PROTOTYPING PRODUCTS WITH DATA & AI

Assignment 1 - AI Prototype

DANCE FINDER

Discover dance classes in Barcelona - powered by AI

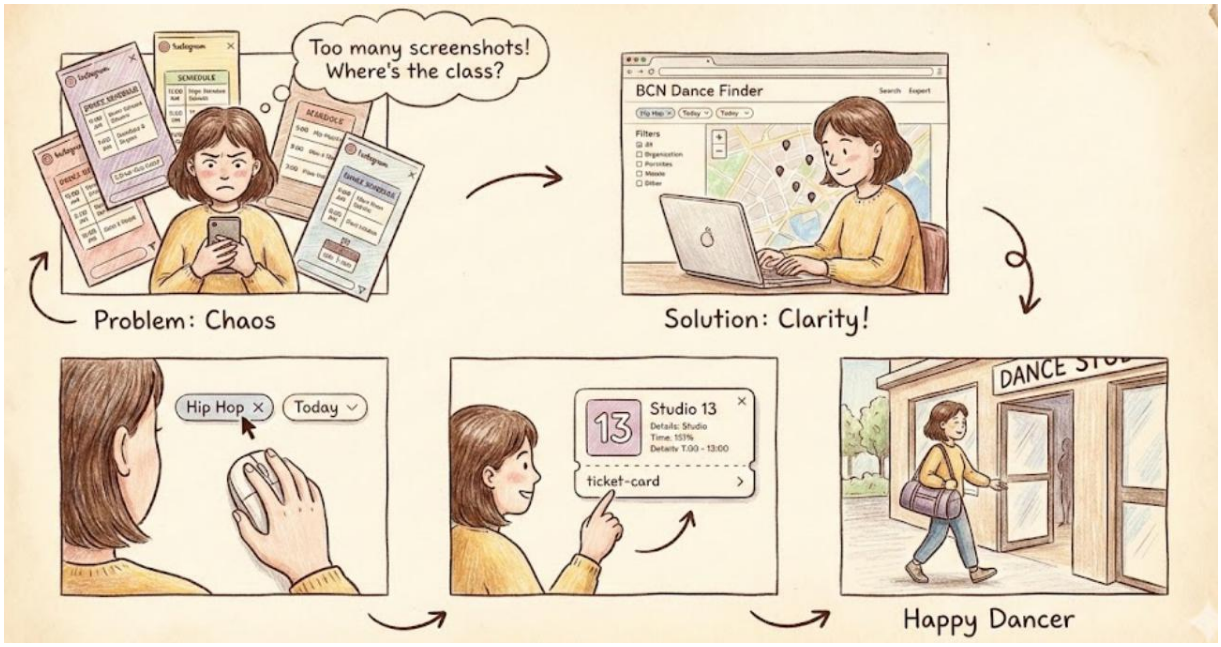
Mara Rüsen

ESADE Business School | Barcelona | 2026

🌐 **Live App**: https://dance-finder-ax2tbcexhyhjqsjij2exkw.streamlit.app

💻 **GitHub Repository**: https://github.com/MaraRusen/dance-finder

⚙️ **Pipeline Page (How It Works)**: https://dance-finder-ax2tbcexhyhjqsjij2exkw.streamlit.app/about



1. Executive Summary

Dance Finder is an AI-powered web application that helps people discover dance classes in Barcelona. Built with Streamlit and powered by a Vision LLM data pipeline, it aggregates class information from Instagram stories and studio websites into one beautiful, real-time interface.

The Core Problem

Finding a dance class in Barcelona (or any other city) today means scrolling through dozens of Instagram stories, checking multiple studio websites, and still not knowing if the class is full or who the teacher actually is. There is no central place to discover what is happening tonight.

The Solution

Dance Finder aggregates all dance classes in Barcelona into one interface, showing who is teaching, where, when, at what price, and how full the class is expected to be. All processed by AI, presented in real time.

2. Value Proposition

Dance Finder addresses a clear gap in the Barcelona dance scene: the complete lack of a centralized, real-time discovery platform for drop-in dance classes.

2.1 Target Users

- **Primary:** Recreational dancers looking for drop-in classes tonight
- **Secondary:** Dance tourists visiting Barcelona who do not know the local scene
- **Tertiary:** Intermediate/advanced dancers who follow specific teachers

2.2 Key Value Drivers

Value Driver	What It Solves	How Dance Finder Delivers It
Time savings	Scrolling 10+ Instagram accounts daily	All classes aggregated in one view
Discovery	Unknown teachers and new studios	Rich teacher profiles with bios and styles
Crowd prediction	Showing up to a full class	AI crowd score per class
Trust & transparency	Not knowing if AI data is accurate	Original IG story source always visible
Free trials	Not knowing which studios offer free entry	Dedicated free trial filter and badge

3. Product Overview

The prototype is a fully deployed Streamlit web application with two pages: the main Dance Finder dashboard and a pipeline explanation page (How It Works).

3.1 Main Page: Dance Finder

The main page is designed around the mental model of a dancer looking for a class tonight. Every design decision prioritizes speed and clarity.

Hero Section

- Full-width background video with dark purple/blue gradient overlay
- Large typographic title: DANCE FINDER
- Live date and city location pulled dynamically
- Call-to-action button linking to the pipeline explanation page

Filter Bar

- All filters on a single horizontal bar — no sidebar needed
- Date picker (st.date_input) — browse classes by day
- Dance style multiselect (st.multiselect) — filter by Hip Hop, Salsa, Breaking, etc.
- Teacher multiselect (st.multiselect) — follow your favorite teacher
- Level multiselect (st.multiselect) — Beginner, Intermediate, Advanced, All Levels
- Free only toggle (st.toggle) — show only free trial classes

Metrics Row

- 4 dynamic metric cards: Classes Today, Free Trials, Selling Fast, Styles Available
- All metrics update in real time based on the active filters

Interactive Map

- Folium dark-tile map centered on Barcelona
- Color-coded circle markers: blue (available), purple (filling up), orange (almost full)
- Clickable popups showing style, studio, time, teacher, and price

Ticket Cards

- Custom HTML/CSS cards styled like event tickets — not boring tables
- Style badge, level badge, and crowd percentage badge per card
- Teacher name, studio, neighborhood, time, and origin clearly displayed
- Price badge: green for free trials, purple for paid classes
- Expandable section: full teacher bio, today's topic, and booking link
- Expandable section: original Instagram story screenshot (AI transparency)

3.2 About Page - How It Works

A dedicated second page explains the full AI pipeline to the user. This page covers the 3 pillars of the prototype, the step-by-step data pipeline, both AI models, and the full tech stack.

4. The 3 Pillars of the AI Prototype

This prototype was built around the three core aspects defined in the PDAI course framework. Each pillar is addressed with specific, intentional design and technical decisions.

Pillar 1 - Appearance & User Experience

The visual design was inspired by real Barcelona dance studios (Studio 13 Danza, Urban Roots BCN) - dark, editorial, and bold. The goal was to create an interface that feels like it belongs in the dance world, not like a generic data app.

- Dark purple/blue color palette (#080818 background, #4F8EF7 blue accents, #7C5CBF purple accents)
- Bebas Neue display font for all headings — the standard in dance and streetwear branding
- Custom HTML/CSS ticket cards rendered inside Streamlit using st.markdown(unsafe_allow_html=True)
- Frosted glass effect on all cards using backdrop-filter: blur(12px)
- Radial gradient background glow for depth and atmosphere
- Video hero section with hue-rotate filter to match the purple/blue palette

Pillar 2 - Data & AI Pipeline

The pipeline is intentionally separated into two phases to keep the Streamlit app fast at runtime. No API calls happen when a user loads the page.

```
OFFLINE PHASE — runs daily via cronjob
──────────────────────────────────────────────────
Step 1: scraper.py downloads IG stories + scrapes studio websites
Step 2: extractor.py (GPT-4o Vision API) reads images → returns JSON
Step 3: Output saved to classes.json + appended to classes_history.csv
Step 4: model.py trains Random Forest on history → saves model.pkl

RUNTIME PHASE — the Streamlit app
──────────────────────────────────────────────────
app.py loads classes.json → renders map + ticket cards + filters
model.pkl loaded → crowd_score displayed as badge on each card
```

Prototype Note

In this prototype, classes.json is a hand-crafted mock dataset that simulates the exact output format of the Vision LLM pipeline. The crowd_score values simulate the Random Forest model output. The architecture is fully production-ready - only the data source is mocked.

Pillar 3 - Accuracy & Transparency

Transparency is a core design principle of Dance Finder. Users should never have to trust the AI blindly.

- Every ticket card has a 'View Original Instagram Story' expander that shows the raw screenshot the AI read
- The crowd score is displayed as a percentage (e.g. 87% Full) — not a binary 'full/available' label
- The About page fully explains how the data was collected, processed, and displayed
- AI Confidence and last update timestamp shown on every IG story viewer

5. Technical Architecture

5.1 Project Structure

```
dance-finder/
├── app.py                  # Main Streamlit app
├── classes.json            # Mock dataset (simulates AI pipeline output)
├── requirements.txt        # Python dependencies
├── .gitignore              # Excludes video + cache files
├── README.md               # Full project documentation
│
├── pages/
│   └── about.py            # How It Works — pipeline explanation page
│
└── static/
    └── dance.mov           # Hero background video (excluded from git)
```

5.2 Streamlit Widgets Used

The assignment required using Streamlit widgets not covered in previous classes. The following widgets were implemented:

Widget	Usage in App	In Class?
st.multiselect	Dance style, teacher, and level filters	No — new
st.toggle	Free trial only filter	No — new
st.date_input	Date picker for browsing classes	No — new
st.metric	4 KPI boxes at the top of the page	No — new
streamlit-folium	Interactive Folium map integration	No — new
st.expander	Teacher bio and IG story viewer	No — new
st.markdown (HTML)	Custom ticket cards and badges	No — new

5.3 The Mock Dataset (classes.json)

The mock dataset contains 6 realistic Barcelona dance classes, each with full metadata to demonstrate the AI pipeline output format:


```
{
  "style": "Hip Hop",
  "studio": "Studio 13",
  "teacher": {
    "name": "Marco Rivera",
    "origin": "Puerto Rico",
    "bio": "Marco trained at Millennium Dance Complex in LA...",
    "todays_topic": "Musicality & groove fundamentals",
    "instagram": "@marco.moves"
  },
  "time_start": "19:30",
  "time_end": "20:30",
  "price_label": "25€ Drop-in",
  "free_trial": false,
  "crowd_score": 0.87,
  "lat": 41.3851,
  "lon": 2.1734
}
```

5.4 The Two AI Models (Planned for the future)

Model 1 - Vision LLM Data Extractor

GPT-4o Vision reads each Instagram story screenshot and returns structured JSON. The prompt used is:

```
system_prompt = """
You are a dance class data extractor. Given an Instagram story image,
extract the following fields and return ONLY valid JSON:
- time_start, time_end (HH:MM format)
- teacher (name, origin, bio, instagram)
- style (e.g. Hip Hop, Salsa, Breaking)
- level (Beginner / Intermediate / Advanced / All Levels)
- price_label (e.g. "25€ Drop-in" or "Free Trial")
- studio (studio name)
- address (full Barcelona address)
"""
```

Model 2 - Crowd Predictor (Random Forest)

After weeks of data collection, classes_history.csv accumulates daily rows. A Random Forest model is trained to predict crowd levels:

Feature	Description
weekday	Monday=0 to Sunday=6
teacher_encoded	Label encoded teacher ID
style_encoded	Label encoded dance style
hour	Class start hour (0–23)
weather_score	Weather API score (0–1)
target: spots_filled_pct	Historical fill rate (0.0–1.0)

Model 3 - Personalized Class Matching (Planned)

As users interact with Dance Finder over time, saving classes, booking, and rating teachers, the app accumulates personal preference data. A collaborative filtering model (similar to those used by Spotify or Netflix) uses this history to recommend classes the user has not tried yet.

The model learns from patterns across all users: if dancers who love Hip Hop and Breaking also tend to enjoy Afrobeats, the model recommends Afrobeats to a new user who has only taken Hip Hop so far.

What the model needs to work: User interaction logs (saved classes, bookings, ratings), Minimum ~50 users with at least 3 interactions each, Weekly retraining as new interaction data accumulates

Output: A personalized "Recommended for You" section on the main page, ranked by predicted match score.

6. How It Was Built - Step by Step

The prototype was built iteratively over one session, following a design-first, then data, then pipeline approach.

Step 1 - Define the Concept

Rather than using a standard dataset (e.g. Paris real estate), the project started from a real personal problem: as a dancer in Barcelona, finding drop-in classes requires checking multiple Instagram accounts and studio websites every day. Dance Finder was designed to solve this exact problem.

Step 2 - Design the Data Model

The classes.json schema was designed to simulate the exact output of the Vision LLM pipeline. Every field maps directly to a UI element in the app — nothing is stored that is not displayed.

Step 3 - Build the Streamlit App (app.py)

The app was built in layers:

1. Page config and global CSS injection (dark theme, fonts, variables)
2. Hero section with base64-encoded background video
3. Filter bar with 5 widgets on a single row
4. Filter logic applied to classes.json in real time
5. 4 metric boxes (st.metric) showing live KPIs
6. Two-column layout: Folium map (left) + ticket cards (right)
7. Custom HTML/CSS ticket cards with badges, hover effects, and expanders

Step 4 - Build the Pipeline Page (about.py)

A second Streamlit page was created to explain the full AI pipeline. This page mirrors the design system of the main app and covers all three pillars, the pipeline diagram, both AI models, and the tech stack.

Step 5 - Deploy

The app was deployed in two stages:

- GitHub repository created at github.com/MaraRusen/dance-finder (public)
- Streamlit Cloud connected to the GitHub repo for automatic deployment
- Every git push triggers an automatic redeploy on Streamlit Cloud

7. Tech Stack

Category	Tool	Purpose
Frontend	Streamlit	Web app framework
Frontend	Folium + streamlit-folium	Interactive map
Frontend	Custom HTML/CSS	Ticket cards, badges, hero section
Frontend	Bebas Neue + Inter (Google Fonts)	Typography
AI Extraction	GPT-4o Vision API	IG story image → JSON (simulated)
Web Scraping	BeautifulSoup	Studio website scraping (planned)
Web Scraping	instaloader	Instagram story download (planned)
ML Model	scikit-learn / XGBoost	Crowd prediction model (planned)
Storage	JSON	Today's classes (runtime)
Storage	CSV	Historical data (model training)
Deployment	Streamlit Cloud	Live hosting
Version Control	GitHub	Code repository

8. Links & Access

Resource	URL
Live App (Streamlit Cloud)	https://dance-finder-ax2tbcexhyhjqsjij2exkw.streamlit.app
GitHub Repository	https://github.com/MaraRusen/dance-finder
Pipeline Explanation Page	https://dance-finder-ax2tbcexhyhjqsjij2exkw.streamlit.app/about

9. Conclusion

Dance Finder demonstrates how AI can solve a real, everyday problem in an elegant and transparent way. The prototype successfully addresses all three pillars of the PDAI assignment framework:

- Appearance & UX: A polished, editorial dark interface with custom ticket cards, interactive map, and hero video — designed to feel native to the dance world
- Data & AI Pipeline: A fully architected two-phase pipeline (offline AI extraction + runtime display) with two AI models (Vision LLM + Crowd Predictor)

- Accuracy & Transparency: Every data point is traceable to its original Instagram story source, and confidence scores are shown numerically rather than as binary labels

The result is a prototype that is not only technically sound but also genuinely useful — something a dancer in Barcelona could use every day.

Key Differentiators vs. Standard Assignments

1. Original use case (not a provided dataset) | 2. Custom HTML/CSS widget system (not default Streamlit) | 3. Two-phase offline/runtime pipeline architecture | 4. AI transparency built into the UX | 5. Fully deployed with a live public URL

Appendix: Screenshots so I don't forget my token.