






✓ 3. 전체 페이지 구성안

탭 번호	이름	설명
 1	공정 변수 이상 탐지	비지도 이상 탐지 결과 시각화 (Z-score, PCA, DBSCAN 등)
 2	불량 예측 시뮬레이터	사용자 입력 or 실시간 입력으로 불량 예측 (XGBoost 등)
 3	예측 근거 및 변수 중요도	SHAP, PI, PDP 등 시각화로 예측 원인 분석
 4	실시간 공정 모니터링	센서 트렌드, 품질 상태, 이상 탐지 점수 실시간 표시
 5	종합 리포트 & 모델 성능	F1, confusion matrix, 변수 상관관계, 모델 비교 등 종합 분석 페이지

✓ 4. 각 페이지 상세 구성

[1] 공정 변수 이상 탐지 페이지

목적: Z-score, PCA, DBSCAN 등으로 이상 공정 탐지

- `st.selectbox("이상 탐지 기법 선택", ["Z-score", "PCA", "DBSCAN"])`
- 이상 점수 plot (예: 이상 점수 top-N, 시계열 변화)
- 주요 센서 기준 이상값 마킹 (ex. 용탕온도가 700도 넘는 시점 표시)
- `st.line_chart()` 또는 `plotly.line()`로 시각화

💡 팁: 결과가 "이상공정"이면 빨간 점으로 표시 (색상 코드 활용)

[2] 불량 예측 시뮬레이터

목적: 사용자가 입력한 값 or 새로 들어온 센서값으로 실시간 예측

- 여러 `st.number_input()`을 통해 센서값 입력 받기
- `model.predict()` / `model.predict_proba()` 호출
- 예측 결과 출력 (`st.success("양품")`, `st.error("불량 가능성 높음")`)
- 입력값 저장 기능 (optional: `session_state` 활용)

💡 팁: `st.slider()`나 `st.form()` 활용해서 UX 좋게 만들기

[3] 예측 근거 및 변수 중요도 페이지

목적: SHAP, Permutation Importance, PDP 등을 통한 설명력 제공

- `st.selectbox("분석 기법 선택", ["SHAP", "Permutation Importance", "PDP"])`
- SHAP summary plot, force plot 시각화
- `st.selectbox("샘플 선택")` → 해당 샘플의 SHAP force plot 보여주기
- 주요 영향 변수 Top-N 표시 (막대그래프)

💡 팁: SHAP은 Tree 모델과 궁합이 좋아서 XGBoost/LGBM과 함께 사용하면 완벽함

☑ [4] 실시간 공정 모니터링 페이지

목적: 센서값 트렌드, 불량률, 이상 탐지 score 등 실시간 관찰

- `st.line_chart()` → 센서 시계열 (ex. molten_temp)
- 실시간 불량률 변화 (`st.metric()` 또는 `plotly.line()`)
- 이상 감지 알림 (`st.warning("이상 감지: 용탕온도 730도 이상")`)
- 작업자용 요약 표 (`st.dataframe()`)

💡 팁: `streamlit_autorefresh()`로 주기적 업데이트 가능 (약 5~10초마다)

📄 [5] 종합 리포트 & 성능 비교

목적: 전체 모델 성능과 특징 요약

- confusion matrix 시각화 (seaborn heatmap)
- F1, precision, recall score
- 모델별 성능 비교 표 (예: XGBoost vs LightGBM)
- `st.download_button()`으로 리포트 CSV 추출 제공

✓ 5. 디렉토리 구조 추천

```
pjt05/
├── app.py           ← Streamlit 메인 앱
├── model/
│   ├── classifier.pkl ← 학습된 모델
│   └── shap_values.npy ← SHAP 값 저장
├── data/
│   ├── train.csv
│   └── test.csv
├── pages/
│   ├── 1_anomaly.py  ← 이상 탐지 페이지
│   ├── 2_predictor.py ← 불량 예측 페이지
│   ├── 3_xai.py      ← 변수 영향 분석
│   ├── 4_monitoring.py ← 실시간 대시보드
│   └── 5_report.py   ← 성능 분석
```

🔗 `app.py`에서 sidebar로 페이지 선택 → `pages/`에 있는 각 페이지 불러오는 구조

✓ 마무리 요약

질문

답변

Streamlit vs Shiny? ☒ Streamlit이 훨씬 적합

페이지 수 5개 탭 권장

질문	답변
구성	이상탐지 / 예측 / 설명 / 모니터링 / 성능리포트
각 탭 구현 방식	<code>st.tabs()</code> 또는 <code>st.sidebar.selectbox()</code> + <code>if-else</code>