Intelligent Reasoning Systems Practice Module

Aegis AI
Recommender System

Group 40

Choy Yong Yi Desmond (A0315402W)

Kenny Lau Jia Xu (A0179912U)

Kevin Manuel (A0315373H)

Soon Fu Meng (A0140502B)

Table of Contents

Executive Summary

Singaporean travelers face significant challenges selecting suitable travel insurance due to policy complexity, lack of transparency, and comparison fatigue. This project introduces Aegis AI, an intelligent recommendation system designed to address these pertinent issues.

In this report, the team will share details on how it has successfully developed and evaluated an LLM-driven workflow applying **Intelligent Reasoning Systems principles** such as automated knowledge representation, hybrid reasoning models, and semantic evaluation to the complex domain of travel insurance recommendation.

Quantitative evaluations demonstrated strong performance: the system achieved high pass rates, ranging from 85% to 100%, in recommending appropriate policies across four distinct and challenging test scenarios designed to probe specific coverage nuances. Furthermore, it successfully covered a high percentage (87.5%) of valid customer requirements identified across 80 diverse test cases, verified through ground truth analysis.

The project showcased the significant advancements in multi-modal LLMs, specifically Gemini 2.5 Pro, for **accurate knowledge extraction** from notoriously complex and variably formatted PDF policy documents. This capability proved to be a critical enabler for the entire reasoning pipeline, ensuring that subsequent comparisons and recommendations were based on reliable, structured data. Additionally, the **hybrid recommendation approach, combining initial quantitative scoring with LLM-driven qualitative re-ranking informed by transcript context**, proved effective in balancing objective requirement matching with nuanced contextual understanding.

The system demonstrably addresses key consumer pain points like policy complexity and lack of transparency by automating the laborious analysis of fine print and presenting comparisons clearly, representing a valuable application of AI reasoning techniques to improve decision-making in the insurance domain.

Introduction

Background & Problem Statement

Travel insurance is a specialized form of insurance covering financial losses that can arise from unforeseen events occurring before or during a trip, applicable to both domestic and international journeys[1]. Its importance extends beyond mere convenience; it addresses critical financial exposures. A key area is overseas

medical emergencies. Standard domestic health insurance plans, such as Medicare in the US or Singapore's MediShield Life, typically offer limited or no coverage outside the traveler's home country[2].

Singaporean travelers increasingly recognize travel insurance as essential, particularly following the widespread travel disruptions experienced during the COVID-19 pandemic[3]. This heightened awareness of risks like cancellations, delays, health emergencies, and even geopolitical instability has driven demand[4]. Notably, there's a shift in consumer behavior, with more travelers purchasing insurance even for short trips to nearby destinations, which were previously perceived as lower risk.

Despite the acknowledged need, Singaporean travelers face significant pain points and challenges in the current market:

- **Complexity and Lack of Transparency:** A major source of frustration is the difficulty in understanding policy details. Travelers struggle to decipher what is covered, the applicable limits and conditions, and, crucially, what is excluded[5]. This lack of transparency can lead to confusion and dissatisfaction, particularly when claims are denied based on poorly understood terms. **This presents a fundamental challenge: consumers demand clear, transparent coverage[5], yet insurers rely on detailed exclusions and conditions to manage risk (like moral hazard and adverse selection) and maintain affordability[3].** This inherent tension between the need for contractual precision and the user's desire for straightforward understanding creates a significant gap that technology could potentially bridge by accurately interpreting and explaining complex terms in simple language.
- **Policy Exclusions:** Directly linked to transparency issues, the limitations imposed by policy exclusions are cited as a top challenge, especially by Generation X travelers (42-56 years old)[5]. Unexpected claim denials due to exclusions that were not fully understood at the time of purchase are a common complaint[6].
- **Outdated "Cookie-Cutter" Approach:** Traditional travel insurance products are increasingly seen as outdated and ill-suited to the needs of modern travelers, particularly Millennials and Gen Z[7]. These demographics seek more progressive, flexible, and even enjoyable purchase experiences, rather than standardized, inflexible policies[7].
- **Delayed Purchase Decision:** The cumulative effect of these pain points – complexity, lack of transparency, difficult claims, and price sensitivity – creates significant friction in the purchasing process. Consequently, many Singaporean travelers, even those who plan other aspects of their trips meticulously, delay buying travel insurance until just days or a week before departure. This procrastination suggests that the perceived hassle of

navigating the confusing landscape outweighs the perceived immediate need for coverage until the trip is imminent, highlighting a substantial opportunity for a solution that drastically simplifies the process and reduces this friction.

Objectives & Success Criteria

The primary aim of this project is to mitigate the complexity consumers face when selecting travel insurance by developing an intelligent, Large Language Models (LLMs)-powered workflow. Key objectives include:

- **Automation of Data Extraction**: To automate the extraction of critical data from dense policy documents and (simulated) customer interaction transcripts. This involves parsing policy PDFs into structured JSON and converting customer needs from transcripts into a validated, structured format.
- **Personalized Recommendation Generation**: To enable rapid, data-driven comparison of insurance policies against extracted customer requirements and generate personalized, justified recommendations for the most suitable policy tier.
- **Enhanced Transparency and Efficiency**: To unlock unprecedented efficiency and transparency in the insurance selection process through structured data extraction and automated analysis.
- **Application of AI and Reasoning techniques**: To apply advanced AI and LLM techniques to address real-world challenges within the insurance domain.

Given the inherent non-determinism and potential for factual inaccuracies (hallucinations) in LLMs, a robust and comprehensive evaluation framework is essential to ensure the reliability, accuracy, and trustworthiness of the multi-stage data extraction and recommendation process. This rigorous validation is necessary to mitigate the risks associated with providing incorrect or poorly justified insurance advice, which could have significant negative consequences for consumers.

Hence, the project's success is measured through a multi-faceted evaluation framework designed to ensure data quality and the accuracy of the recommendation process. The core components of this evaluation, which serve as success criteria, include:

- **Transcript Quality Evaluation**: Assessing the quality, relevance, and completeness of synthetically generated customer transcripts using LLM-based analysis before they are processed further. This ensures the

input data for requirement extraction is realistic and covers necessary ground.

- **Policy Data Extraction Accuracy**: Verifying the accuracy and completeness of structured data extracted from policy PDFs against the source documents using multi-modal LLM comparison.
- **Comparison Report Quality**: Evaluating the accuracy and factual grounding of the generated policy comparison reports by comparing them against the source policy documents and customer requirements using an LLM.
- **Requirement Coverage Validation**: Quantifying how well the final recommended policy covers the individual requirements stated by the customer. This involves using semantic matching techniques against a curated ground truth knowledge base (data/ground_truth/ground_truth.json) to determine the coverage status (Covered, Not Covered, Requirement Not Found) for each requirement.
- **Scenario-Based Recommendation Accuracy**: Assessing the appropriateness of the overall final recommended policy by comparing it against predefined expected outcomes for specific test scenarios (e.g., golf coverage, pet care coverage) stored within the ground truth knowledge base. This validates the system's ability to handle specific, known situations correctly.
- **Scenario Pass Rate Calculation**: Measuring the overall success rate across different test scenarios by programmatically calculating the pass rates based on the scenario recommendation evaluations.

Business Case / Market Research

Industry & Stakeholder Analysis

The travel insurance market in Singapore and the wider Southeast Asian region is experiencing robust growth, presenting a fertile ground for innovative solutions.

Table 1: Singapore & SEA Travel Insurance Market Overview

	Singapore	Southeast Asia (SEA)
Market Size	USD 143.2M (2023)[8]	USD 356.22M (2019)[9]
Projected Size	USD 646.7M (2030)[8]	USD 1.39B (2032)[9]
CAGR	22.6% (2024-2030)[8]	13.9% (2024-2032)[9]
Key Growth Drivers	- Increased travel activity - Digital platform growth[8] - Heightened risk awareness[10] - Marketing & awareness efforts[8]	- Increased travel activity[9] - Digital platform adoption[9] - Demand for customization[9] - Integration with booking platforms [9] - Growth in adventure tourism[9]
Dominant Segments	- Distribution: Intermediaries/Online (Growth)[9] - End User: Family/Leisure Travelers[11] - Coverage: Single-Trip[12]	- Distribution: Insurance Intermediaries[11] - End User: Family Travelers (2019)[11] - Coverage: Single-Trip[12] - Mode: Online (Highest Growth)[11]

Singapore's advanced digital infrastructure and tech-savvy population have enabled insurers to offer innovative solutions (like online claims and real-time assistance), contributing to high adoption rates[11]. In fact, even back in 2017 a survey found nearly *72% of Singapore travelers* "always or often" purchase travel insurance for leisure trips, a figure that likely grew further after COVID-19 due to heightened safety awareness. This indicates that individual travelers (the primary target users of our recommendation system) are already widely convinced of the need for travel insurance, especially in Singapore and the broader SEA region.

Market Size & Growth

- **Singapore:** The domestic market demonstrated significant value, estimated at

USD 143.2 million in 2023. Projections anticipate strong expansion, reaching USD 646.7 million by 2030, reflecting a compound annual growth rate (CAGR) of 22.6%[8]. Historically, Singapore has been a dominant force, holding the largest market share in Southeast Asia in 2019[11].

- **Southeast Asia (SEA):** The regional market was valued at USD 356.22 million in 2019. It is forecast to grow substantially to USD 1.39 billion by 2032, indicating a healthy CAGR of 13.9% over the forecast period (2024-2032)[11].
- **Asia Pacific (APAC):** As the broader regional context, APAC represents the largest travel insurance market globally, accounting for 39.1% of the share in 2024 with an estimated value of USD 9.3 billion[12]. Size estimates for 2023 vary, ranging from USD 5.49 billion[13] to USD 10.3 billion[14], with projected growth reaching approximately USD 24-25 billion by 2030/2033, driven by CAGRs estimated between 9.6% and 22.0%[13].
- **Global:** The global market is also on a strong upward trajectory, projected to increase from USD 24.0 billion in 2024 to USD 120.5 billion by 2034, representing a CAGR of 17.5%[12].

Key Growth Drivers

Several factors are fueling this expansion:

- **Increased Travel Activity:** A surge in both international and domestic travel, encompassing leisure and business segments, is a primary driver[9]. Post-pandemic "revenge travel" has been particularly noted in Singapore.
- **Economic Growth & Disposable Income:** Rising disposable incomes, especially within the expanding middle class in emerging APAC economies like parts of Southeast Asia, enable more people to travel internationally[12].
- **Heightened Risk Awareness:** Increased consciousness among travelers regarding potential risks – including trip cancellations, medical emergencies abroad, baggage loss, safety concerns, and health risks like pandemics – drives demand for protection[9].
- **Digitalization:** The proliferation of online platforms, comparison websites, and mobile apps has significantly improved accessibility and convenience, making it easier for consumers to research, compare, and purchase policies[8]. Younger demographics heavily utilize digital channels for purchasing insurance.
- **Evolving Needs:** Demand for specific coverage types, such as comprehensive COVID-19 benefits or policies covering adventure tourism activities, caters to modern travel patterns[9].
- **Integration:** Embedding travel insurance offers within airline booking flows, online travel agencies (OTAs), and banking platforms simplifies the purchase process[9].

Key Trends Shaping the Market

- **Digital Transformation:** The industry is rapidly embracing digitalization. This includes the shift to online sales and comparison portals, the development of insurer mobile apps (like FWD's FWDFLYER[15]) for policy management and claims, digital claims processing, and the provision of real-time assistance services.[1] Insurers globally are focusing on digitalization efforts[10].
- **Personalization and Customization:** There's a clear trend towards offering more tailored insurance solutions. This involves customizing plans based on individual traveler needs, specific travel styles (e.g., adventure travel, 'bleisure' combining business and leisure), demographics (especially younger generations), and desired coverage levels[9].
- **Embedded Insurance:** Integrating travel insurance purchase options seamlessly into the booking processes of airlines, OTAs, and banks is becoming increasingly common[9]. Research indicates a strong consumer preference for this convenience, with 75% of travelers preferring to book insurance alongside flights and accommodation[7].
- **Enhanced Customer Experience (CX):** Insurers are focusing on improving the overall customer journey. This includes streamlining the purchase process, offering faster and more efficient claims handling[10], and providing value-added services like parametric payouts for delays, airport lounge access, or real-time safety alerts[16]. Positive claims experiences are recognized as a key driver of customer loyalty.
- **Addressing Evolving Risks:** The market is adapting to cover emerging travel risks beyond traditional concerns. This includes issues related to technology dependence (e.g., phone battery life, internet access in remote areas), new scam types (e.g., ride-share scams), misrepresentation of online bookings, variable safety standards from providers, climate-related disruptions, evolving health risks, and geopolitical instability[7].
- **Sustainability Focus:** Environmental, Social, and Governance (ESG) considerations are becoming increasingly important for travel insurers[10].

Key **stakeholders** in the travel insurance ecosystem include:

- **Individual Travelers (Consumers)** – They seek adequate coverage at a reasonable price and a hassle-free buying experience. Their pain points revolve around understanding coverage, comparing options, and trust in the insurer or policy.
- **Insurance Providers (Insurers)** – These are companies underwriting travel insurance (e.g. global players like Allianz, AIG, or regional ones like Chubb, NTUC Income in Singapore). They are interested in reaching more customers, differentiating their products, and managing risk exposure. Insurers in competitive markets like Singapore are increasingly leveraging

technology (digital sales, automated claims) to attract customers[9].

- **Distribution Channels** – This includes travel insurance aggregators, comparison websites, travel agencies, airlines, and brokerages that sell policies on behalf of insurers. In Singapore and SEA, **online platforms** and intermediaries have become a major distribution channel, improving consumer access and intensifying competition among insurers[8]. For example, airlines and online travel agencies (OTAs) often offer insurance add-ons during flight bookings, and independent aggregator sites allow users to compare quotes from multiple insurers.
- **Regulators** – Insurance is a regulated industry. Bodies like the Monetary Authority of Singapore (MAS) set guidelines for fair marketing, disclosure of terms, and licensing of those giving advice. Any recommendation system must ensure compliance (e.g. providing disclaimers that it is informational and ensuring transparency in how recommendations are generated). Regulators also focus on consumer protection – requiring that advice (even if automated) is suitable and that data privacy is maintained.
- **Potential B2B Clients** – In a future B2B model, stakeholders could include insurance companies or financial institutions using the system to enhance their customer offerings, as well as corporate travel management companies or large travel agencies seeking to bundle smarter insurance recommendations for their customers. These stakeholders will value the system's accuracy and efficiency in customizing insurance offers, as it could improve their sales conversion or customer satisfaction.

Pain Points & Opportunity Sizing

Despite healthy market growth, there remain pain points (from the consumer perspective) that translate into business opportunities for a better solution. Many travelers, especially in SEA, still perceive buying insurance as confusing or worry that it's not worth the extra cost[8]. Price sensitivity is a notable barrier – travelers on a budget might view insurance as an unnecessary add-on, particularly for short or inexpensive trips[8]. This is often exacerbated by a lack of understanding: if customers aren't clearly aware of the benefits or have had no issues in past trips, they may opt out.

In Singapore's market analysis, it's noted that **perceived high cost and lack of transparency in coverage can lead individuals to skip purchasing coverage**, even though doing so exposes them to significant financial risk[8]. This underscores an opportunity to better educate and inform travelers at the point of sale, highlighting the value for money of a well-chosen policy.

Another pain point is **analysis paralysis** – with dozens of travel insurance plans available (from basic plans to premium ones with myriad optional benefits), travelers can be overwhelmed. This often results in either a rushed decision (e.g. picking the cheapest or a familiar brand without fully evaluating coverage details) or abandonment of purchase. For instance, a U.S. study found that about 60% of travelers now use online comparison websites to help in purchasing travel insurance, which indicates consumers actively seek tools to simplify decision-making. However, traditional comparison sites still require the user to manually read through and compare policy details. If those details are not clearly presented, important factors like coverage limits or exclusions might be overlooked. **A smarter recommendation system that can parse policy PDFs and match them to a customer's specific needs (e.g. "Does this plan cover my scuba diving activity?") addresses a real gap in the user experience.**

The size of the opportunity is reflected in how many travelers could benefit from such a tool. Travel insurance uptake is high among older travelers but drops among younger ones – globally, approximately 45% of travelers aged 18–29 purchase travel insurance, compared to ~60% of travelers in their 40s and 50s[17]. Younger travelers often assume their risk is low or rely on minimal coverage from credit cards [17]. Converting a portion of this hesitant segment by using a user-friendly, educational recommendation approach could significantly expand policy sales. Moreover, with international travel rebounding strongly post-pandemic (94% of American adults, for example, planned a trip in the first half of 2024[17], there is a surge of potential customers seeking insurance. Many of these travelers are newly aware of issues like trip cancellations due to health emergencies or border closures – **they have *fresh pain points* that a tailored insurance recommendation can address (for example, ensuring COVID-19 coverage or coverage for sudden trip disruptions).** This growing demand, combined with existing pain points, represents a sizable opportunity.

In Singapore alone, the travel insurance market is projected to grow over 4.5× from USD 143 million in 2023 to USD 646 million by 2030[18], indicating hundreds of thousands of policies sold each year. **Capturing even a small fraction of this market through a superior recommendation experience could translate into substantial revenue.**

The analysis of market dynamics, consumer perspectives, and the competitive landscape reveals distinct gaps and opportunities that the proposed LLM-powered recommendation system is well-positioned to address.

- **Identified Market Gaps:** The core frustrations revolve around the inherent **complexity** of insurance policies, the **lack of transparency** regarding what is truly covered versus excluded, the **difficulty and length of the claims process**, and a desire for greater **personalization** that traditional products

often fail to deliver[16]. The friction involved in navigating this landscape contributes to delayed purchasing decisions[7].

- **The "Comparison Fatigue" Gap:** Existing comparison platforms, while helpful for initial discovery and price sorting[19,20], often leave users facing "comparison fatigue." They present headline figures and promotional offers, but the onus remains on the user to delve into the complex policy documents (the PDFs the system aims to analyze) to understand the critical details, limitations, and exclusions[19]. This burden means aggregators only partially solve the problem, failing to address the deeper need for comprehension.
- **The "True Suitability" Gap:** Current comparison methods heavily emphasize price and major benefit categories (e.g., overall medical limit, trip cancellation limit)[20]. A significant gap exists in matching the *nuances* of policy wordings – specific sub-limits, definitions of covered activities (like adventure sports[21]), precise conditions for claims (e.g., delay duration thresholds), and detailed exclusion clauses[3] – to the *specific circumstances and requirements* of an individual traveler's planned trip. Assessing true suitability requires understanding these details, not just comparing top-level numbers.

Competitive / Alternative Solutions

The competitive landscape for travel insurance recommendations ranges from direct competitors (if any) offering AI-driven advisory, to indirect competitors like comparison sites and traditional channels. It's important to examine both direct competition and broader alternative solutions that customers use to meet the same need:

- Insurance Comparison Platforms (Direct Competitors): These include independent online platforms that let users compare quotes and coverage from multiple insurers. Globally, examples are sites like *Squaremouth*, *InsureMyTrip*, or *Insubuy* (in the U.S.), and regionally in Asia, sites like *PolicyPal*, *GoBear* (which was active in Singapore/Malaysia), or *MoneySmart*. Such platforms are popular – *62% of consumers* used online platforms to compare plans before purchasing travel insurance as of 2022[17] – and they address the basic challenge of shopping around for price and coverage. However, their interfaces are typically form-based and list-oriented. They may not delve into nuanced policy wording or personalize the experience beyond filters (destination, dates, traveler age, etc.). There is often still a burden on the user to read the fine print. Our proposed system, by using an LLM to actually read policy PDFs and customer queries, would go a step further by giving conversational, context-specific advice. To our knowledge, fully conversational insurance recommendation (where a user can ask, for example, "I'm going hiking in Nepal, is altitude sickness covered?" and get a precise answer) is an emerging capability – likely not yet offered by most

mainstream aggregators, which presents a competitive edge.

- Insurance Aggregators & Brokers (Broad Competitors): Traditional insurance brokers or large aggregator brands can also be considered competitors. In Singapore and SEA, banks or insurance brokers sometimes package travel insurance from multiple providers; similarly, aggregator brands (like *CompareFirst* or *Agents* who have access to multiple insurers) can advise customers. These channels often involve human agents or static comparison tables. They do provide a level of personalized recommendation (a human advisor asking about your trip and suggesting a plan), but they may not always be convenient or unbiased. Additionally, because human agents are involved, this route may incur extra fees or take longer (scheduling a consultation). Our AI system could offer similar personalized guidance instantly and for free (to the consumer), which could outcompete traditional brokers for tech-savvy users seeking quick answers.
- Embedded Insurance via Travel Providers: A large portion of travel insurance is sold as an add-on when booking flights, tours, or hotel packages. Online Travel Agencies (OTAs) like Expedia or Booking.com, and airlines like Singapore Airlines, often partner with a single insurer to offer a policy during checkout. While convenient, the choice is limited to that partner's policy. Travelers who purchase insurance this way might miss out on a plan that is better suited or cheaper from another provider. These embedded offers are essentially a one-size-fits-all *default recommendation* (usually covering basic trip cancellation and some medical coverage). The lack of choice and potential bias (only one insurer's product) is a drawback, and it's here that an independent recommendation system can provide value – by scanning multiple options across insurers to find the truly best match for the customer's needs, not just what a single travel provider is selling.
- Do Nothing / Self-Insurance (Indirect Alternative): Unfortunately, an alternative many travelers choose is to not buy any travel insurance, or to rely on free coverage such as credit card travel insurance. This is the status quo competitor we aim to change. Travelers skip insurance either due to cost, lack of awareness, or the hassle factor. By addressing the hassle and clearly quantifying the value (e.g., demonstrating what benefits they'd gain for the price), our system can convert more of these uninsured travelers into insured customers. For instance, if the AI assistant can transparently show: "Your $3,000 trip could be protected for $150, covering you for up to $50,000 in medical emergencies, etc.," it may persuade someone who otherwise wouldn't bother. This not only grows business but also fulfills a consumer protection role by reducing the number of people traveling with no safety net.

In summary, **while there are numerous avenues for purchasing travel insurance, there is no dominant personalized recommendation platform yet** – most solutions either focus on price comparison or one-click add-ons. The competitive advantage of our proposed system lies in its multi-stage AI pipeline that can truly understand individual customer conversations (needs/fears) and the fine details of policy documents, then make *transparent* recommendations. It can co-exist with existing channels (for example, such an AI could even be offered *through* an aggregator site or an airline's app to enhance their experience). Notably, insurance companies themselves are exploring AI: 82% of insurance companies report relying on AI to better understand their customers[22], which suggests that if we develop a successful recommendation engine, insurers or brokers might choose to partner with or license it (rather than build their own from scratch). This leads to a compelling value proposition and ROI discussion for our solution.

Value Proposition & ROI

The core value proposition of the LLM-powered travel insurance recommendation system should be clearly articulated to resonate with the identified needs of Singaporean travelers: **"Effortlessly understand and compare travel insurance policies based on your unique needs, ensuring you get the *right coverage*, not just a cheap price. We decode the fine print, so you don't have to."** This proposition directly addresses the core market frustrations and leverages the system's unique capabilities.

- **Simplicity:** This is achieved through multiple facets of the system's design. LLMs enable natural language interaction, potentially through conversational interfaces, making requirement gathering intuitive. Very manual and tedious tasks like comparing policy jargon and reading dense PDFs have been automated. The final output – clear, concise summary reports and comparisons – drastically reduces the cognitive load on the user compared to navigating complex aggregator filters or deciphering policy documents themselves[16].
- **Transparency:** The system fosters transparency by going beyond headline figures. It leverages LLMs to identify and surface key exclusions, conditions, sub-limits, and definitions that are directly relevant to the user's specific travel scenario. Crucially, it aims to explain these potentially complex terms in simple, understandable language. By providing side-by-side comparisons of these *meaningful* coverage details, it empowers users to see the real differences between policies, directly tackling the pain points of opaque coverage and confusing terms.[3] The use of a Ground Truth Knowledge Base for validation underpins this transparency with factual grounding.
- **Accuracy:** Accuracy is paramount and is built into the system's architecture.

The latest and most capable LLMs (Gemini 2.5 Pro) are leveraged. Coupled with detailed instructions within the prompt, we have ensured fidelity to the source documents. Continuous validation against the Ground Truth Knowledge Base minimizes errors. The LLM's role in reranking adds a layer of qualitative assessment, ensuring recommendations are not just quantitatively matched but also contextually relevant based on a deeper understanding of policy language.

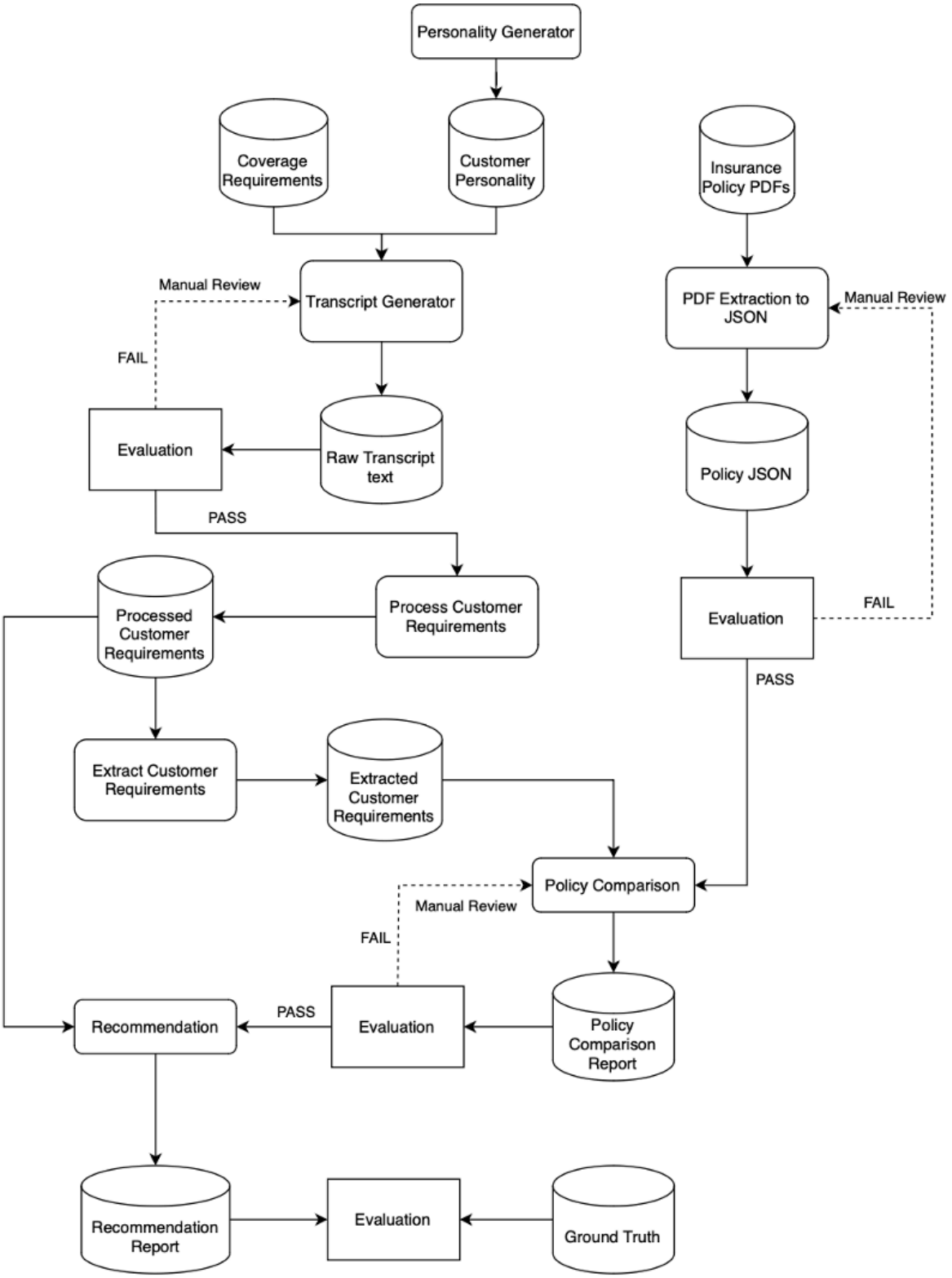
The primary **benefit for the target user is a significant reduction in time and effort spent researching and comparing insurance**. It alleviates the confusion and stress associated with complex policies, leading to increased confidence in the final purchase decision. Most importantly, by focusing on suitability and clearly highlighting relevant coverage details and exclusions, it minimizes the risk of purchasing inadequate insurance, thereby preventing potential financial losses and disappointment during travel[7].

The **potential Return on Investment (ROI)** can be viewed from multiple perspectives:

- **For Users:** The ROI is measured in time saved, reduced cognitive effort and stress, and enhanced financial security through better-matched insurance coverage, potentially avoiding significant out-of-pocket expenses from uncovered events.
- **For the Business:** Success translates into high user engagement and satisfaction. If employing a referral or partnership model, this can lead to strong conversion rates. The system offers significant differentiation in a crowded marketplace. Over time, the aggregated, anonymized data on user preferences versus policy features could become a valuable asset for B2B insights or licensing.
- **For Insurers (Potential B2B Value):** If partnerships are formed or B2B services offered, insurers could benefit from receiving better-qualified leads (users who understand the recommended product), gaining valuable insights into granular customer preferences and how they map to specific policy clauses, and potentially seeing a reduction in basic customer service inquiries regarding coverage details.

System Design

High-level Architecture Diagram



The system design revolves around a sequential data processing pipeline, clearly illustrated in the README's system diagram:

Category	Component / Task	Description
Data Generation	Personalities Generation	Creates customer service personality types
	Transcript Generation	Produces synthetic customer-agent conversations based on scenarios and requirements
Data Processing Pipeline	Transcript Parsing	Standardizes transcript format
	Requirement Extraction	Extracts structured customer requirements using CrewAI/OpenAI
	Policy Extraction	Converts PDF policies to structured JSON with detailed coverage information
Report Generation	Policy Comparison	Generates insurer-level comparison reports
	Recommendation	Two-stage process (scoring + LLM re-ranking) for final recommendations
Evaluation Framework	Transcript Evaluation	Assesses quality and coverage of generated transcripts
	PDF Extraction Evaluation	Verifies accuracy of policy extraction
	Comparison Report Evaluation	Checks quality of comparison reports
	Scenario Recommendation Evaluation	Tests final recommendations against ground truth
	Ground Truth Coverage	Evaluates how well recommended policies cover customer requirements
Core Services	LLM Service (Gemini)	Centralized interface for Google Gemini API
	OpenAI API	Used by Extractor Agent and EmbeddingMatcher
	Embedding Utilities	Semantic matching for evaluation

Orchestration	Orchestration Script	End-to-end workflow automation
	Demo Script	Simplified single-scenario execution
Frontend (aegis-recsys.netlify.app)	Landing Page	Entry point with customer ID input
	Disclaimer Page	Legal notices before viewing reports
	Report Viewer	Tabbed interface for recommendations, comparisons, requirements, and transcript
	Feedback UI	Collects user feedback on recommendations

Data Flow Summary

1. **Input Generation**: Scenarios, coverage requirements, and personalities guide transcript generation
2. **Transcript Processing**: Evaluation → Parsing → Extraction
3. **Policy Processing**: PDF → Structured JSON → Evaluation
4. **Report Generation**: Requirements + Policies → Comparison Reports → Recommendation Report
5. **Evaluation**: Multiple evaluation points throughout the pipeline
6. **Frontend Display**: Reports and data presented through React-based UI

Key Design Principles

- **Modularity**: Functionality is broken down into distinct scripts (Extractor, Analyzer, etc.) and scripts (data generation, extraction, comparison), promoting separation of concerns.
- **LLM-Centric:** LLMs (primarily Gemini via LLMService, also OpenAI via CrewAI) are core to multiple stages: data extraction from PDFs, synthetic data generation, transcript evaluation, requirement extraction, and policy comparison.
- **Data Structuring & Validation:** Emphasis on converting unstructured/semi-structured data (PDFs, conversations) into structured JSON using Pydantic models (PolicyExtraction, TravelInsuranceRequirement) for validation and reliable downstream processing.
- **Automation:** Scripts automate significant parts of the data preparation, processing, and comparison workflow.
- **Configurability**: The LLMService and GeminiConfig provide a way to manage API keys, models, and generation parameters centrally. The system

is designed as an LLM-driven pipeline to ingest policy documents and customer interactions, process them into structured formats, perform comparisons, and ultimately recommend suitable travel insurance policies.
- **Evaluation Framework:** Includes a dedicated step and scripts for evaluating the quality of generated transcripts against requirements.

Integration with NUS-ISS Semester 1 Curriculum

Decision Automation: Knowledge-Based Reasoning Techniques

This is a core strength of the system, implemented through:

- **Two-Stage Recommendation Logic**: The generate_recommendation_report.py script implements a hybrid approach combining:
    - **Stage 1**: Rule-based scoring system that evaluates policy matches against customer requirements
    - **Stage 2**: LLM-based re-ranking that applies contextual reasoning to the initial scores
- **Structured Knowledge Representation**: The system transforms unstructured policy documents into structured JSON with detailed coverage information, enabling systematic reasoning about policy features and limitations.
- **Ground Truth Knowledge Base**: data/ground_truth/ground_truth.json serves as a formalized knowledge base defining:
    - Which specific requirements are covered by which policy tiers
    - Expected policy recommendations for specific test scenarios

Business Resource Optimization: Informed Search Techniques

The system implements informed search through:

- **Policy Tier Selection**: The comparison script (generate_policy_comparison.py) performs an informed search across multiple policy tiers to identify the optimal tier per insurer based on customer requirements.
- **Recommendation Ranking**: The recommendation script uses a heuristic scoring function to rank policies based on requirement coverage and pricing, effectively implementing a form of informed search across the policy space.

- **Embedding-Based Search**: The EmbeddingMatcher implements a similarity-based search to find the most relevant ground truth entries for customer requirements, using semantic distance as the heuristic.

Knowledge Discovery & Data Mining Techniques

While not fully implemented yet, the system has foundations for knowledge discovery:

- **Requirement Extraction**: The Extractor Agent (src/agents/extractor.py) mines unstructured conversation transcripts to discover and structure customer requirements.
- **Policy Feature Extraction**: The policy extraction script (extract_policy_tier.py) mines complex PDF documents to discover and structure policy features, limits, and conditions.
- **Evaluation Analytics**: The system generates evaluation data across multiple components, providing a foundation for mining insights about system performance and recommendation patterns.

System Designed with Cognitive Techniques or Tools

- **Conversational Context Understanding**: The recommendation system considers the full conversation transcript when making final recommendations, demonstrating cognitive awareness of conversational context.
- **Transparent Reasoning**: The system provides detailed justifications for recommendations, showing the cognitive process behind decisions.
- **Multi-Modal Processing**: The evaluation components demonstrate cognitive capabilities by comparing textual data (JSON) against visual/textual documents (PDFs).

System Development & Implementation

Below, we elaborate on the technologies employed, the specific construction of each core component, the development process undertaken, and the challenges encountered in translating the design into a functional prototype. The implementation leverages a combination of Large Language Models (LLMs), agent frameworks, and custom scripts to automate the complex tasks of data generation, information extraction, policy comparison, and evaluation.

Development Environment & Technologies

The system was developed primarily using Python 3.11, chosen for its extensive libraries supporting data science, machine learning, and API interactions. Key technologies and frameworks underpinning the implementation include:

- **LLM Interaction:** Google Generative AI SDK for accessing Gemini models and OpenAI SDK (utilized via the CrewAI framework).
- **Agent Framework:** CrewAI was employed for implementing the autonomous Requirement Extraction agent, facilitating structured task execution.
- **Data Validation:** Pydantic was used extensively for defining data schemas and validating the structured outputs (JSON) from LLM processes and data transformations, ensuring data integrity throughout the pipeline.
- **Core LLM Services:** The system relies on Google's Gemini API (specifically `gemini-2.5-pro-exp-03-25` for most generation and extraction tasks due to its capabilities and experimental access) and OpenAI's API (`gpt-4o`, used by the CrewAI agent).
- **Embedding & Semantic Matching:** OpenAI's text embedding models were utilized for semantic comparison tasks within the evaluation framework, facilitated by custom utilities (`src/embedding/embedding_utils.py`). NLTK and Scikit-learn were included as dependencies, primarily supporting underlying functionalities of other libraries.
- **Web Frontend:** A user interface for demonstrating report viewing was developed using React, TypeScript, Material UI (MUI), and Vite, deployed via Netlify.
- **Development & Version Control:** Visual Studio Code served as the primary IDE, with Git and GitHub used for version control and collaboration

Component Implementation Details

The system comprises several interconnected components, each implemented primarily as Python scripts or modules, orchestrating LLM interactions and data processing tasks.

Data Generation

- **Personality Generation (scripts/data_generation/generate_personalities.py):** To inject variability into synthetic conversations, a script utilizes the Gemini LLM to generate a

list of common customer service personality types. Pydantic models validate the output, ensuring a structured JSON list is produced (`data/transcripts/personalities.json`).

- **Transcript Generation (`scripts/data_generation/generate_transcripts.py`):** This script automates the creation of synthetic customer-agent conversations. It takes predefined personalities, coverage requirements (`data/coverage_requirements/coverage_requirements.py`), and specific scenarios (`data/scenarios/`) as input. The Gemini LLM (gemini-2.5-pro-exp-03-25) generates conversations based on detailed prompts, refined iteratively to improve requirement coverage and contextual relevance. Scenario-specific instructions are incorporated via a --scenario flag. Output is saved as structured JSON (`data/transcripts/raw/synthetic/transcript_{scenario_name_or_no_scenario}_{uuid}.json`), including speaker turns and metadata. Measures like setting max_output_tokens=10000 were implemented to mitigate potential response truncation

Data Processing Pipeline

- **Transcript Parsing** (`src/utils/transcript_processing.py`): Raw generated transcripts (JSON format) are processed by this utility to standardize them into a consistent format (`data/transcripts/processed/parsed_transcript_{scenario_name}_{uuid}.json`), extracting the core conversation list for downstream use. It includes batch processing capabilities.
- **Requirement Extraction** (`src/agents/extractor.py`): An autonomous agent, built using the CrewAI framework and powered by an OpenAI model, processes parsed transcripts. Its defined task is to identify and extract key customer requirements (e.g., destination, duration, specific coverage needs) into a structured JSON format, validated against the TravelInsuranceRequirement Pydantic model. The agent operates in batch mode, processing multiple transcripts and saving outputs to `data/extracted_customer_requirements/requirements_{scenario_name}_{uuid}.json`.
- **Policy Extraction** (`scripts/extract_policy_tier.py`): This script leverages the multi-modal capabilities of the Gemini LLM (gemini-2.5-pro-exp-03-25) to extract detailed, structured information directly from travel insurance policy PDF documents. Refined Pydantic models (CoverageDetail, SourceDetail, ConditionalLimit) guide the extraction and validate the output JSON (`data/policies/processed/`). **The prompt instructs the LLM to consolidate benefits, differentiate**

base and conditional limits, and extract source snippets, enhancing the granularity and traceability of the extracted data. Robustness is improved through API retry logic.

Report Generation

- **Policy Comparison** (`scripts/generate_policy_comparison.py`): This script performs an insurer-level comparison. It takes extracted customer requirements (JSON) and multiple processed policy JSON files as input. For each insurer, it uses the Gemini LLM (gemini-2.5-pro-exp-03-25) with a detailed prompt to **select the single most suitable policy tier, provide justification, and conduct a requirement-by-requirement analysis comparing the customer's needs against the chosen tier's coverage, including source details.** Insurers are processed asynchronously for efficiency. Results are saved as Markdown reports (`results/{uuid}/policy_comparison_report_{insurer}_{uuid}.md`).
- **Recommendation** (`scripts/generate_recommendation_report.py`): A two-stage hybrid approach generates the final recommendation.
    - Stage 1 involves a scoring mechanism (based on comparison report analysis, implemented in Python) to rank policies.
    - Stage 2 employs the Gemini LLM to re-rank the top candidates, incorporating context from the original customer transcript to infer priorities and nuances not explicitly stated in the extracted requirements.

  The final output is a comprehensive Markdown report (`results/{uuid}/recommendation_report_{uuid}.md`) including the final recommendation, justification with source references, and the Stage 1 ranking. An `--overwrite` flag prevents redundant processing.

Evaluation Framework

A significant focus was placed on developing a robust evaluation framework to assess the quality and accuracy of the system's outputs, crucial for validating the research approach.

- **Transcript Evaluation** (`scripts/evaluation/transcript_evaluation/`): A modular system evaluates the quality and requirement coverage of generated transcripts using the Gemini LLM. It employs Pydantic schemas for structured JSON output and incorporates scenario-specific requirements into the evaluation

prompt, acting as an essential quality gate before transcripts proceed down the pipeline.

- **PDF Extraction Evaluation** (`scripts/evaluation/pdf_extraction_evaluation/eval_pdf_extraction.py`): This script utilizes a multi-modal LLM (Gemini) to perform a two-way verification between the extracted policy JSON and the source PDF document, assessing the fidelity of the extraction process. It supports filtering input files using a --file_pattern argument.
- **Comparison Report Evaluation** (`scripts/evaluation/comparison_report_evaluation/eval_comparison_report.py`): This component evaluates the generated comparison reports. It uses an LLM to compare the report's claims against the source policy documents, validating the accuracy of the comparison logic. Outputs are saved to `data/evaluation/comparison_report_evaluations/{uuid}/eval_comparison_{insurer}.json`.
- **Scenario Recommendation Evaluation** (`scripts/evaluation/scenario_evaluation/evaluate_scenario_recommendations.py`): The final policy recommendations for specific test scenarios are evaluated against predefined expected outcomes stored in `data/evaluation/scenario_evaluation/scenario_ground_truth.json`. This script can target specific run UUIDs (--target-uuid) for focused evaluation.
- **Ground Truth Coverage Evaluation** (`scripts/generate_ground_truth_coverage.py`): This script assesses how well a recommended policy covers the individual customer requirements identified during extraction. It uses an EmbeddingMatcher (`src/embedding/embedding_utils.py`) employing OpenAI embeddings and hybrid matching techniques to compare requirements against a curated knowledge base of policy coverage defined within `data/ground_truth/ground_truth.json`.

Core Services

- **LLM Service** (`src/models/llm_service.py, gemini_config.py`): A centralized service class was implemented to abstract interactions with the Google Gemini API. This promotes code reuse, centralizes configuration (default model, token limits, safety settings), and incorporates essential features like structured output generation (with Pydantic validation and parsing fixes), multi-modal input support, streaming, batch operations, and automatic retry logic, significantly improving development efficiency and robustness. For all tasks, a temperature of 0.1 was used

(`src/models/gemini_config.py, DEFAULT_PARAMETERS`) for all tasks and this project favours more deterministic and focused outputs.
- **OpenAI API Service**: Used implicitly through the CrewAI library for the Requirement Extraction agent.
- **Embedding Utilities** (src/embedding/embedding_utils.py): Contains the EmbeddingMatcher class, providing reusable functionality for semantic text comparison using embeddings, including a caching mechanism (`src/embedding/cache/`) to reduce redundant API calls and costs.

Orchestration & Demonstration

- **Orchestration Script** (`scripts/orchestrate_scenario_evaluation.py`): Automates the execution of the entire pipeline (data generation, processing, report generation, evaluation) for predefined test scenarios. Reliability was improved by adjusting parallel execution strategies (running transcript evaluations per file in parallel, but report generation sequentially per customer UUID) to manage LLM API load. An --only_aggregate flag allows running only the final evaluation aggregation step.
- **Demo Script** (`scripts/run_recsys_demo.py`): Provides a simplified way to execute the pipeline for a single, dynamically generated transcript and scenario, facilitating testing and demonstration of the core workflow. Debugging involved resolving subprocess argument mismatches, error handling, and ensuring correct parameter passing between chained script calls.

Frontend Application

A web-based frontend was developed using React, TypeScript, and Material UI to visualize the generated reports. Deployed on Netlify (aegis-recsys.netlify.app), it features:

- A landing page for initiating report viewing (using pre-generated UUIDs for demo purposes).
- A disclaimer page preceding report access.
- A tabbed report viewer displaying the Recommendation, Policy Comparison (with dynamic insurer selection and Table of Contents), extracted Customer Requirements (JSON view), and the original Transcript.
- Integration of feedback components (non-functional backend).
- The Netlify deployment required specific build configurations (`netlify.toml`, `package.json` scripts) including dependency installation within the correct subdirectory and an asset synchronization script (`scripts/sync-public-assets.cjs`) to copy necessary data files

(results/, etc.) into the build output. Dark mode compatibility was addressed using MUI theming.

Development Process & Iteration

The system's development was highly iterative, driven by experimentation and evaluation:

- **Prompt Engineering:** Significant effort was invested in crafting and refining prompts for LLM tasks. This involved iterative testing to improve the quality, accuracy, structure, and relevance of generated text, extracted data, and analytical comparisons. Designing specific prompts is akin to giving instruction to a young human toddler. Instructions are to be simple (e.g. reply true or false), not ambiguous (e.g. look for medical coverage, validate if pre-existing conditions are included, etc). LLM outputs were assessed using different temperature and top-P settings. A generally lower and more factual setting for this application, however, these key parameters cannot be close to minimal as results will become to conservative and repetitive to detect new insights or mistakes in the case of evaluation.
- **Evaluation-Driven Refinement:** The outputs from the various evaluation scripts provided critical feedback, directly informing modifications to upstream components. For instance, low scores in transcript evaluation led to prompt adjustments in the generation script, while inaccuracies identified by the PDF extraction evaluation prompted refinements in the policy extraction prompt and Pydantic models. Ground truth evaluation results highlighted the need to adjust the strictness of expected outcomes (`scenario_ground_truth.json`).
- **Debugging & Problem Solving:** Development involved addressing numerous challenges typical of complex, multi-component systems. These included handling LLM API rate limits (leading to adjustments in orchestration parallelism), parsing inconsistent or malformed LLM outputs (requiring enhanced parsing logic in the LLMService), resolving file path and dependency issues across scripts and during frontend deployment, and ensuring correct data flow and filename conventions between pipeline stages.
- **Modularity and Reusability:** Conscious efforts were made to build reusable components, such as the centralized LLMService and EmbeddingMatcher, to reduce code duplication and improve maintainability. Scripts were designed with command-line arguments to enhance flexibility and facilitate orchestration.

Findings & Discussion

Implementation Challenges

Several key challenges were encountered during implementation:

- **LLM Output Consistency:** Ensuring LLMs consistently produced output adhering to the desired structured format (JSON) required careful prompt engineering, the use of Pydantic for validation, and robust error handling for parsing.
- **API Limitations:** Managing costs and rate limits of commercial LLM APIs necessitated strategies like batch processing, asynchronous calls where appropriate, and adjustments to parallel execution in orchestration scripts.
- **Evaluation Complexity:** Designing effective evaluation methods, particularly for complex outputs like comparison reports and multi-modal PDF extraction verification, required significant effort and experimentation. Defining appropriate ground truth data also proved challenging.
- **System Integration:** Ensuring seamless data flow and compatibility between the numerous interconnected scripts and components required careful management of file paths, data formats, and dependencies. Debugging issues spanning multiple pipeline stages was complex.
- **Frontend Deployment:** Integrating backend data (results, transcripts) with the frontend application and configuring the Netlify build process involved resolving path issues, dependency conflicts, and ensuring necessary assets were correctly bundled.

Structured Prompting + Advancements in LLMs for Knowledge Extraction

A cornerstone of this project relies on the automated extraction of structured information from complex, often lengthy travel insurance policy documents in PDF format. Historically, this task presented significant challenges for traditional Optical Character Recognition (OCR) and rule-based Natural Language Processing (NLP) techniques, which struggle with varied layouts, dense insurance jargon, nested clauses, and the implicit relationships between different coverage sections. The fidelity of this initial extraction step is paramount, as any inaccuracies or omissions directly propagate downstream, undermining the reliability of subsequent policy comparisons and final recommendations. Therefore, achieving high-quality, automated extraction from these unstructured documents was identified as a critical prerequisite for the system's success.

The advent of advanced Large Language Models (LLMs) with multi-modal

capabilities represents a paradigm shift in addressing such challenges. This project leveraged Google's Gemini 2.5 Pro model (gemini-2.5-pro-exp-03-25) for the policy extraction task (`scripts/extract_policy_tier.py`), capitalizing on its proficiency in interpreting not only the text but also the layout and structure inherent in PDF documents. Preliminary evaluations and qualitative analysis revealed **Gemini 2.5 Pro's state-of-the-art ability to parse complex tables, identify distinct coverage benefits even when described across multiple paragraphs, extract granular details like conditional limits and their associated source text**, and structure this information accurately according to a predefined Pydantic schema.

Notably, comparative tests conducted during development demonstrated a marked improvement in extraction quality and completeness when using Gemini 2.5 Pro compared to its predecessor, Gemini 1.5 Pro. This improvement was particularly evident when processing complex PDF documents that contain nested tables and scattered coverage information (across multiple tables, appendices at the end, etc). It also exhibited a superior ability to consolidate information pertaining to a single coverage benefit even when that information was dispersed across multiple pages or sections within the PDF. It could effectively synthesize descriptions, limits, and conditions mentioned in different locations (e.g., Page 6, 9 & 28) into a coherent structured representation for a specific coverage like "Car rental excess".

However, unlocking the full potential of Gemini 2.5 Pro for this specific task required more than just leveraging its inherent capabilities. Recognizing the need for consistent, highly structured, and verifiable output tailored to our downstream recommendation logic, we complemented the model's power with a meticulously crafted, instructive prompt (`scripts/extract_policy_tier.py`,

PROMPT_TEMPLATE found on line 144 onwards).

The design philosophy behind this prompt was to **guide the model's advanced reasoning capabilities explicitly towards the specific challenges of insurance policy extraction**. Standard prompts might yield factually correct but poorly structured or incomplete data for our purposes. Our prompt template, therefore, incorporates several key elements designed to elicit high-fidelity, actionable information:

- **Strict Schema Enforcement**: It defines a precise, nested JSON output structure, ensuring consistency and facilitating automated downstream processing. This goes beyond simple field extraction, dictating relationships between data points (e.g., nesting limits under conditions).
- **Benefit Consolidation Mandate**: Insurance documents often describe aspects of a single benefit (e.g., "Car Rental Excess") in multiple locations (main benefits table, general exclusions, specific conditions section). The prompt explicitly instructs the model to synthesize this scattered information into a single, consolidated coverage object. This requires the model to identify conceptual relationships across disparate text sections.
- **Distinction between Base and Conditional Limits**: The prompt forces the model to differentiate between standard base_limits and conditional_limits that apply only under specific circumstances (like purchasing an add-on). This level of granularity is crucial for determining strict suitability.
- **Emphasis on Verifiable Source Details** (source_specific_details): Acknowledging the need for traceability and trust, the prompt demands more than just page numbers. It mandates the extraction of detail_snippets – summarized key conditions, inclusions, or exclusions directly from the source text – alongside their source_location. Crucially, it forbids generic placeholders like "Refer to Section X," compelling the model to demonstrate its understanding by summarizing the relevant content. This provides direct textual evidence supporting the extracted structured data.
- **Explicit Handling of Common Patterns and Negatives**: The prompt includes instructions for interpreting common table symbols (✓, X) and provides clear positive and negative examples, guiding the model away from frequent pitfalls like incorrect structuring or superficial referencing.

This engineered synergy between the advanced multi-modal understanding of Gemini 2.5 Pro and the detailed guidance provided by our structured prompt results in a superior extraction capability. This distinction is not merely academic; it has profound implications for the effectiveness of the downstream recommendation system. **The nuanced details uniquely captured through this process – such as specific conditions for coverage activation, benefit sub-limits under particular scenarios, or limitations tied to optional add-ons**

– are precisely the factors that determine strict suitability for a user's essential requirements. When these details are omitted, as was often the case with less sophisticated models or simpler prompts, the recommendation logic operates on incomplete information. This can lead to suggesting policies or tiers that appear suitable based on headline limits but fundamentally fail to meet essential coverage needs when granular conditions are considered. Conversely, a truly suitable option might be overlooked because a crucial nuance was missed during extraction.

Consequently, the superior extraction fidelity achieved enables more refined and accurate reasoning within the recommendation engine. It allows the system to move beyond simple limit comparisons and truly evaluate if a policy's specific terms and conditions align with the essential, non-negotiable requirements. This ultimately leads to recommendations that are genuinely suitable and trustworthy, fulfilling the project's core objective. This comparison underscores the significant advancements in machine reasoning embodied by Gemini 2.5 Pro, particularly when coupled with targeted prompt engineering. Its enhanced proficiency in navigating intricate documents, understanding contextual dependencies, and extracting fine-grained details directly addressed the core challenge of achieving high-fidelity automated knowledge extraction, thereby laying a robust foundation for the subsequent analytical and recommendation stages of this project.

Quantitative Evaluation

Evaluation Methodology and Rationale

To objectively assess the performance and reliability of the implemented recommendation system, a quantitative evaluation framework was established. This framework focuses on two key aspects: the correctness of the final policy recommendation for specific, challenging scenarios, and the granular coverage of individual customer requirements by the recommended policy.

Crucially, these evaluations are designed not as a one-time assessment but as an integral part of the ongoing development and refinement process. The automated evaluation scripts (`scripts/evaluation/scenario_evaluation/evaluate_scenario_recommendations.py` and `scripts/generate_ground_truth_coverage.py`) serve multiple purposes:

- **Regression Detection:** By re-running evaluations after code changes (e.g., prompt updates, logic modifications, model upgrades) and/or changes to

LLM API endpoints, regressions that negatively impact performance can be quickly identified.

- **Faithfulness and Correctness Verification:** Evaluations against predefined ground truth provide a measure of whether the system behaves as expected and adheres to desired outcomes for known inputs.
- **Objective Comparison and Benchmarking:** The quantitative metrics allow for objective comparison between different system configurations, facilitating data-driven decisions when experimenting with alternative LLMs, prompts, or recommendation logic.

Scenario-Based Recommendation Evaluation

This evaluation assesses the system's ability to recommend the correct final policy (or one of a set of acceptable policies) for predefined test scenarios, each designed to probe specific coverage nuances or complex requirement combinations. The system was tested against four distinct scenarios (golf_coverage, pet_care_coverage, public_transport_double_cover, uncovered_cancellation_reason), each with 20 generated customer cases. Performance was measured by comparing the system's final recommendation against the expected outcome defined in the scenario ground truth (`data/evaluation/scenario_evaluation/scenario_ground_truth.json`).

The results, summarized in Table 5.1 (derived from data/evaluation/scenario_evaluation/scenario_evaluation_results.md), indicate strong performance in most scenarios. The system achieved perfect accuracy (100% pass rate) for the golf_coverage and uncovered_cancellation_reason scenarios. High accuracy was observed for pet_care_coverage (95% pass rate, 1 failure out of 20). The public_transport_double_cover scenario proved most challenging, with an 85% pass rate (3 failures out of 20). Further analysis of failures (detailed in the results file) can pinpoint specific weaknesses in the recommendation logic or upstream processing for these edge cases.

Table 5.1: Scenario Recommendation Evaluation Results

Scenario	Pass Rate	Cases Passed	Total Cases
golf_coverage	100.00%	20	20
pet_care_coverage	95.00%	19	20
public_transport_double_cover	85.00%	17	20
uncovered_cancellation_reason	100.00%	20	20

Ground Truth Coverage Evaluation

Beyond evaluating the final recommendation correctness, this evaluation assesses how well the recommended policy covers the *individual* requirements extracted from the customer transcript. This provides a more granular measure of the recommendation's quality and alignment with specific user needs. The evaluation uses an EmbeddingMatcher to compare extracted requirements against a curated ground truth knowledge base (`data/ground_truth/ground_truth.json`) defining which requirements are covered by which policy tiers.

Two primary metrics were calculated across the 80 test cases (20 per scenario):

- **Total Coverage Rate:** Percentage of all extracted requirements covered by the recommended policies.
- **Valid Coverage Rate:** Percentage of *valid* requirements covered, excluding requirements identified as not existing in any available policy (e.g., highly niche requests). This is considered the primary metric.

The overall results (summarized from `data/evaluation/ground_truth_evaluation/coverage_evaluation_summary.md`) show a **Valid Coverage Rate of 87.53%** across all 481 valid requirements identified in the 80 customer cases. The Total Coverage Rate (including non-existent requirements) was 80.19%.

Further analysis reveals the distribution of coverage quality (Table 5.2). While 30% of customers (24 out of 80) received recommendations covering 100% of their valid requirements, the majority (65% or 52 customers) received recommendations covering 80-89% of their valid needs. A small number fell into lower coverage bands.

Table 5.2: Distribution of Valid Requirement Coverage

Valid Coverage Band	Number of Customers	Percentage
100%	24	30.0%
90-99%	0	0.0%
80-89%	52	65.0%
70-79%	3	3.75%
60-69%	1	1.25%
<60%	0	0.0%

Analysis of the specific uncovered requirements for customers with less than 100% valid coverage (detailed in `coverage_evaluation_summary.md`) frequently points towards nuances in "damaged luggage" or specific exclusions like "adventurous activities" or "pre-existing conditions" not being met by the recommended policy, even if it was the best overall fit according to the scenario ground truth. **This highlights the trade-offs involved in recommendation and provides specific areas for future improvement in the requirement weighting or policy comparison logic.**

Summary of Quantitative Findings

The quantitative evaluations demonstrate a generally high level of performance for the implemented system. Scenario-based testing shows strong accuracy in recommending appropriate policies for complex situations, while the coverage evaluation indicates that the recommended policies generally address a high percentage (87.53%) of valid customer requirements. The framework established provides essential tools for ongoing monitoring, regression testing, and objective benchmarking as the system evolves. The detailed results also pinpoint specific areas, such as handling nuanced requirements like luggage damage variations or specific activity exclusions, where further refinement could enhance recommendation quality.

Qualitative Feedback / Learning Outcomes

Creation of PDF extraction & Policy Comparison Report Evaluation

- **Adapting Evaluation Scope**: Initial attempts to validate against a fixed set of predefined coverage types proved insufficient. The evaluation framework needed redesigning to dynamically handle the specific, varied requirements present in each unique customer profile.
- **Leveraging LLMs for Fact-Checking**: Utilizing an LLM to directly compare analysis made in the processed policy or generated comparison report against the source PDF policy documents was a core validation strategy.
- **Structured Output Facilitates Flexibility**: Structured JSON output formats (for requirement-level and summary-level checks) to guide the LLM's analysis. Defining clear JSON schemas for the LLM's validation output was essential. This allowed the evaluation to adapt to varying numbers of requirements per customer and provided machine-readable results for analysis.

- **Nuanced LLM Judgement**: The LLM validator demonstrated an ability to identify subtle inaccuracies, such as partially supported justifications (e.g., coverage requiring an add-on not mentioned in the report), leading to more stringent and realistic validation outcomes.
- **Multi-Step Validation**: Employing a two-task validation process (first checking individual requirements, then validating summary statements) provided a more robust assessment of the comparison report's overall accuracy and faithfulness to the source documents.

Creation of Ground Truth Coverage Evaluation

- **Partial Coverage of Multi-Part Requirements**: User requirements often combined multiple related needs (e.g., "lost or damaged luggage"). The evaluation sometimes flagged these as uncovered if the recommended policy addressed only one part (e.g., loss but not damage), highlighting a challenge in evaluating partial fulfillment.
- **Specificity Mismatch (User vs. Policy)**: Users frequently expressed highly specific needs (e.g., "reimbursement for unused green fees"), while policies offered broader coverage categories (e.g., "golf coverage"). This mismatch led to requirements being marked as uncovered by the automated evaluation, despite potential semantic overlap.
- **Semantic Similarity Limitations for Niche Activities**: Standard embedding-based semantic similarity struggled to reliably match specific user-mentioned activities (e.g., "hot air ballooning") to general policy categories (e.g., "Adventurous activities"). A heuristic workaround (checking for population of a dedicated policy field) was implemented to improve matching for these cases.
- **Handling Non-Existent Coverage**: Requirements for coverage types not offered by any available policy (e.g., "space travel incidents") were identified and explicitly excluded from the primary coverage metrics to avoid unfairly penalizing recommendations.
- **Time Constraints and Edge Case Prioritization**: Project timelines necessitated prioritizing the mapping of common and impactful coverage details. Exhaustive mapping of every conceivable sub-coverage or niche edge case was not feasible within the scope of this iteration, representing a known limitation.

Why Hybrid Recommendation Logic?

- **Hybrid Approach Value:** A purely rule-based scoring system was deemed insufficient for capturing nuanced trade-offs, while relying solely on LLMs for comparing all options was inefficient. A hybrid approach, using quantitative scoring (Stage 1) to filter candidates followed by qualitative LLM re-ranking (Stage 2), emerged as a practical solution balancing efficiency and decision quality.
- **Limitations of Simple Scoring:** Implementing the Stage 1 quantitative score highlighted its inherent limitations. While efficient, scoring based solely on counts of "Fully Met" / "Partially Met" requirements ignores the relative importance of different needs and the qualitative insights present in the comparison report text.
- **Dependency on Upstream Consistency:** The success of the Stage 1 scoring parser was highly dependent on the consistency and strict adherence to the defined Markdown format by the upstream LLM generating the comparison reports. This underscored the critical importance of precise prompt engineering and output structure control for reliable pipeline processing.
- **LLM Role in Nuanced Decision-Making:** The Stage 2 LLM re-ranking was essential for incorporating holistic understanding, comparing qualitative strengths/weaknesses, and considering subtle context (like transcript cues) that the quantitative score missed, reaffirming the value of LLMs for complex reasoning tasks.
- **Value of Transcript Context:** Integrating the original customer transcript into the Stage 2 re-ranking process was identified as a key enhancement, enabling the LLM to potentially infer requirement priorities and generate more personalized, contextually relevant justifications.
- **Integrated Evaluation is Key:** Building the scenario-based ground truth evaluation alongside the recommender logic proved essential. It provided immediate feedback for regression testing, correctness verification, and objective benchmarking during development and iteration.
- **Ground Truth Definition Challenges:** Defining the ground truth itself, including determining acceptable policies and justifications for complex scenarios, required careful consideration and domain understanding, representing a significant effort within the evaluation process.

Enhancing Report Reliability through Consensus Mechanisms

A critical challenge in leveraging LLMs for generating analytical outputs, such as the policy comparison reports central to this project, lies in managing their inherent non-determinism. Initial runs of our policy comparison script (`scripts/generate_policy_comparison.py`), which compares customer requirements against individual insurance policy tiers using Google's Gemini 2.5 Pro, revealed minor inconsistencies between generations. Variations in extracted details or assigned "Match Certainty" scores, while often subtle, posed a risk: **inaccuracies in these intermediate reports could propagate downstream, potentially compromising the final policy recommendation, irrespective of the sophistication of the final analysis stage.** This highlighted the need to enhance the reliability of the comparison report generation process itself. Inspired by techniques designed to improve reasoning reliability in LLMs, such as self-consistency[23], we initially explored a multi-stage consensus-building approach.

The core idea was analogous to generating multiple reasoning paths and selecting the most consistent outcome. In our context, this involved:

1. Candidate Generation: For each policy tier, generate multiple (e.g., M=10) interim comparison reports using a faster, potentially less powerful LLM (e.g., GPT-4o Mini).
2. Synthesis & Verification: Feed these M interim reports, along with the original structured policy data (extracted via `scripts/extract_policy_tier.py`), into a powerful LLM (e.g., Gemini 2.5 Pro). This LLM would be tasked with analyzing consistency across the candidates, verifying claims against the source policy JSON, synthesizing the findings, and producing a single, consolidated, high-fidelity comparison report for that specific tier.

This approach was theoretically appealing as it aimed to "average out" stochastic variations or minor errors from the initial generation stage, thereby increasing the robustness and trustworthiness of the reports fed into the subsequent insurer-level tier selection and final recommendation steps (`scripts/generate_recommendation_report.py`).

However, detailed planning revealed significant practical challenges associated with this multi-stage consensus approach, particularly given the project's operational constraints (including a sub-one-month timeline for specific deliverables):

- **Workflow Complexity**: Orchestrating the generation, storage, and processing of numerous interim reports (e.g., 90 reports for 9 tiers in the example scenario) before feeding them into consolidation, tier-selection, and final recommendation stages introduced considerable system complexity.
- **Latency and Cost**: The cumulative LLM inference time and associated API costs were projected to increase substantially compared to the single-pass approach (estimated 90 cheap + 13 powerful calls vs. ~10 powerful calls).
- **Implementation Effort**: Developing and rigorously testing the prompts for the crucial synthesis stage, alongside building the necessary orchestration logic, represented a significant engineering effort with inherent risks within the available time frame.

Faced with these pragmatic constraints, a decision was made to adopt an enhanced version of the existing single-pass architecture. This involved:

1. **Prompt Refinement**: Investing effort in optimizing the prompt used by the powerful LLM (Gemini 2.5 Pro via LLMService in `scripts/generate_policy_comparison.py`) to maximize its accuracy and consistency in a single generation pass.
2. **Basic Validation**: Implementing lightweight automated checks on the generated Markdown reports to verify structural integrity and the presence of key fields (e.g., "Match Certainty", "Justification") as a basic sanity check, while acknowledging these tests cannot fully validate semantic correctness.
3. **Downstream Robustness**: Relying on the subsequent stages, particularly the final recommendation script (`scripts/generate_recommendation_report.py`) which incorporates scoring logic and an LLM-based re-ranking step using customer transcript context, to implicitly handle minor variations present in the input comparison reports.

This exploration underscored a crucial learning point in applying advanced AI techniques: **the necessary balance between theoretical optimality and practical feasibility**. While the multi-stage consensus approach, drawing parallels with self-consistency, offered a potentially more robust solution to LLM variability, its implementation overhead was prohibitive under the project's constraints. The adopted pragmatic approach, focusing on optimizing the existing pipeline and adding basic safeguards, allowed for the timely delivery of a functional end-to-end system (`scripts/orchestrate_scenario_evaluation.py`, `scripts/run_recsys_demo.py`). This gave the team insights to the importance of iterative development and accepting incremental improvements in complex AI systems. The multi-stage consensus mechanism remains a valuable concept noted for potential future iterations where development resources and timelines may

permit its implementation.

Limitations of Whole-Document Embeddings for Granular Policy Matching

In pursuit of optimizing the recommendation workflow, particularly to manage the computational load associated with detailed LLM-based comparisons, an alternative pre-filtering strategy using content-based filtering via document embeddings was explored. The hypothesis was that representing the entire structured customer requirement JSON (e.g., `requirements_{scenario}_{uuid}.json`) and each processed policy JSON (`data/policies/processed/{insurer}_{tier}.json`) as single high-dimensional vectors could allow for rapid initial filtering.

The proposed method involved:

1. Embedding Generation: Creating a single vector embedding for each customer requirement file and each policy file.
2. Similarity Calculation: Computing the cosine similarity between the customer requirement vector and each policy vector.
3. Pre-filtering: Selecting a small subset (e.g., top 5) of policies with the highest similarity scores to pass onto the more resource-intensive comparison and recommendation stages executed by scripts like `scripts/generate_policy_comparison.py` and `scripts/generate_recommendation_report.py`.

This approach initially seemed promising due to its potential for speed and simplicity. By performing a single vector comparison per policy, it offered a computationally inexpensive way to narrow down a large pool of potential insurance options, theoretically improving system responsiveness and scalability.

However, experimentation revealed critical limitations that rendered this whole-document embedding strategy unsuitable for the project's core requirements of providing accurate, granular, and explainable policy recommendations:

- **Loss of Granularity**: The primary drawback was the significant loss of detail inherent in compressing complex, structured documents into single vectors. Specific, often crucial, customer requirements (e.g., coverage for "golf equipment rental" or "pet accommodation during travel delay"), meticulously extracted by the Extractor Agent (src/agents/extractor.py), had negligible influence on the overall document embedding. Consequently, a policy lacking such specific coverages could still yield a high similarity score

if it broadly matched on general travel insurance concepts, failing the objective of “suitable recommendation” over “relevant recommendation”.

- **Poor Explainability**: A single cosine similarity score provides no insight into *why* a policy is considered similar or *which specific requirements* it satisfies. This lack of transparency directly conflicts with the project's goal of generating justified recommendations, as seen in the detailed Markdown reports produced by the final stages. It would be impossible to explain to a user why certain policies were filtered out or why the recommended policy was chosen based solely on an opaque similarity score.
- **Vulnerability to Noise and False Positives**: The embeddings were susceptible to being skewed by generic terms ("travel," "medical," "accident") or irrelevant administrative details present in the policy JSON files. This could lead to false positives, where policies rank highly due to superficial term overlap rather than genuine alignment with the user's specific needs.

Given these findings, it was concluded that while whole-document embedding can be effective for high-level document similarity tasks, it lacks the necessary precision and transparency for the nuanced task of matching detailed customer insurance requirements against specific policy provisions. The risk of filtering out suitable policies or recommending unsuitable ones due to the loss of granularity was too high.

This outcome reinforced the necessity of the current approach, which involves detailed, albeit more computationally intensive, analysis by LLMs capable of examining the structured data point-by-point (`scripts/generate_policy_comparison.py`). It also serves to distinguish this application from the project's *successful* use of embeddings (`src/embedding/embedding_utils.py`), which focuses on comparing *individual requirement statements* against a curated ground truth knowledge base for evaluation purposes – a task where semantic similarity at a finer grain is appropriate and effective.

Limitations & Risks

- **Pipeline Error Propagation and Validation Effectiveness**: Errors in early stages can impact downstream tasks, and evaluation effectiveness has limits.
    - Implemented Mitigation(s):
        - Multiple evaluation scripts exist - providing checks at various pipeline stages.
        - Transcript evaluation (`eval_transcript_main.py`) acts as an explicit quality gate before extraction.
        - PDF extraction evaluation (`eval_pdf_extraction.py`) uses multi-modal LLM comparison against the source document.
        - Semantic evaluation against ground truth (`evaluate_scenario_recommendations.py`, `generate_ground_truth_coverage.py` using EmbeddingMatcher) provides checks beyond simple keyword matching for final recommendations and requirement coverage.
        - Structured data validation via Pydantic models helps catch formatting errors early.
- **LLM Performance, Consistency, and Bias**: The system's core functions rely heavily on LLMs (Gemini, OpenAI), introducing risks of non-determinism, potential inaccuracies, and sensitivity to model versions or biases.
    - Implemented Mitigation(s):
        - Centralized LLMService (src/models/llm_service.py) standardizes Gemini API calls, incorporating retry logic and robust JSON parsing to handle minor formatting inconsistencies.
        - Pydantic models are used extensively (`src/utils/transcript_processing.py`, `scripts/extract_policy_tier.py`, evaluation scripts) to enforce structured outputs from LLMs, acting as a validation layer.
        - Prompt engineering efforts aim to guide LLMs towards more consistent and accurate outputs.
        - The decision to use specific, powerful models (e.g., gemini-2.5-pro-exp-03-25) for critical tasks like policy extraction and comparison aims to maximize single-pass accuracy.

- **Input Data Constraints**:
    - **Reliance on Synthetic Data**: Primary testing used synthetic transcripts, which may not fully capture real-world complexities.
        - Implemented Mitigation(s): Scenario-based transcript generation (`scripts/data_generation/generate_transcripts.py` with data/scenarios/) attempts to add more realistic constraints and coverage requirements to the synthetic data.
    - **Limited Policy Corpus**: Development utilized a restricted set of policy documents.
        - Implemented Mitigation(s): None currently implemented within the system itself to address the breadth of the corpus. This remains a limitation addressed only by future data acquisition.
- **Ground Truth Data Maintenance:** Semantic evaluation accuracy depends on the curated ground truth.
    - **Implemented Mitigation(s):** The ground truth data (`data/ground_truth/ground_truth.json`) is centralized and explicitly version-controlled within the project repository, facilitating easier updates and tracking compared to scattered files.
- **Absence of Pricing Information:** Policy documents lack pricing, limiting actionable recommendations.
    - **Implemented Mitigation(s):** None implemented. The system focuses on feature comparison as pricing data is unavailable in the source documents.

Future Enhancements

Building upon the current system's foundation and addressing its identified limitations, several avenues for future enhancement are proposed to increase its robustness, utility, commercial viability, and scalability:

- **Transition to Real-Time Conversational Interaction**: Evolve the system from its current batch-processing architecture to a real-time, interactive platform. This would involve developing conversational agents capable of engaging users directly, eliciting requirements dynamically, and providing immediate policy recommendations, significantly improving user experience and responsiveness.
- **Expand Insurance Product Scope**: Extend the system's capabilities beyond single-trip travel insurance to encompass a wider range of insurance products. This could include variations like annual multi-trip travel policies, as well as entirely different domains such as Life Insurance, Critical Illness (CI), or Total Permanent Disability (TPD) coverage, leveraging the core document analysis and comparison engine.
- **Integration with External Services and Data Sources**:
    - Pricing and Provider APIs: Address the critical limitation of absent cost data by integrating external pricing APIs or establishing direct data feeds from insurers. This would enable cost-sensitive recommendations and potentially facilitate direct policy applications.
    - Identity/Data Verification Services: Integrate with services like Singpass (in the relevant regional context) to streamline user data collection, potentially pre-filling demographic information or verifying identity securely.
- **Advanced Mechanisms for Report Reliability**: Implement more sophisticated techniques to further mitigate LLM non-determinism. This could involve revisiting multi-stage consensus approaches (generation-synthesis-verification) for critical outputs or developing more advanced automated validation methods capable of assessing semantic correctness.
- **Enhanced Evaluation Framework with Human-in-the-Loop (HITL)**: Augment the current automated and LLM-based evaluation framework (scripts/evaluation/) by incorporating human review. HITL validation is crucial for assessing nuanced outputs, verifying complex reasoning, handling edge cases missed by automated checks, and building greater confidence in the system's accuracy, particularly during refinement stages.
- **Integration and Analysis of Real User Data**: Post-pilot deployment, incorporating data derived from actual user interactions. Analyzing real

conversational data and user feedback (collected via the ux-webapp feedback components) can refine requirement extraction models, improve synthetic data generation, potentially enable LLM fine-tuning, and guide user experience improvements.

- **Knowledge Discovery and Predictive Analytics via Machine Learning**: Implement and expand upon the planned supervised machine learning models (`notebooks/supervised_learning/`). Leveraging accumulated user requirement data (`data/extracted_customer_requirements/`), policy choices, and potentially feedback/claims data could enable advanced use cases such as:
    - Predicting customer churn or lifetime value.
    - Identifying underserved market segments or emerging coverage needs.
    - Optimizing recommendation strategies based on user profiles.
    - Potentially informing dynamic pricing models (if pricing data is integrated).
    - Personalizing conversational agent interactions.
- **Scalability, Performance, and Cost Optimization**: Systematically investigate and implement strategies to enhance system throughput and reduce operational costs. This includes exploring advanced caching, model distillation/quantization, API usage optimization, asynchronous processing improvements, or architectural redesigns to handle exponential growth in users and policies efficiently.
- **Value-Added Services and Monetization**: Explore premium features built upon the core policy understanding capabilities. For instance, offering subscribers real-time, in-app checks against their active policy documents to verify coverage for specific scenarios (e.g., "Is this hospital covered for emergency treatment under my current plan?") could provide significant post-purchase value.
- **Continuous Improvement of Explainability**: Further enhance the transparency of recommendations by refining justification generation and potentially exploring visualization techniques to clearly illustrate how user requirements map to specific policy clauses and benefits.

Conclusion

This NUS ISS MTech project provided valuable insights into the practical application and evaluation of **Intelligent Reasoning Systems**, specifically LLM-based approaches, for complex real-world tasks such as insurance policy analysis.

Key challenges underscored the nuances of deploying such systems:

- managing LLM output consistency, which necessitated careful prompt engineering and iterative validation loops for **reliable reasoning**;
- designing robust multi-faceted **evaluation frameworks suitable for AI systems**, acknowledging their non-deterministic nature and requiring checks beyond simple accuracy metrics; and
- integrating multiple AI and script-based components into a coherent pipeline, managing data dependencies and potential error propagation.

The successful implementation of **multi-modal knowledge extraction** using Gemini 2.5 Pro was particularly noteworthy, underscoring the rapid advancements by enabling direct processing of industry-standard PDF documents and bypassing brittle pre-processing steps. Furthermore, the development process highlighted the necessity of a **hybrid reasoning** approach; relying solely on quantitative scoring proved insufficient for capturing contextual nuances, while a purely LLM-based ranking was less efficient and potentially overlooked explicit requirement mismatches. This led to the adopted two-stage model (quantitative filtering + qualitative LLM re-ranking). Experimentation also revealed the limitations of techniques like whole-document embeddings for granular policy matching, reinforcing the need for feature-level analysis in this domain.

Ultimately, Aegis AI serves as a practical demonstration of harnessing sophisticated **AI reasoning techniques** – knowledge extraction, structured reasoning, and semantic evaluation – to address tangible consumer pain points, effectively bridging the gap between theoretical AI potential and impactful application within the constraints and timeline of a Master's project.

References

1. What Is Travel Insurance, and What Does It Cover? *Investopedia* https://www.investopedia.com/terms/t/travel-insurance.asp.
2. What does travel insurance cover? - It's Your Yale. https://your.yale.edu/sites/default/files/files/Travel%20Insurance%20101.pdf.
3. Your Guide to Travel Insurance, gia.org.sg. https://gia.org.sg/images/resources/For-Consumers-PDF-file/GIA_TravelInsurance_Guide.pdf.
4. Trends in travel insurance as the world takes flight. https://anziif.com/professional-development/the-journal/volume-46/issue-4/trends-in-travel-insurance-as-the-world-takes-flight.
5. Singapore Consumers Travel Insurance Needs. https://ancileo.com/singapore-consumers-travel-insurance-needs/.
6. What's the best travel insurance? https://www.reddit.com/r/askSingapore/comments/1ih53zs/whats_the_best_travel_insurance/.
7. Klook survey exposes travel insurance gap in Asia. https://www.ttgasia.com/2024/09/04/klook-survey-exposes-travel-insurance-gap-in-asia/.
8. Singapore Travel Insurance Market Size and Statistics - 2030. https://www.nextmsc.com/report/singapore-travel-insurance-market.
9. Southeast Asia Travel Insurance Market Size, Share, Competitive Landscape and Trend Analysis Report, by Insurance Cover, Distribution Channel, End User, Mode : Opportunity Analysis and Industry Forecast, 2024-2032. *Allied Market*

Research

https://www.alliedmarketresearch.com/southeast-asia-travel-insurance-market-A324610.

10. Trends in travel insurance as the world takes flight.

https://anziif.com/professional-development/the-journal/volume-46/issue-4/trends-in-travel-insurance-as-the-world-takes-flight.

11. Southeast Asia travel insurance Market to Hit $1,390.32 Mn by 2032.

https://www.alliedmarketresearch.com/press-release/southeast-asia-travel-insurance-market.html.

12. Travel Insurance Market Size, Share | CAGR of 17.5%.

https://market.us/report/global-travel-insurance-market/.

13. Asia-Pacific Travel Insurance Market Size and Statistics - 2030.

https://www.nextmsc.com/report/asia-pacific-travel-insurance-market.

14. Asia-Pacific Travel Insurance Market. *MarketResearch.biz*

https://marketresearch.biz/report/asia-pacific-travel-insurance-market/.

15. 10 Best Business Travel Insurance In APAC.

https://trutrip.co/blog/business-travel-insurance-in-apac/.

16. Singapore Consumers Travel Insurance Needs.

https://ancileo.com/singapore-consumers-travel-insurance-needs/.

17. Insubuy Insurance Statistics 2024.

https://www.insubuy.com/travel-insurance-statistics/.

18. Singapore Travel Insurance Market Size and Statistics - 2030.

https://www.nextmsc.com/report/singapore-travel-insurance-market.

19. Beware when getting travel insurance - some are now offering

non-comprehensive plans. https://www.reddit.com/r/singapore/comments/1et8mfm/beware_when_getting_travel_insurance_some_are_now/.

20. Compare Best Travel Insurance Policies Singapore - SingSaver. https://www.singsaver.com.sg/travel-insurance/comparison.

21. Best MSIG Travel Insurance in Singapore. *MoneySmart.sg* https://www.moneysmart.sg/travel-insurance/msig.

22. Dhanashree, B. How AI is Transforming Travel Insurance Quoting. https://www.alltius.ai/glossary/how-ai-is-transforming-travel-insurance-quoting.

23. Wang, X. *et al.* Self-consistency improves chain of thought reasoning in language models. *arXiv preprint arXiv:2203.11171* (2022).

Appendix

Project Proposal

Project proposal presentation slides are available at this link.

Mapping of System Functions to Lesson Materials

Machine Reasoning

- **Explainable AI (XAI):** The project emphasizes transparency through the generation of comparison reports (`scripts/generate_policy_comparison.py`) and recommendation rationales (`scripts/generate_recommendation_report.py`). LLM prompts are specifically engineered to provide justifications for comparisons and final recommendations. Furthermore, mitigating LLM hallucination risks via multi-layered validation (transcript eval, PDF extraction eval, ground truth checks) and a structured scoring/re-ranking process enhances trustworthiness, reflecting good practices in XAI. The caveat is that the quality of explanation is inherently tied to the LLM's capabilities (Gemini 2.5 Pro, in our case) and the effectiveness of prompt engineering.
- **Categorization & Structuring:** A core function involves converting unstructured conversation logs into actionable, structured requirement sets (`src/agents/extractor.py` using the TravelInsuranceRequirement Pydantic model). This structuring and categorization of raw text into a defined schema is crucial for enabling systematic downstream matching and analysis.
- **Hybrid Reasoning (Rule-Based + Statistical):** This project implements a two-stage hybrid reasoning system within `scripts/generate_recommendation_report.py`. Stage 1 uses rule-based scoring (explicit criteria for policy matching) for initial filtering and ranking. Stage 2 employs an LLM-driven re-ranking. This deep learning model, utilizing attention mechanisms for long-range contextual understanding, performs a nuanced re-ranking based on its semantic interpretation of the customer transcript and the comparison reports. This hybrid approach leverages the strengths of both symbolic and sub-symbolic AI reasoning methods.

- **Ethics, Privacy, and Trustworthiness:** Trustworthiness is addressed by mitigating LLM hallucination risks through multiple validation layers (transcript, PDF extraction, scenario outcome, ground truth coverage evaluations). The use of structured data formats (Pydantic models) ensures consistency and reduces ambiguity. Ethical considerations are reflected in the inclusion of a disclaimer page (`ux-webapp/src/pages/DisclaimerPage.tsx`) and the focus on transparency in recommendations, echoing the importance of responsible AI.

Reasoning Systems

- **Content-Based Filtering:** The system fundamentally operates on content-based filtering principles. It matches user requirements (analogous to a user profile), extracted from conversational transcripts (`src/agents/extractor.py`), to policy features (analogous to item content/attributes), extracted from PDF documents (`scripts/extract_policy_tier.py`).
- **Feature Extraction:** The project heavily relies on extracting structured information, mirroring the "Content Analyser" concept. (scripts/extract_policy_tier.py) uses the advanced multi-modal capabilities of Gemini 2.5 Pro to parse complex PDF policy documents and extract detailed, structured features (coverage details, limits, conditions, source text) into JSON format. This is a sophisticated form of extracting item attributes. (src/agents/extractor.py) uses an AI agent (CrewAI with OpenAI) to process customer transcripts and extract structured requirements (destination, duration, specific needs) into a validated requirement in JSON format. This builds the "user profile."
- **Data Preparation and Feature Engineering**: The lessons stresses the importance of data cleaning and transformation before ML modeling. Policy PDFs and user conversations are transformed into structured JSON formatting. Validation of format is verified using Pydantic methods. Documents are normalized and cleaned to ensure data readiness for comparison, reflecting rigorous preprocessing.
- **Hybrid Recommender Systems:** The multi-stage process of the project results in each stage producing an output which becomes the input to the next model, akin to Feature Augmentation referenced in Taxonomy by Burke, 2002.

Cognitive Systems

- **Vectorization with Embeddings** In the ground truth evaluation. We employed OpenAI model *text-embedding-3-small* to represent text as vectors in both the ground truth knowledge base and customer requirements.
- **Semantic Similarity** The EmbeddingMatcher evaluates how well policies cover specific customer requirements. Both customer requirements and ground truth knowledge base embeddings are projected into vector space and cosine similarity is computed to identify requirements that match.
- **Ground Truth Evaluation:** The project leverages multiple curated knowledge bases within `data/ground_truth/ground_truth.json` to assess system performance rigorously, evaluating both high-level scenario outcome correctness and granular requirement-level coverage against trusted data.
- **Multi-Step Reasoning Pipeline:** Instead of relying on a single, complex prompt, this project implements a multi-step reasoning pipeline (Extraction -> Comparison -> Recommendation). This decomposition guides the overall process, breaking the problem into manageable sub-tasks. This approach enhances transparency, allows for intermediate evaluation, and provides opportunities for debugging or intervention, contrasting with monolithic end-to-end LLM calls.
- **Prompt Engineering:** Prompt engineering is foundational. Prompts across the system (`scripts/data_generation/`, `scripts/extract_policy_tier.py`, `scripts/generate_policy_comparison.py`, `scripts/generate_recommendation_report.py`, evaluation scripts) are carefully crafted to be clear, goal-driven, and context-rich. The emphasis on requesting structured JSON output is a key technique to constrain LLM outputs and ensure reliability.
- **Intent & Slots Detection (Analogy):** The practice of designing prompts that expect structured JSON output can be viewed analogously to intent classification and slot filling. The prompt defines the "intent" (e.g., extract requirements, compare policies), and the JSON schema defines the "slots" the LLM needs to fill, simplifying the LLM's task and making the output predictable and usable.

Installation and User Guide

Detailed installation instructions on how to install the libraries and to set up API keys can be found here.

A dedicated demo script (`scripts/run_recsys_demo.py`) was created to run the entire backend recommendation system workflow. This script runs all the main steps automatically and creates a summary report showing what happened. Detailed instructions on how to run the demo workflow can be found here.

Important Cost Note: Running this demo script involves multiple calls to LLM APIs. Expect approximate costs of USD $3 for Google Gemini API usage and a few cents (USD) for OpenAI API usage per run. Costs may vary based on API pricing changes and specific run complexity.

To execute individual steps instead of the whole process, detailed instructions can also be found here. Note that LLM API costs will also apply.