# Interview Tasks

## Introduction

Thank you for taking the time to complete our technical assessment. This project is designed to give us some insight into your skills across the entire stack, from backend services and machine learning to frontend implementation and architectural thinking.

## A Note on Your Time

We value your time and have designed this task to be completed in stages. We expect the Core Tasks (1 and 2) to take approximately 4-6 hours. Please focus on quality, clarity, and testing over completing every single feature. The Advanced Task (Task 3) is optional and provide an opportunity for you to showcase additional skills if you have the time.

**Task 1**: Housing Price Prediction Model API

**Objective:**
Build, containerise, and deploy a simple regression model that predicts housing prices based on provided features (dataset attached).

**Requirements:**

- Api Endpoints:
    - /predict - Accepts housing features and returns price predictions. Both single and batch numbers
    - /model-info - Returns model coefficients and performance metrics
    - /health - Simple health check endpoint

**Technical Constraints:**

- Python 3.12+
- FastAPI
- Scikit-learn

**Deliverables:**

1- Source code in github
2- Dockerfile
3- Ability to show this live during the interview via (Swagger/OpenAPI)

**Task 2**: Property Portal Application

**Objective:**

Build a Next.js frontend and a Python backend that together form a complete web application. The backend will communicate with the Pricing API (Task 1).

**Backend Requirements (Python/FastAPI)**

- Proxy/Orchestrate calls to Pricing API
- Additional endpoints:
    - api/estimate: Frontend submits property data, backend calls Pricing API
    - api/market-data: Serve aggregated market stats (can be mocked)
    - api/history: Return previous estimates (mock or in-memory storage)
    - Error handling If Pricing API is down, return errors
    - Request/response validation using Pydantic

**Frontend Requirements (Next.js)**

- Web portal with navigation between:
    - Valuation Tool: Form to submit property details, see prediction
    - Market Dashboard: Charts, filters, data table
    - Responsive UI: with Tailwind CSS
- Client-side form validation
- Display results in table and chart
- Loading/error states for API calls

**Integration Requirements**

- Frontend calls: Your Python backend, not directly the Pricing API
- Backend must communicate with the containerised Pricing API
- Handle network errors between services

**Technical Stack**

- Backend:
    - Python 3.12+, FastAPI, Pydantic
- Frontend:
    - Next.js 15+, App Router, Tailwind

**Deliverables**

1- Two services in one repo (or two repos)
    a. Pricing-api/ (Task 1)
    b. property-portal/ (backend + frontend)
2- Clear setup instructions (docker-compose optional but appreciated)
3- Source code in github
4- Ability to run full stack locally during interview

**Task 3**: System Design (Optional Advanced Task)

**Objective**

Design a cloud-ready microservice architecture for this system.

**Requirements**

Create a diagram and brief explanation covering:

1- Service decomposition: How would you split Pricing API, Portal Backend, Frontend?
2- Communication: Service discovery, API gateways, network security
3- Data flow: From user input to prediction response
4- Deployment: Container orchestration, CI/CD, monitoring

**Deliverables**

1- Architecture diagram
2- Short document (<1 page) explaining key decisions