**🧠 What Is This Project?**

**Project Name (Working Title):**  
**Clinical Trial Recruitment Optimizer using AWS Bedrock**

**Purpose / Use Case:**  
This application is designed to **streamline and optimize the process of patient recruitment for clinical trials**. It uses **AWS Bedrock’s GenAI capabilities** (specifically Claude) to:

1. Automatically **extract eligibility criteria** (inclusion/exclusion) from clinical trial protocols (text files),
2. Evaluate and **score patients** from a database (CSV) against the criteria,
3. **Visualize** results with insights like top matches, demographic distributions, and match score breakdowns,
4. Generate a **comprehensive report** that helps clinical researchers choose patients more effectively.

**🏥 Why Is This Useful to Saama?**

Saama Technologies operates in the **clinical analytics and life sciences AI** space. This project aligns perfectly with:

* **Core business goals**: Optimizing clinical trials, reducing recruitment timelines, automating medical document processing.
* **Current tech direction**: Using AWS Bedrock and GenAI (which your mentor likely wants you to become comfortable with).
* **Differentiation**: Shows how Bedrock can be operationalized, not just as a chatbot but as a **data integration and decision-making tool**.

**🧠 High-Level Idea**

Clinical trials often require strict criteria (like age, gender, specific conditions, etc.). Manually matching patients to trials is time-consuming.

Your app solves this by:

1. **Uploading a protocol document** (usually written in unstructured clinical language).
2. **Using Bedrock Claude model** to extract structured eligibility rules from it.
3. **Uploading a patient CSV database**.
4. **Automatically scoring and ranking patients** based on how well they match the criteria.
5. **Visualizing results** and **generating a comprehensive markdown report**.

**🔧 Technical Architecture**

**✅ Frontend (UI)**

* Built in **Streamlit** (easy-to-use Python web framework).
* Sidebar for AWS credential input.
* Upload sections for:
  + Clinical trial protocol .txt
  + Patient database .csv
* Interactive visual tabs for:
  + Score distribution
  + Demographics
  + Criteria stats
  + Advanced stats

**🧠 Backend (Logic)**

* ClinicalTrialAgent is the main class:
  + **extract\_eligibility\_criteria()**: Uses Claude via Bedrock to extract structured JSON.
  + **Fallback regex-based extraction** if Claude fails.
  + **score\_patients()**: Computes a percentage score based on how well each patient matches.
  + **generate\_report()**: Summarizes the results into a readable Markdown report.
  + **create\_visualizations()**: Makes graphs like bar charts, histograms, pie charts.
  + **create\_matplotlib\_visualizations()**: Advanced statistical plots using Seaborn/Matplotlib.

**☁️ AWS Bedrock**

* Claude 3 Sonnet is used to:
  + Understand and parse complex clinical protocol text.
  + Output eligibility criteria in structured JSON format.

**🔍 Key Features**

**🧾 Protocol Understanding via Claude 3**

* Extracts:
  + Inclusion/Exclusion criteria
  + Age/gender limits
  + Required medical conditions
  + Medication restrictions

**🧬 Patient Matching**

* Matches against:
  + Age range
  + Gender
  + Known medical conditions
  + Specific keywords from inclusion/exclusion criteria

**📊 Visual Analysis**

* Match scores by patient
* Criteria met vs unmet
* Score distributions
* Demographics (age, gender breakdown)

**📄 Reporting**

* Downloadable report in **Markdown format** summarizing:
  + Criteria
  + Top patients
  + Visual summaries
  + Match score analysis

**🧪 Example Workflow**

1. Upload protocol → “Inclusion: Must be 18-65, no cancer history, must have diabetes”
2. Upload patient CSV → John Smith, 45, male, has diabetes, no cancer
3. Agent:
   * Parses the protocol with Bedrock
   * Scores John 100% match
4. Outputs dashboard:
   * John ranked #1
   * Bar charts, match % pie chart
   * Downloadable report