SESSION 1 REFERENCE GUIDE

Session 1: Your First Space

AI + Research Level 2 — Supplementary Material

Concept: INPUT → MODEL → OUTPUT
Space: Silly Phrase Finder
Model: `valhalla/distilbart-mnli-12-3` (zero-shot classification)
Pre-built fallback: profplate/silly-phrase-finder

Time Breakdown (2 hours)

0:00–0:15 — Introductions & Space Tour

Welcome students. Quick intros: name, grade, one thing you've used AI for.

Then tour 3–4 deployed Spaces. Share your screen and visit each one live:

Space	URL	What to try
Background Removal	https://huggingface.co/spaces/KenjieDec/RemBG	Upload a photo, watch the background vanish
ACE-Step Music Gen	https://huggingface.co/spaces/ACE-Step/Ace-Step-v1.5	Generate a short clip
Text to Emotion	https://huggingface.co/spaces/Elegbede/Text_to_emotion_classifier	Type a sentence, see the emotion
DeepSeek OCR	https://huggingface.co/spaces/merterbak/DeepSeek-OCR-Demo	Upload a photo of text

Space Observation Card — for each Space, ask students:
1. What goes in? (text, image, audio, etc.)
2. What comes out?
3. What happens if you give it weird input?

Landing line: "These are all Hugging Face Spaces. By the end of tonight, you'll have one too."

0:15–0:30 — Account Setup

Walk students through:
1. Go to huggingface.co and create a free account
2. Brief hub tour:
- **Spaces** page — browse what people have built
- **Models** page — the library of AI models anyone can use
- Their **profile** page — this is where their Spaces will live

Talking points:
- "Your HF profile is your AI portfolio. Everything you build here is public and yours."
- "Think of the Models page like an app store, except everything is free and open-source."

0:30–0:45 — Show the Finished Space

Open the pre-built fallback: profplate/silly-phrase-finder

Demo flow:
1. Paste one of the examples. Show the result.
2. Ask students to suggest inputs verbally. Try 3–4.
3. Try something that surprises the model — a sentence that seems silly to humans but the model rates as serious, or vice versa.

Key question: "How does it know what's silly? Nobody taught it 'silly.' We never showed it examples of silly sentences."

Let this question hang. Don't answer it yet.

0:45–1:20 — Build It Live

Create a new Space on HF from scratch:

1. Click **New Space** on HF
2. Name it `silly-phrase-finder`
3. Select **Gradio** as SDK
4. Choose **Public** and **Free CPU**
5. Click **Create Space**

Now open the Files tab and create `app.py`. Type the code line by line. Explain each piece as you go:

```
import gradio as gr
from transformers import pipeline
import re
```

Say: "We're importing three tools. `gradio` builds the web interface. `transformers` loads the AI model. `re` helps us split text into sentences."

```
classifier = pipeline("zero-shot-classification", model="valhalla/distilbart-mnli-12-3")
```

Say: "This one line loads the AI model. `pipeline` is like a shortcut — it handles all the complicated stuff. `zero-shot-classification` is the task. The model name tells it which brain to download."

```
def find_silliest(text):
```

Say: "Now we write a function. Text goes in, the silliest phrase comes out."

Walk through the rest of the function:
- Input validation ("what if someone clicks with no text?")
- Sentence splitting ("we need to compare sentences, so first we chop the text up")
- Labels ("we're asking: is this silly, serious, or ordinary?")
- Scoring loop ("check every sentence, keep the silliest one")
- Return string ("format it nicely for the user")

Then the Gradio interface:

```
demo = gr.Interface(
    fn=find_silliest,
    inputs=gr.Textbox(lines=10, placeholder="Paste a paragraph or two here..."),
    outputs=gr.Textbox(label="The Silliest Phrase"),
    title="Silly Phrase Finder",
    ...
)
demo.launch()
```

Say: "`gr.Interface` connects your function to a web page. `fn` is the function, `inputs` is what the user types, `outputs` is what they see."

Also create `requirements.txt`:

```
transformers
torch
gradio
```

Say: "This tells HF what packages to install. Without it, the Space won't know it needs transformers."

Commit and watch the Space build.

1:20–1:40 — Test It

Once the Space is live:
1. Try the examples from the code
2. Ask students to suggest inputs — try at least 5–6
3. Look for surprises: when does the model get it "wrong"?

Pre-prepared inputs to try:
- A paragraph from a news article (serious)
- A paragraph from a children's book (should find something silly)
- A mix: "The meeting is at 3pm. A duck wearing sunglasses walked into the boardroom. Please bring your laptops."
- Student-suggested inputs (the best part)

1:40–1:50 — Name the Concept: INPUT → MODEL → OUTPUT

Draw this on screen (or describe it):

```
TEXT (input) → ZERO-SHOT MODEL (processing) → SILLIEST PHRASE (output)
```

Talking points:
- "Every AI tool you've ever used follows this pattern. ChatGPT: your prompt goes in, text comes out. Image generators: a description goes in, a picture comes out."
- "The model is the engine. We didn't build the engine — we just told it what to look for."
- "Zero-shot means the model was never trained on 'silly.' It figures it out from what it already knows about language."

Quick check: "Can someone give me the input, model, and output for that background removal Space we saw earlier?"

Now show the model card for `valhalla/distilbart-mnli-12-3`:
- https://huggingface.co/valhalla/distilbart-mnli-12-3
- Point out: what it was trained on (MultiNLI), what task it does, who made it

1:50–2:00 — Notebook Time

Share the Colab link in the Zoom chat. This is students' first time opening a notebook.

Walk through together:
1. "Click the link — it opens in your browser"
2. "See the cells with code? Hit the play button on the left of the first cell"
3. "Wait for the green check — that means it worked"
4. Run the setup cell and the first experiment cell together on the call

Say: "This is a notebook. It's like a document where you can run code. You'll finish the experiments on your own — they're fun, I promise."

Notebook skill being introduced: Opening a Colab notebook from a link, running a cell

GitHub skill being introduced: "Before next session, create a GitHub account at github.com if you don't have one. We'll use it starting next week."

2:00 — Wrap Up

Share the between-session challenge (see BETWEEN-SESSION.md). Emphasize it's optional but fun.

Say: "Next week we're going to swap the engine. Same Space, different model. Things will get weird."

What Could Go Wrong

Problem	Fix
HF Space takes 3–5 min to build	Fill time by showing the code in the Files tab. Explain what's happening: "It's downloading the model and installing packages."
Student can't create HF account	Use profplate/silly-phrase-finder as shared demo. They can create accounts between sessions.
Space crashes on deploy	Check `requirements.txt` exists and has all three packages. Check for typos in `app.py`.
Model gives weird results	That's a feature! "The model doesn't think like us. We'll explore why next session."
Student falls behind during live build	They'll catch up via the pre-built Space. The live build is instructor-led on shared screen.

Key Vocabulary (introduce casually, don't drill)

- **Space** — a web app on Hugging Face
- **Model** — an AI brain trained on data
- **Pipeline** — a shortcut that loads a model and makes it easy to use
- **Zero-shot** — the model can do tasks it was never specifically trained for
- **Gradio** — the Python library that builds the web interface