**Project Report: Mr.HelpMate AI**

**By: Abhishek Paul**

**1. Objectives**

• **Goal**: Build a robust generative search system capable of effectively and accurately answering questions from a policy document.

• Illustrate the integration of OpenAI’s language models with Chroma.

• Showcase practical applications of generative search in various domains.

• Evaluate the performance and accuracy of the search results.

**2. Design**

• **System Architecture**:

• **Components**:

• OpenAI Language Models: Used for generating natural language responses based on input queries.

• Chroma: Provides a chromatic analysis and context for the search results.

• **Workflow**:

1. User inputs a query.

2. The query is processed by OpenAI’s language model.

3. Chroma provides contextual analysis to enhance the search results.

4. The results are presented to the user.

• **Data Flow**:

• Input: User query.

• Processing: Query analysis, context generation, and response synthesis.

• Output: Enhanced search results.

**3. Implementation**

• **Environment Setup**:

• Tools and Libraries: Python, PdfPlumber, Jupyter Notebook, OpenAI API, Chroma.

**4. Challenges**

• **Technical Challenges**:

• API Integration: Issues faced with generating and using openai key while integrating OpenAI API and Chroma.

• Data Handling: Challenges in managing and preprocessing large datasets.

• Performance: Ensuring the system responds in a timely manner.

• **Operational Challenges**:

• Resource Management: Efficiently using computational resources.

• Error Handling: Managing and debugging errors during execution.

**5. Lessons Learned**

• **Technical Insights**:

• Importance of robust API integration.

• Effective data preprocessing techniques.

• Optimizing model performance for faster responses.