**1. Backend & Modeling**

| **Component** | **Purpose** | **Recommended Tech** | **Notes** |
| --- | --- | --- | --- |
| **Backend Framework** | Serve APIs, manage requests, handle data | **Flask (Python)** | Lightweight, easy to integrate with ML models. |
| **Predictive Modeling** | Temporal pattern learning | **LSTM (TensorFlow/Keras or PyTorch)** | Captures sequential data (30/60/90-day windows). |
| **Risk Classification** | Stratify members into tiers | **XGBoost (Python)** | Combines LSTM embeddings with static features. |
| **Data Handling / Preprocessing** | Cleaning, windowing, feature engineering | **Pandas & NumPy** | Widely used, easy integration with ML models. |
| **File Storage / IO** | Store member risk scores, intervention mapping | CSV / JSON files | Lightweight for a prototype; supports dashboard visualization. |

**2. Frontend & Visualization**

| **Component** | **Purpose** | **Recommended Tech** | **Notes** |
| --- | --- | --- | --- |
| **Frontend Framework** | Interactive dashboard | **React.js** | Popular, component-based, easy integration with APIs. |
| **Styling** | Layout and responsive design | **Tailwind CSS** | Lightweight, fast styling. |
| **Charts / Graphs** | Visualize risk tiers, trajectories, ROI | **Recharts or Chart.js** | Easy-to-use React chart libraries. |
| **Tabs & UI Components** | Tabbed dashboard for population vs member views | React component library (e.g., **Material UI or shadcn/ui**) | Provides ready-to-use components for tabs, tables, and cards. |
| **API Calls** | Fetch model predictions and risk files | **Axios** | Simplifies REST API communication. |

**3. Data / Prototype Infrastructure**

| **Component** | **Purpose** | **Recommended Tech** | **Notes** |
| --- | --- | --- | --- |
| **Database (optional)** | Store member data if needed | SQLite / MySQL | **SQLite is lightweight and perfect for a prototype; MySQL if you want DB integration.** |
| **Synthetic Data Generation** | Pretrain models safely | Synthea (or custom scripts in Python) | **Generates realistic but synthetic healthcare data.** |
| **Version Control** | Manage prototype code | Git + GitHub | **Essential for collaboration and tracking changes.** |

**4. Integration Flow**

1. **Data Ingestion:** Upload CSV or connect to DB → preprocess in Pandas.
2. **Model Prediction:** Run LSTM + XGBoost → generate **risk scores for 30/60/90-day windows**.
3. **Intervention Mapping:** Map risk tiers to **conceptual interventions** → generate CSV/JSON.
4. **Frontend Dashboard:** React fetches risk and intervention files via Flask API → displays:
   * Tab 1: Population Overview (risk distribution, trajectories, feature importance, ROI)
   * Tab 2: Member Details (individual risk scores, trajectory, suggested interventions)