor"
expertise_area	Broad area (e.g., Legal Advisor, Design Expert)
industry_preference	Preferred domain (e.g., SaaS, EdTech)
preferred_project_type	What kind of projects they enjoy
core_skill	Their strongest skill
availability	How soon they can begin work

What You Need to Do:

1. Understand Matching Logic

Design a matching algorithm that considers the following:

- **Domain relevance** (e.g., a HealthTech mentor is more relevant for HealthTech founders)
- **Skill match** (e.g., project needs Python → provider has Python)
- **Project type compatibility** (UI/UX Revamp ↔ Design Expert)
- **Timeline fit** (Deadline vs Availability)
- Optional: Match by startup stage and experience area

Define a scoring or ranking system (out of 100) for each potential match pair.

2. Build Two-Sided Match Functions

- **Founder → Top 3 Recommended Providers**
- **Service Provider → Top 3 Matching Projects/Founders**

Return a structured recommendation list (with user IDs, match score, and key reason why they match).

3. Visualize or Present Your Output

Create:

- A match matrix (rows = founders, columns = providers, values = match scores)
- Top N Matches (sorted list per user)
- Bonus: Interactive dashboard (Streamlit, Plotly Dash, or CSV summary)

Output Requirements:

- Code (Python recommended)
- Notebook or script with match logic
- Summary table with Top 3 matches per user (at least 5 founders and 5 providers)
- Explanation of how your logic works
- Bonus: Visualizations (bar chart, heatmap, Sankey, etc.)

Tools & Guidelines:

You're free to choose tools you're most comfortable with. However, we recommend:

- **Python + Pandas** for data handling
- **Scikit-learn, sentence transformers, or cosine similarity** for scoring logic
- **Optional NLP** for fuzzy match between project descriptions and expertise
- **Colab or Jupyter Notebook** for clarity

You may use logic-based ranking (rules) or basic machine learning if confident.

Evaluation Criteria

Area	Max Score	Details
Matching Logic Design	30	Does the system balance all relevant fields smartly?
Code Structure & Readability	20	Clean code, comments, logical flow
Recommendation Quality	20	Do top matches make sense? (domain, skill, timeline)
Output Presentation	15	Is the summary table intuitive and clear?
Bonus Creativity	5	Visualizations, interactivity, dashboard

Reminder:

This is what we solve at ScaleDux - making **startup ecosystems smarter and faster** by reducing decision noise. This task isn't about deep AI - it's about solving a practical matching problem using structured thinking and applied intelligence.