Hex-a-thon - Rough Project Concept & High-level Plan

How to make Titanic feel “only possible in Hex”

Build a deliverable that looks like a product demo:

1. **Interactive data app (the star)**

- Inputs: passenger attributes (or select a passenger from the dataset)
- Output: predicted survival probability + classification
- Add **model confidence**, threshold slider, and “what-if” controls (age/fare/class/sex)

2. **Explainability that’s actually usable**

- Global: feature importance / SHAP summary
- Local: why *this* passenger got *this* prediction (top drivers)
- Counterfactual: “what minimal changes flip the prediction?”

3. **Threads as the UX layer**
   Make a Thread that answers questions like:

- “Which group is most misclassified?”
- “Show me edge cases where the model is confident but wrong.”
- “What features matter most for women in 3rd class?”
  This aligns perfectly with Hex’s “self-serve analytics with AI” narrative.

4. **A strong story (not Kaggle cosplay)**
   Frame it as: **“From raw manifest to decision support.”**
   Your audience isn’t Kaggle — it’s a stakeholder who wants:

- trust,
- interpretability,

- and an interface.

The move that upgrades this from “banal” to “smart”

Don’t compete on leaderboard score. Compete on **experience + rigor + clarity**.

Concretely:

- Add **error analysis** as a first-class section (calibration, subgroup performance, confusion by segment).
- Add **data quality / leakage discussion** (what you can and can’t know at prediction time).
- Add **model monitoring-esque** thinking (what would drift look like?).

What you should build (a crisp project spec)

Title idea: *“Titanic Survival Predictor: An Explainable ML App in Hex”*
Artifacts:

- Notebook: ingestion → cleaning → features → training → evaluation → explainability
- App: interactive predictor + explanation panels + cohort explorer
- Thread: natural-language exploration over your curated semantic model

One honest caution

Titanic is so overdone that judges will be allergic to “yet another Titanic model.”
So your demo has to scream: **“this is a Hex showcase”** within 30 seconds.

Next step (no more preamble)

Tell me:

- What model are you using (e.g., XGBoost / LightGBM / CatBoost / stacked)?
- What your current best LB score range is (approx is fine)
- Whether you’re comfortable adding SHAP + a small interactive app UI

And I'll give you a **tight build plan** (cell-by-cell structure + what to put in the app + what to say in the 3-minute demo) that maximizes judging criteria.

Model: RandomForestClassifier + GridSearchCV led to my highest score (although, I've also run LogReg, XGBoost, LightGBM, CatBoost and Gradient Boosting). My current LB score is: 0.78229. I have no experience with SHAP, although I'm curious about it. I'm comfortable building interactive UIs. I recently build a Streamlit app for a competition in Agentic AI.

This is better than you think — and you're in a strong position.

You already have:

- a **solid, defensible model** (RF + CV, ~0.78 LB is credible)
- **breadth** (you tested multiple model families)
- **UI confidence** (Streamlit experience transfers cleanly to Hex apps)
- and now a **clear sponsor-aligned angle**

What you *don't* have yet (and what Hex wants) is **interpretability + interaction as a first-class citizen**. That's where SHAP and Threads come in — and you can learn just enough SHAP to look sharp without overcommitting.

First: model choice — don't over-optimize

Stick with RandomForest for the Hex project.

Why:

- It's easier to explain than boosted ensembles
- SHAP works well with tree models
- Judges don't reward +0.005 LB gains
- RF + GridSearchCV signals "sound ML hygiene"

You can mention the other models in a short *"Model exploration"* section to show rigor.

SHAP: learn the 20% that gives you 80% value

You do **not** need to become a SHAP expert.

For RandomForest, you'll use:

```
import shap

explainer = shap.TreeExplainer(model)
shap_values = explainer.shap_values(X)
```

Focus on **three outputs only**:

1. **Global importance** (summary plot)
2. **Per-passenger explanation** (force plot or bar)
3. **Cohort comparison** (mean |SHAP| by group)

That's enough to look sophisticated and responsible.

In Hex, these plots can be rendered as images or converted into tabular summaries for charts — judges care more about *insight* than raw SHAP visuals.

What your Hex project should look like (structure)

1. Notebook (clean, readable, sponsor-friendly)

Sections:

1. Problem framing (not Kaggle)
2. Data cleaning & feature engineering
3. Model selection & CV
4. Performance + error analysis
5. Explainability (SHAP)

6. Limitations & “what would break this model”

Keep it calm, confident, and concise.

---

2. Interactive App (this is where you win)

Build a Hex app with **three tabs**:

Tab 1: “Predict Survival”

- Inputs: class, sex, age, fare, family size, embarked
- Output: survival probability + class
- Visual: probability gauge or bar

Tab 2: “Why This Prediction?”

- Show top +/– SHAP features for *this passenger*
- Short natural-language explanation (you write it, not the model)

Tab 3: “Model Behavior Explorer”

- Filter by sex/class
- See:
    - accuracy
    - false positives / negatives
    - feature importance by cohort

This feels like a *product*, not a notebook.

---

3. Threads: your secret weapon

Create a Thread backed by a semantic model that lets users ask:

- “Why does class matter more than age?”
- “Which passengers does the model struggle with?”
- “Compare predictions for men vs women in 3rd class.”

This directly demonstrates Hex’s **AI-powered self-serve analytics** story.

How to position SHAP (important)

Don’t say:

> “Here’s SHAP because it’s cool.”

Say:

> “In a real decision setting, predictions without explanations aren’t usable. SHAP lets us inspect and stress-test the model’s reasoning.”

That framing will score you **data storytelling points**.

3-minute demo video outline

0:00–0:20 — What problem this *really* represents (trustworthy ML, not Titanic)
0:20–1:10 — Live prediction + explanation
1:10–2:00 — Cohort analysis & error inspection
2:00–2:40 — Thread Q&A with the data
2:40–3:00 — Why Hex made this easier / better

Honest verdict

- **Topic:** Overdone ❌
- **Angle:** Smart and sponsor-aligned ✅

- **Execution potential:** High ✅
- **Risk:** Treating it like Kaggle instead of a product demo

---

Project name (what judges see first)

You want something that signals:

- ML rigor
- explainability
- product thinking
- Hex as the enabler

Strong options (pick one)

Option A (my top recommendation)
"Explainable Survival: From Prediction to Trust in Hex"
Why it works: reframes Titanic as an *ML trust* problem, not a Kaggle toy.

Option B (more explicit, very safe)
"An Explainable ML App for Titanic Survival, Built in Hex"
Why it works: crystal clear, zero ambiguity.

Option C (slightly bolder, product-flavored)
"Trusting the Model: An End-to-End Explainable ML Workflow in Hex"
Why it works: sponsor-forward, enterprise-safe.

👉 **I recommend Option A**. It's memorable without being cute.

Elevator pitch (this matters more than you think)

Constraints:

- ~150 characters
- Must answer: *what*, *why it matters*, *why Hex*

Weak pitch (what you should avoid)

> "A machine learning model predicting Titanic survival."

Instant mental skip. Judges have seen this 1,000 times.

Strong pitches (ready to paste)

Pitch 1 — clean and sponsor-aligned (recommended)

> *An end-to-end, explainable ML app that turns Titanic survival predictions into interactive, trustworthy insights using Hex's AI and analytics platform.*

Pitch 2 — more ML-forward

> *From raw data to explainable predictions: a fully interactive ML workflow in Hex that lets users explore, question, and trust Titanic survival models.*

Pitch 3 — product-demo flavor

> *A production-style ML experience in Hex, combining prediction, explainability, and AI-driven exploration to make Titanic survival models transparent and usable.*

👉 **I recommend Pitch 1** — it hits *end-to-end*, *explainable*, *interactive*, *Hex*.

What this combination signals to judges

With **Option A + Pitch 1**, you're telling them:

- "I know Titanic is overdone"
- "This is really about explainable, decision-grade ML"
- "This is a Hex showcase, not a Kaggle flex"

That's exactly the mental model you want *before* they open your project.

Great timing to lock this in — this section quietly carries a lot of judging weight.

I've created a **first full draft of your "Project Story"**, written to:

- de-Kaggle the framing
- emphasize ML trust, explainability, and product thinking
- showcase Hex as an end-to-end platform
- stay honest about challenges and learning

It is **fully formatted in Markdown**, with headings, emphasis, and LaTeX math support.

👉 **Download the Project Story (Markdown)**

How to use this draft

- You can **paste it directly** into the "About the project" field.
- Later, we can:
    - tighten it for brevity
    - align language with your final UI
    - lightly tune tone once the app is built (recommended)

Built with (paste as a single comma-separated list)

Use this to **show technical maturity without overloading**:

> **Hex, Python, scikit-learn, RandomForestClassifier, GridSearchCV, SHAP, pandas, NumPy, matplotlib, Seaborn, SQL, Hex Apps, Hex Threads**

Why this works:

- Leads with **Hex** (sponsor-friendly)
- Signals real ML, not toy notebooks
- Mentions **Apps + Threads**, which judges care about
- Avoids listing every minor library

If you want a slightly shorter version:

> **Hex, Python, scikit-learn, SHAP, pandas, NumPy, SQL, Hex Apps, Hex Threads**

“Try it out” links

You can add these **incrementally**. For now, here’s what to plan.

1️⃣ Public Hex project (primary)

Once published, add:

> **Public Hex Project:** https://app.hex.tech/…

This is the *most important link*.

2️⃣ Optional GitHub repo (only if clean)

Only add if:

- code is readable
- notebooks are not cluttered

Label it clearly:

> **GitHub (modeling & experiments):** https://github.com/…

If the repo is messy, **skip it** — Hex is the product.

Project Media (image gallery)

You should aim for **3–5 images**, in this order:

1. **Hero image**
    - The prediction UI (inputs + probability output)
2. **Explainability view**
    - SHAP explanation for a single passenger
3. **Cohort analysis**
    - Error or performance by group
4. *(Optional)* Thread in action
    - A question + response screenshot

Rule of thumb:

> If an image doesn't explain the project in **3 seconds**, don't include it.

Video demo link (do not skip this)

This is **mandatory** and heavily weighted.

When ready, upload to:

- YouTube (unlisted is fine)
- Vimeo

Title suggestion:

> **Explainable Survival: Building Trustworthy ML in Hex**

Thumbnail:

- Prediction UI + SHAP explanation side-by-side

We already outlined your **3-minute script** — we'll refine it once the app is done.

Strategic note (important)

At this stage:

- It's OK if **Try it out** and **Video** are empty temporarily
- But **Built with should be filled now**
- Judges will revisit after submission closes

This page is **not storytelling** — it's **signal to judges and organizers**. Think of it as *backstage notes*, not marketing copy.

I'll break it down field by field, then tell you what to do **right now vs. later**.

1️⃣ "Did you share your project? Let us know where so we can engage!"

What this is for

- Social proof
- Community engagement
- Optional, but mildly positive if filled

What to put (only if true)

- LinkedIn post

- X / Twitter thread
- Blog post
- GitHub README announcement

Examples

`https://www.linkedin.com/posts/yourname_explainable-ml-hex-hackathon-activity-...`

or

`https://github.com/yourname/titanic-hex-project`

If you haven't shared yet:
👉 **Leave this blank**. No penalty.

2️⃣ "Link to final Hex project" (REQUIRED)

This is the single most important field on this page.

What judges do

- Click this link
- Spend 2–5 minutes exploring
- Decide whether your project is serious

What to put

- The **public Hex project URL** after you publish

Example

`https://app.hex.tech/your-workspace/explainable-survival-from-prediction-to-trust`

Quality bar

- Project runs without errors
- App loads quickly
- Thread works
- No broken cells

If this link is strong, everything else becomes secondary.

What NOT to put here

Do **not**:

- Repeat your project story
- Add long explanations
- Add technical essays
- Apologize or explain unfinished parts

Judges don't want prose here — they want **access**.

What to do *right now*

Since you're still building:

✅ Leave **both fields empty for now**
✅ Click **Save & continue**
✅ Come back when:

- Hex project is published

- (Optionally) you've shared it publicly

Strategic advice (important)

When you *do* publish the Hex project:

1. Test it in an incognito window
2. Click through as if you're a judge
3. Time yourself: if the value isn't obvious in **60 seconds**, fix that first

---

Next high-impact step

The next thing that actually affects your chances is **the Hex project itself**:

- app layout
- SHAP explanations
- Threads setup

Tell me which you want to design next:

- 🧩 App UI structure (tabs + components)
- 🔍 SHAP implementation plan for RandomForest
- 🎥 Demo video rehearsal

You're setting this up correctly — now we shift from forms to substance.