

PRAGMADIC: PENANG-BASED DIGITAL NOMAD PLATFORM WITH RAG-INTEGRATED ANSWER ENGINE

Name: Matthew Loh Yet Marn

Coventry Student ID: 13445539

Supervisor Name: Ms. Usha Jayahkudy

Bachelor of Science (Hons) in Computer Science, 3+0

In Collaboration with Coventry University

School of Computing, INTI International College

Penang

21st June 2024

6001CEM Declaration of Originality

I declare that this project is all my own work and has not been copied in part or in whole from any other source except where duly acknowledged. As such, all use of previously published work (from books, journals, magazines, internet etc.) has been acknowledged by citation within the main report to an item in the References or Bibliography lists. I also agree that an electronic copy of this project may be stored and used for the purposes of plagiarism prevention and detection.

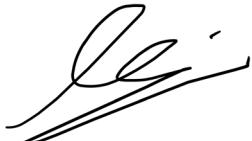
Statement of copyright

I acknowledge that the copyright of this project report, and any product developed as part of the project, belong to Coventry University. Support, including funding, is available to commercialise products and services developed by staff and students. Any revenue that is generated is split with the inventor/s of the product or service. For further information please see www.coventry.ac.uk/ipr or contact ipr@coventry.ac.uk.

Statement of ethical engagement

I declare that a proposal for this project has been submitted to the Coventry University ethics monitoring website (<https://ethics.coventry.ac.uk/>) and that the application number is listed below (Note: Projects without an ethical application number will be rejected for marking)

Signed:



Date: 8 November 2024

Please complete all fields.

First Name:	Matthew
Last Name:	Loh Yet Marn
Student ID number	13445539
Supervisor Name	Ms. Usha Jayahkudy

This form must be completed, scanned and included with your project submission to Turnitin. Failure to append these declarations may result in your project being rejected for marking.

ACKNOWLEDGEMENT

I would like to acknowledge the various sources of motivation and inspiration that powered me throughout the entirety of this project. I want to first extend my deepest appreciation to the sacrifices my parents made to provide me with this opportunity and for always continuously supporting me in my endeavours. The journey would have been exponentially more difficult without the holistic support gracefully provided as I accumulated more and more burdens and challenges as the going got tough. I would also like to take this opportunity to extend my heartfelt appreciation to my supervisor, Ms. Usha Jayahkudy for exercising her sound judgement and patience with my many bouts of setbacks and burnout. I would like to acknowledge the learning environment provided to me by INTI International College Penang and Coventry University over the last 3 years. All in all, I have grown deeper in my maturity and expanded my skill set.

TABLE OF CONTENTS

ACKNOWLEDGEMENT	ii
TABLE OF CONTENTS	iii
LIST OF TABLES	iv
LIST OF FIGURES	iv
LIST OF ABBREVIATIONS	iv
ABSTRACT	v
CHAPTER 1: INTRODUCTION	1
1.1 Introduction	1
1.2 Problem Statement	1
1.3 Research Questions	6
1.4 Keywords	6
1.5 Objectives	6
1.6 Clients, audience and motivation	7
1.7 Project Scope and System Limitations	8
1.8 Chapter Summary	9
CHAPTER 2: LITERATURE REVIEW	10
2.1 Introduction	10
2.2 Digital Nomadism, MDEC's DE Rantau Nomad Pass and Existing System	10
2.2.1 Global Comparison of Digital Nomad Visa Programs	10
2.2.2 Existing Systems – DE Rantau by HostAStay	15
2.2.3 Citizen Remote	17
2.2.4 Product Requirements of PRAGmadic	20

2.3 Overview of JavaScript Meta-Frameworks	23
2.3.1 Next.js	25
2.3.2 Remix	26
2.3.3 SolidStart	27
2.4 Analysis of Backend-as-a-Service Providers	29
1.1.1 Supabase	29
2.4.1 Firebase	31
2.4.2 AWS Amplify	32
2.5 Production-scale AI Infrastructure in Large Language Models for Embeddings and Retrieval-Augmented Generation	33
1.1.1 OpenAI API	33
2.5.1 Amazon Bedrock	35
2.5.2 Open-source Large Language Models as Infrastructure –Locally Hostable Ollama Services	37
2.6 Infrastructure as a Service for Web and AI Deployments and CI/CD on the Edge	38
1.1.1 Vercel	38
2.6.1 Render	40
2.6.2 Amazon Web Services	41
2.7 Proposed Work	42
1.1.1 Retrieval Augmented Generation System Architecture	42
2.7.1 Overall Frontend and Backend Technology Selection	46
2.7.2 High-level Overview of Technology Integration	48

2.8 Conclusion	49
CHAPTER 3: RESEARCH METHODOLOGY	50
3.1 Introduction	50
3.2 Prototyping Model	50
3.2.1 Communication / Requirements Gathering	51
3.2.1(a) Questionnaire Analysis	52
3.2.2 Planning	65
3.2.2(a) Gantt Chart	66
3.2.2(b) Kanban Board	67
3.2.3 Modelling	68
3.2.3(a) Activity Diagram	68
3.2.3(b) Use Case Diagram	72
3.2.3(c) Entity Relationship Diagram (ERD)	74
3.2.4 Construction	79
3.2.4(a) Developer Environment	79
3.2.4(b) Version Control Strategy – Feature Branching	80
3.2.5 Testing	82
3.2.5(a) Black Box Testing	82
3.2.5(b) Integration Testing	83
3.2.5(c) Security Testing in Web and LLM-based Applications (Non-functional)	
84	
3.2.6 Deployment and Maintenance	84

3.2.7 Feedback and Iteration	85
3.3 Conclusion	86
CHAPTER 4: PROJECT IMPLEMENTATION	88
4.1 Introduction	88
4.2 Login and Registration Module	90
4.2.1 Login and Registration with Supabase - OAuth, Magic Link, Email and Password	91
4.2.2 Role-based Access Control (RBAC) Implementation using Custom Claims	94
4.2.3 Profile Management and Dashboards	98
4.3 Chatbot Module	107
4.3.1 Chat Interface	107
4.3.2 Knowledge Base Management	113
4.3.3 Document Processing and Retrieval	119
4.4 Onboarding Support Module	123
4.4.1 Rich Interactive Mapbox Onboarding Support Map Component	124
4.4.2 Browsing Location-based Events	126
4.4.3 Hub and Communities Events	128
4.5 Analytics Module	142
4.5.1 Data Retrieval and Visualization	142
4.5.2 LLM-based Unstructured Data Processing and Suggestions Generation	144
4.5.3 LLM Interactivity and Other Metrics	146
4.6 Deployment Readiness through Vercel and Supabase	153

4.7 Conclusion	157
CHAPTER 5: TESTING AND EVALUATION	158
5.1 Introduction	158
5.2 Black Box Testing	158
5.2.1 Login and Registration Module Black Box Testing	158
5.2.2 Getting Started Module Black Box Testing	161
5.2.3 RAG Chatbot Module Black Box Testing	162
5.3 Integration Testing	163
5.3.1 Hub Module Integration Testing	164
5.3.1(a) Hub Creation Flow	164
5.3.1(b) Events and Hub Integration	164
5.3.1(c) Events, Markers and Participation Integration Testing	165
5.3.2 Hub Analytics Integration Testing	165
5.3.2(a) Analytics Data Flow Testing	165
5.3.2(b) Chart Libraries and Data Processing Testing	166
5.3.2(c) Analytics Chatbot Tool Calling Integrations	167
5.4 Security Testing in Applications with Web and LLM Integrations	167
5.4.1 Rate Limiting	167
5.4.2 LLM Guardrails	168
5.4.3 Access Control	169
CHAPTER 6: DISCUSSION	171
6.1 Achievements	171

6.2 Limitations	171
6.3 Future Enhancements	171
6.4 Conclusion	171
CHAPTER 7: REFLECTION	172
7.1 Strengths	172
7.2 Challenges Overcame	172
7.3 Self-improvement	172
CHAPTER 8: CONCLUSION	173
REFERENCES	174
APPENDICES	177

LIST OF TABLES

Table 2.1: Feature and Requirements Comparison of Existing Systems	20
Table 2.2: Classification of Feature Priority	21
Table 2.3: Overview of JavaScript Meta-framework Criteria	23
Table 3.1: Brief Overview of Key Entities and Relationships of Pragmatic	76
Table 5.1: Login and Registration Black Box Testing Table	158
Table 5.2: Getting Started Module Black Box Testing Table	161
Table 5.3: RAG Chatbot Module Black Box Testing Table	162

LIST OF FIGURES

Figure 1.1: Statistical Summary of DE Rantau Nomad Pass, DE Rantau Nomad Hubs and DE Rantau Ecosystem up to June 2024 (Appendix A)	2
Figure 1.2: Problem Outline of Proposed Project	5
Figure 2.1: The Estonian DNV launched in 2020 in contrast to e-Residency (Republic of Estonia, n.d.)	11
Figure 2.2: Statistical infographic of the DE Rantau Nomad Pass Applications up to July 2023 (Aziz, 2023a)	13
Figure 2.3: DE Rantau Mobile Application Features	15
Figure 2.4: HostASStay, a DE Rantau-based short-stay accommodation booking platform.	16
Figure 2.5: Citizen Remote Onboarding Screen (Citizen Remote, n.d.)	17
Figure 2.6: Citizen Remote guides digital nomad visa application (Citizen Remote, n.d.)	17
Figure 2.7: Citizen Remote is partnered with local advisors (Citizen Remote, n.d.)	18
Figure 2.8: Comparison of JavaScript Meta-frameworks in 2024 (Hempenius & Osmani, 2024)	23
Figure 2.9: Logo of Next.js 14, the latest stable iteration of Next.js as of writing	25
Figure 2.10: In essence, Remix is a framework built upon React Router and Vite (Lybrand, B., 2024)	26
Figure 2.11: SolidStart 1.0 is a JavaScript Framework designed to build SolidJS apps	27
Figure 2.12: SolidStart describes itself as a Composable Meta-framework with Rock-Solid Developer Experience. (SolidStart, 2024)	28
Figure 2.13: Logo of Supabase	29
Figure 2.14: Supabase has entered general availability on multiple platforms	30
Figure 2.15: Firebase	31

Figure 2.16: AWS Amplify	32
Figure 2.17: OpenAI Embeddings API	34
Figure 2.18: OpenAI Chat Completions API (OpenAI, n.d.,)	35
Figure 2.19: Amazon Bedrock	35
Figure 2.20: AWS Console for Amazon Bedrock and provided Foundational Models	36
Figure 2.21: Key use cases for Amazon Bedrock including integration into preexisting infrastructure	37
Figure 2.22: Ollama is an open-source project to make running LLMs on local machine seamless	37
Figure 2.23: Vercel's logo	39
Figure 2.24: Vercel automatically infers project structure to optimize deployments	39
Figure 2.25: Render's logo	40
Figure 2.26: Render's built-in support for various technologies	41
Figure 2.27: Amazon Web Services Example Cloud Service Architecture	41
Figure 2.28: A historical analysis of the evolution of early neural networks to the current transformer lightning-in-a-bottle landscape	43
Figure 2.29: Proposed RAG Architecture	45
Figure 2.30: Proposed Project Implementation Architecture	48
Figure 3.1: The Prototyping Paradigm (Roger & Bruce, 2015)	51
Figure 3.2: Percentage of respondents based on category of applicant / digital nomad	52
Figure 3.3: Demographic Questions (Pre-application & Pre-arrival Digital Nomads)	54
Figure 3.4: Motivation to Pursue DN Lifestyle (Pre-application & Pre-arrival Digital Nomads)	55

Figure 3.5: Information Gathering Methods (Pre-application & Pre-arrival Digital Nomads)	55
Figure 3.6: Degree of Difficulty Faced in Various Aspects of Researching (Pre-application & Pre-arrival Digital Nomads)	56
Figure 3.7: Degree of Difficulty Faced in Various Categories of Information (Pre-application & Pre-arrival Digital Nomads)	56
Figure 3.8: Demographic Information (Current & Completed Digital Nomads)	57
Figure 3.9: Motivation Information (Current & Completed Digital Nomads)	58
Figure 3.10: Information Searching Methods (Current & Completed Digital Nomads)	59
Figure 3.11: Perceived Difficulty of Research Aspects (Current & Completed Digital Nomads)	59
Figure 3.12: Perceived Difficulty of Categories of Information (Current & Completed Digital Nomads)	60
Figure 3.13: Challenges Experienced as Digital Nomads (Current & Completed Digital Nomads)	60
Figure 3.14: Existing Systems and Platforms Used (Current & Completed Digital Nomads)	61
Figure 3.15: Potential Generative AI Features (Current & Completed Digital Nomads)	62
Figure 3.16: Interest in Community Platform (Current & Completed Digital Nomads)	62
Figure 3.17: Key Networking Opportunity Options (Current & Completed Digital Nomads)	63
Figure 3.18: Key Feedback on DE Rantau Program (Current & Completed Digital Nomads)	64
Figure 3.19: Gantt Chart with Overview of Detailed Tasks	66
Figure 3.20: Kanban Board of Pragmatic	67

Figure 3.21: Activity Flow Diagram of DE Rantau Nomad Pass Visa Application and Approval Process	69
Figure 3.22: Activity Diagram of Pragmatic Answer Engine	70
Figure 3.23: Activity Diagram of Pragmatic Onboarding Support Platform	71
Figure 3.24: Use Case Diagram for RBAC and Registration	72
Figure 3.25: Additional Use Case Diagrams for RAG Chat and Onboarding	73
Figure 3.26: Entity Relationship Diagram of Pragmatic	75
Figure 3.27: Cursor with Pragmatic Main Branch Open	79
Figure 3.28: GitHub and Git as Version Control Tools	80
Figure 3.29: GitHub Repository of Pragmatic	81
Figure 3.30: Git CLI used to create feature branches	82
Figure 3.31: OWASP Top 10 for Large Language Model Applications (OWASP, 2024)	84
Figure 4.1: Pragmatic Landing Page	89
Figure 4.2: Login Page	90
Figure 4.3: Registration Page	90
Figure 4.4: auth-actions.ts: login function uses signInWithEmailAndPassword from supabase.auth	91
Figure 4.5: auth-actions.ts signup Server Action for registration by Email and Password	92
Figure 4.6: Magic Link and OAuth logins use API routes for Proof Key for Code Exchange (PKCE) flows	93
Figure 4.7: Users are successfully created into the Supabase instance using OAuth and Magic Link	94
Figure 4.8: Pragmatic uses triggers to manage new users across the various possible login and registration verification methods	95

Figure 4.9: The set _user _role function is used to provide custom claims for each JWT	96
Figure 4.10: Client-side and Server-side functions to decode the JWT through Auth Hooks.	97
Figure 4.11: New user is successfully created using Email Password sign up.	98
Figure 4.12: Dashboard for Newly Registered User	98
Figure 4.13: Getting started page for new users to create base profile information	99
Figure 4.14: Successful Profile Creation and Profile Picture Uploaded	99
Figure 4.15: Role Selector	100
Figure 4.16: Create Nomad Profile and Obtain Nomad Role	101
Figure 4.17: Create Hub and Obtain Hub Owner Role	102
Figure 4.18: Code snippet for rendering profiles and roles of user.	103
Figure 4.19: Forms for nomad and hub owner utilize React Query for client-side asynchronous state management	104
Figure 4.20: Hub Dashboard (Unpopulated and Populated) for Hub Owner	105
Figure 4.21: Code Snippet for DashboardSwitcher	106
Figure 4.22: RAG-Integrated Answer Engine of Pragmatic	107
Figure 4.23: ChatContentClient.tsx	108
Figure 4.24: Chat Settings Sidebar Component and Model Selector	109
Figure 4.25: Document Selector Code	110
Figure 4.26: Example conversation by referencing a knowledge base document	111
Figure 4.27: Ability to View Past Chats, Delete Chat and Select Different Models	111
Figure 4.28: Ability to upload images for context	112

Figure 4.29: Admin-exclusive Sidebar Navigation Links and Middleware to Protect Admin Routes	114
Figure 4.30: Admin Management Dashboard	115
Figure 4.31: Knowledge Base Management Page of admin/chat	115
Figure 4.32: Folder View (DE Rantau Visa)	116
Figure 4.33: File Upload Component built with React Drop Zone with loading states	116
Figure 4.34: File Upload and File List Components Code Snippets	117
Figure 4.35: generateEmbeddingsForPdfAction server action	119
Figure 4.36: The chunked and embedded document is stored in the document_chunks table with related metadata.	120
Figure 4.37: chat/route.ts is the endpoint which is called by useChat hook	121
Figure 4.38: Custom Model used to Wrap Different Providers with ragMiddleware	122
Figure 4.39: Onboarding Component available to all users	123
Figure 4.40: Interactive Map Component using MapboxGL under the hood	124
Figure 4.41 Dark Mode Aware Styling and Ability to Control Map using Mapbox APIs	124
Figure 4.42: Dark mode Integrated Styling and Ability to Control Map using Mapbox APIs (Geocoding)	125
Figure 4.43: Browsing Events on the MapSidebar and MapComponent	126
Figure 4.44: Onboarding Map Page fetches data on the server to be consumed by MapComponent as props	127
Figure 4.45: Hubs Browsing Page (Owner View)	128
Figure 4.46: Hub Member and Nomad View	129
Figure 4.47: Owner View (Alternate)	130

Figure 4.48: Hub Creation and Edit Form	131
Figure 4.49: Hub View (Owner Profile Info Tab)	132
Figure 4.50: Hub View Events Tab (Hub Owner and Event Creator)	133
Figure 4.51: Hub Reviews Tab View	134
Figure 4.52: Hub Page Members View (Owner of Hub)	135
Figure 4.53: Hub Member Request Creation	136
Figure 4.54: Hub Member Join Request Rejected	136
Figure 4.55: Hub Member Join Request Accepted and Leave Button	137
Figure 4.56: Create Event Form (Owner / Admin Member)	137
Figure 4.57: Event View (Creator)	138
Figure 4.58: Participant View (“member” Role)	139
Figure 4.59: Communities List Page	140
Figure 4.60: Community Posts of a Community	141
Figure 4.61: Analytics Module View for Hub Owner	142
Figure 4.62: Pie Chart visualization of member role breakdown and growth details	143
Figure 4.63: Analytics Chat supports Member and Review analytics	144
Figure 4.64: Animated Prompt Suggestions	145
Figure 4.65: Analytics Chatbot is built on Vercel AI SDK toolcalling	146
Figure 4.66: memberStatsTool for the LLM to call to retrieve data	147
Figure 4.67: eventStatsTool to retrieve event statistics and participation date	148
Figure 4.68: participationTool	148

Figure 4.69: hubMetricsAnalysisTool	149
Figure 4.70: Review Analysis Tool, the most complex tool	150
Figure 4.71 Engagement Metrics Card	151
Figure 4.72: Calculations and formulas for metrics	152
Figure 4.73: Pragmatic Project on Vercel Hobby Tier	153
Figure 4.74: Pragmatic Supabase dashboard on Hobby Tier	154
Figure 4.75: OpenAI Platform Billing and Usage Dashboards	156
Figure 5.1: Example of Upstash Rate Limit implemented to protect chat API endpoint	168
Figure 5.2: Example of Multi-step RAG Implemented for Accuracy	169

LIST OF ABBREVIATIONS

BaaS	Backend-as-a-Service
RAG	Retrieval Augmented Generation
MDEC	Malaysia Digital Economy Corporation
DN	Digital Nomad
API	Application Programming Interface
LLM	Large Language Model
SSR	Server-side Rendering
CSR	Client-side Rendering
SSG	Static Site Generation
CI/CD	Continuous Integration / Continuous Delivery
IaaS	Infrastructure-as-a-Service
UI/UX	User Interface / User Experience
CNN, RNN, LSTM	Convolutional Neural Networks, Recurrent Neural Networks, Long Short-Term Memory

ABSTRACT

PRAGmadic is a web-based platform developed to facilitate the onboarding support of digital nomads (DNs) in Malaysia, specifically targeting applicants and members of the DE Rantau Nomad Pass visa program. This project addresses the critical problem of fragmented information and lack of centralized resources for digital nomads, which hampers the ability to seamlessly integrate into new environments and capitalize on local opportunities. The primary objective of PRAGmadic is to create a maintainable resource and service platform that utilizes Retrieval Augmented Generation (RAG) and large language models (LLMs) to aggregate and provide real-time, personalized information to DNs. The scope includes features for chat-based onboarding during the application and post-arrival phase and features catering to coworking space networking. The region targeted by the system falls under the region of Penang, Malaysia. The research methodology involves a prototyping approach to iteratively develop and refine the platform, leveraging Next.js for the frontend, Supabase for the backend, and OpenAI and Ollama for AI capabilities. Functional and non-functional testing methodologies, such as black box testing, integration testing, and Google Lighthouse audits, are discussed to ensure robust performance, accessibility and compatibility. The Prototyping model is selected for the ability to adapt well to changes in the underlying stakeholder requirements and also the rapidly iterating LLM environment. The expected outcomes of the project include a fully integrated platform that significantly reduces the information obfuscation for DNs, enhances the ability to find and connect with the local digital nomad ecosystem, and supports local businesses in the DE Rantau Hub Partner program.

CHAPTER 1: INTRODUCTION

1.1 Introduction

The contents of this chapter will first extensively cover the problem statement before proceeding to serve as the foundation and precursor to the overall project report. In said problem statement, the report will expand on the current challenges and opportunities presented by the steady emergence of digital nomadism in relation to Malaysia's response to a dynamic digital economy. Then, further details will be brought in within the context of Penang's response to the economic channels of digital nomadism that aim to act as the project's core focus. After the codification of the problem, research questions and keywords will be distilled to provide structure and guide the project. Subsequently, the key objectives of the project closely related to the problem statement will be defined such that these objectives are precisely formed according to the SMART criteria. The boundaries of the project will be drawn in the aftermath of objective formation to ensure achievability and narrowing of feature creep in the project scope and system limitations subchapters before the chapter is summarized.

1.2 Problem Statement

The emergence of digital nomadism has fundamentally transformed how a unique group of highly-skilled professionals work and travel, offering freedom and flexibility to those who choose to pursue a location-independent lifestyle. However, the allure of working transnationally comes with digital nomads (DNs) often experiencing significant challenges in finding suitable policies to accommodate a vast range of differing personal circumstances. The systematic analysis of Bednorz, (2024) expands on the current global landscape of digital nomad visas as of January 2023. From the findings focused on the supply side, Malaysia in particular has made concerted economic efforts in growing its digital economy, capitalizing on attracting high-skilled workers in the digital sector. More importantly, the policies stand out among competitors due to the policies' exceptionally permissive nature to allow DNs to be able to integrate into the local Malaysian workforce and work for local companies (Bednorz, 2024). Notably, another policy of the visa is the ability for pass holders to enter Malaysia with immediate family members, which is unlike a traditional professional visit pass.

Total Approved Applications

**1,443 applicants out of 3,096 submissions
(46.6%)**

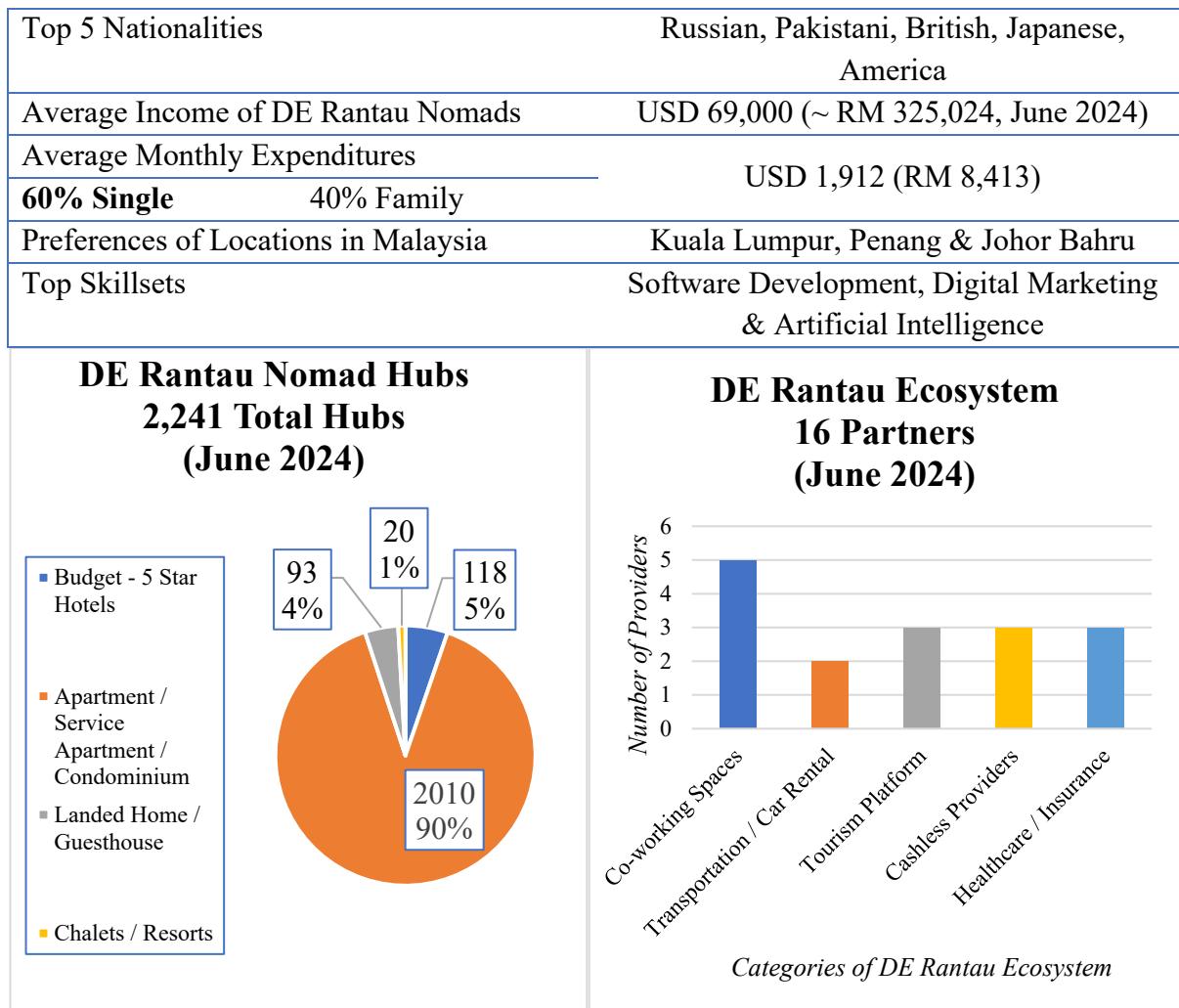


Figure 1.1: Statistical Summary of DE Rantau Nomad Pass, DE Rantau Nomad Hubs and DE Rantau Ecosystem up to June 2024 (Appendix A)

In response to the digital nomad movement, enter Malaysia's catalyst in the new frontier of tourism revolving around digital nomadism, digital professional mobility and digital economic development. The Malaysian government's Digital Economy Rantau (DE Rantau) program aims to establish Malaysia as a preferred digital nomad hub in ASEAN (Malaysia Digital Economy Corporation, 2024). In an interview with BERNAMA TV dated the 10th of April 2024, the chief executive officer of MDEC, Ts. Mahadhir Aziz, reports that since the inception of the DE Rantau program, the nationality composition of DNs ranges from European countries to Japan, Australia and more. In the program's operation for the year 2023, it was reported that out of 2600 applicants that were audited in coordination with the Ministry of Home Affairs of Malaysia, over 1000 applications were successfully approved (Aziz, 2024).

A strong pulse of the digital nomad lifestyle transformation can be identified in the cities of Kuala Lumpur and George Town (Boluda Chova & von Ehrlich-Treuenstätt, 2023). von Ehrlich-Treuenstätt specifically describes an empirical finding from an interview revealing a lack of deeper onboarding assistance for digital nomads (DNs). This is evidenced by an individual DN, who had frequented Malaysia multiple times, describing the lack of communication on topics like business opportunities, settling down or DN rights by the government after successfully receiving the DE Rantau pass and entering the country. This deficiency is a symptom of a broader challenge in the objective of attracting DNs, which is the absence of a centralized and accessible resource hub to navigate the exhaustive information available to DNs. Researching information is challenged by fragmented sources at each step of the process, from the processing documents in the visa application stage to understanding accommodation options, local laws, and cultural nuances. The absence of a cohesive platform focused on the personalization of information to individual DNs also worsens the information obfuscation, hindering the ability to seamlessly integrate into new locations and subsequently maximize local economic spill-over gains from opportunities like networking.

The preferred location of digital nomads in Malaysia consists of a majority 73% in Selangor and Kuala Lumpur (Aziz, 2023b), consequently, there is a necessity to create a greater draw of DNs to benefit Penang's economic spillover reception. The rise of coworking spaces has made George Town a hotspot for networking, collaboration, cultural integration, knowledge and skill-sharing, and onboarding support for digital nomads (Boluda Chova & von Ehrlich-Treuenstätt, 2023). Therefore, the motivation of the project takes a multi-pronged approach to improve DN accessibility aside from just essential information related to the application process of the digital nomad visa, but also to accommodate for onboarding successful DN applicants interested in finding a home base in Penang. To accurately gauge features necessary for the application, a questionnaire will be executed for the project and is attached as Appendix B.

Therefore, to address these challenges of exhaustive information and provide solutions for the problems faced by digital nomads who want to apply for the DE Rantau program and then explore locations like Penang for career and lifestyle opportunities, the proposed project is the development of PRAGmadic, a web-based platform designed to utilize large language models to aggregate through an answer engine using retrieval-augmented generation (RAG) the publicly available DE Rantau related information for both tourism and business-focused digital

nomads. to seamlessly both create and keep up with local networking opportunities at locations such as coworking spaces. The goal is to also allow these local co-working space owners to use the platform to assimilate information and provide real-time feedback on questions digital nomads may have regarding the owners' establishment.

**Fragmented Information
and Lack of Centralized
Resource Hub
(Visa Application Process)**

Description:
Difficulty in finding relevant, personalized and up-to-date info. Fragmented and varied sources.
Proposed Solution:

- Development of Pragmatic: An AI-based platform to aggregate, host and provide seamless access to essential information.
- Implement an AI Answer Engine trained answer accurately based on DE Rantau information.
- Utilize large language models to create tooling for querying comprehensive resource hub.

**Inadequate Onboarding
Assistance and Post-arrival
Communication
(Pre-arrival & Post-arrival)**

Description:
Lack of clear and responsive communication on key topics post-arrival.
Insufficient onboarding support for digital nomads, i.e: networking opportunities, coworking space and local digital nomad ecosystem
Proposed Solution:

- Provide real-time feedback and personalized assistance to new arrivals.

**Community Integration of
DE Rantau Nomad Hubs and
Coworking Spaces**

Description:
Draw digital nomads to Penang for economic spillover through a community-tailored platform.
Enhance coworking space information visibility and communication through personalized guides.
Proposed Solution:

- Create a platform for digital nomads to connect, network, and find local opportunities.
- Support local business owners by providing tools to assimilate information and offer real-time answers to nomads' queries.

Figure 1.2: Problem Outline of Proposed Project

1.3 Research Questions

- What type of issues are currently faced by digital nomads eager to operate in ASEAN and Malaysia in general that may lead to challenges in information accessibility and how can an onboarding support application better support both DNs and MDEC in optimizing the process?
- What features and functionalities should be incorporated into the platform to provide effective onboarding support for digital nomads in Penang, while also facilitating networking opportunities and increasing discovery through information dissemination by DE Rantau Hub Partners?
- What are the qualitative and quantitative analysis methodologies required to increase the effectiveness and accuracy of system features to digital nomads by incorporating retrieval-augmented generation?

1.4 Keywords

Web application; retrieval augmented generation; digital nomad visa information; onboarding support platform.

1.5 Objectives

- To develop a web-based platform that delivers maintainable and easily accessible insights based on a comprehensive and cited AI-aggregator answer engine utilizing retrieval-augmented generation on first-party DE Rantau information for DE Rantau's nomad pass/visa applicants.
- To develop a platform for the onboarding support needed by digital nomads to immerse and connect with locals in Penang, simultaneously providing platform-facilitated opportunities to network and engage with other digital nomads based on info dissemination by DE Rantau Hub Partners participants in Penang to additionally increase business discovery.
- To develop a data-driven and data visualization dashboard application to tailor dissemination and promotion of local Penang DE Rantau partners like co-working spaces and collaboration platforms to develop the DE Rantau presence in Penang.

1.6 Clients, audience and motivation

The clients of the RAG-based answer engine are digital nomads from the international digital nomad community who are interested in applying for the DE Rantau program to acquire a DE Rantau Nomad Pass visa and be granted tailored entrance to Malaysia. These highly mobile professionals seek a streamlined solution to the process of finding answers to questions based on an exhaustive amount of available first-party information or community-driven experience and insights. These individuals may be interested in retrieving information related to dynamic personal circumstances like visa eligibility requirements, documentation required, dependant-related inquiries, payment structure and so forth before going into the visa application process. Additional research may be required at a later time while the application and approval phase are carried out to plan for the pre-arrival stage of the overall DE Rantau process flow. Therefore, interested digital nomads may use the system to answer inquiries conversationally using a large language model to access answers retrieved from first-party updated documents curated, uploaded, and maintained by system administrators that emulate the role of MDEC and DE Rantau organizing authorities. By incorporating AI-based tools such as large language models (LLMs) and embedding technology, PRAGmadic aims to provide seamless natural language querying capabilities, and real-time personalization of information to these users.

The audience of the digital nomad onboarding support system consists of DE Rantau Nomad Pass holders who have already been approved after handing over the paperwork necessitated by MDEC, the Ministry of Home Affairs, and the Malaysian Immigration Department and have already entered Malaysia. The DE Rantau Nomad Pass holders may want to integrate with the local digital nomad communities to suitably plan for accommodation, services, networking opportunities at coworking spaces, and local happenings. The audience also consists of DE Rantau Nomad Hub Partners that include parties behind coworking spaces and accommodations, who may want to seize an advantageous position in the digital nomad ecosystem by providing business catering to digital nomads and organizing events. Coworking spaces in particular may utilize community-focused features facilitated by the system to help organize events and notify on other networking opportunities for local digital nomads.

The motivation of the system includes but is not limited to, desiring to deliver a minimum viable product in response to the current boom in LLM technology possessing conversational capabilities. The hypothesis of interest is investigating whether leveraging the lower barrier of entry in the current generative AI development landscape could have a positive effect on

solving difficult real-world challenges like onboarding support for digital nomads. The challenges that lurk behind each corner as various industries continue to lean into AI-integrated platforms will allow this project to be both a great learning opportunity as much as it aims to generate potential value for its users in time saved and problems solved.

1.7 Project Scope and System Limitations

The scope of the project is as follows:

- The PRAGmadic platform is deployed for digital nomads to have onboarding support in both the DE Rantau Nomad Visa program and related benefits including DE Rantau Nomad Hubs and DE Rantau Ecosystem Partners in Penang.
- The application targets using frontend technologies including but not limited to: Next.js, React, shadcn, TailwindCSS, Vercel AI SDK, Framer Motion and more.
- The application aims to utilize backend technologies such as Supabase Authentication, Storage, Edge Functions and Drizzle ORM as the ORM layer.
- The web application aims to be able to interface with both locally run and deployed large language models, namely Ollama using Llama3 8B by Meta AI or commercially available application programming interfaces (APIs) like OpenAI's API.
- The web application aims to provide digital nomads with an authenticated and secure platform to query information and utilize RAG-integrated features to conveniently access information of interest.
- The web application aims to provide a content management system that is integrated into the platform to update the repository of DE Rantau program information to authorized users.
- The web application intends to maintain best practices in AI safety and security when handling sensitive information related to the personal details of users required to apply for the digital nomad visas.

The system limitations initially include, but are not limited to:

- The web application's RAG system will only be developed with English language internationalization capabilities.

- The application will incur some operational overhead in AI-based querying response times depending on trade-offs made in accuracy versus computational costs.
- The system may require paid services such as the acquisition of credits under the payment structure of OpenAI's platform.

1.8 Chapter Summary

The contents of this chapter serve to introduce the development blueprint for a RAG-based platform that includes detailing the target problem and market. Digital nomads, the DE Rantau authorities, and DE Rantau Hub partners were identified as essential moving parts in a sophisticated system for an onboarding support platform for the digital nomads either established or potentially eager to get a soft landing in Penang. Developmentally, primarily, the system will be engineered using web technologies and aims to deliver on the essentials that come with the web including availability, security, authentication, authorization and accessibility. The platform will try to provide much-needed quality of life and convenience improvements where achievable using scalable and maintainable large language model technologies. Essentially, the application aims to utilize generative artificial intelligence in the form of large language models where logical and reliable to further facilitate digital nomad presence. The following chapter is the literature review, which will entail presenting the results of arduous and comprehensive research findings and analysis.

CHAPTER 2: LITERATURE REVIEW

2.1 Introduction

This chapter serves to introduce the fundamental literature and research material relevant to the development of a full-stack web application integrated with modern libraries and artificial intelligence application programming interfaces (API). Key topics first include an analysis of digital nomad visa policies from a global perspective. The statistical background on the DE Rantau Program as Malaysia's response up to July 2023 and its existing systems relevant to the project's requirements gathering are additionally included. Further topics discussed will be development-focused, such that the analysis could help minimize technical debt and ensure abstractions are used prudently. The technologies discussed include modern JavaScript Meta-Frameworks, modern Backend-as-a-Service (BaaS) providers, production-ready AI infrastructure for use cases including embeddings or text generation and finally modern Infrastructure-as-a-Service (IaaS) providers for the web and backend. Finally, higher-level artefacts depicting the environments upon which these technologies will be integrated will be provided, such as the Retrieval-Augmented Generation implementation architecture and preliminary system architecture diagram.

2.2 Digital Nomadism, MDEC's DE Rantau Nomad Pass and Existing System

The literature surrounding digital nomadism and the dynamic policies provided by the international stage has seen a meteoric rise, especially in the post-COVID-19 pandemic era of remote work. Digital nomad visas are generally specialized permits allowing individuals to live and work in a foreign country while being employed by an overseas company or operating through a self-owned business. These visas usually aim to attract skilled professionals, boost local economies, and diversify tourism revenue by encouraging longer stays and higher spending from digital nomads that continue to become more diverse in definition (Bednorz, 2024).

2.2.1 Global Comparison of Digital Nomad Visa Programs

From a global perspective, Estonia is notably recognized as the pioneer behind the concept of the digital nomad visa, allowing entrepreneurs eager to establish operations in the EU to work through digital operations. Estonia was quite early in its approach to adopting the human capital

approach (Bednorz, 2024), which served as inspiration for the inception of MDEC's DE Rantau Nomad Pass program.

The figure consists of two separate promotional sections. The top section is for 'e-Residency' and the bottom section is for 'digital nomad visa'. Both sections include a small image on the left, descriptive text in the center, and a call-to-action button on the right.

e-Residency:

- I want to apply for e-Residency**
- Apply to start a company 100% online & run your business remotely**
- ✓ No paperwork, physical meetings or bureaucracy**
- ✓ A digital way to do business from anywhere (Backed by Estonian Government)**
- ✓ Access Estonia's e-services for remote management of your business**
- Apply here**

Digital Nomad Visa:

- I want to apply for digital nomad visa**
- Apply for the digital nomad visa if you meet the following requirements**
- ✓ Ability to work independent of location & work remotely**
- ✓ Active employment contract with a company registered outside of Estonia, conduct business through your own company registered abroad, or work as a freelancer for clients mostly outside of Estonia.**
- ✓ Your income met the minimum threshold (€4,500 gross of tax) during the six months preceding the application**
- Apply here**

Figure 2.1: The Estonian DNV launched in 2020 in contrast to e-Residency (Republic of Estonia, n.d.)

The programs of Estonia such as the e-Residency and digital nomad visa program seen in Figure 2.1, share a substantial amount of overlap in terms of the intentions behind the provided policies with Malaysia. Some important aspects include both the Estonian DNV and DE Rantau Pass not granting a clear pathway to residency but rather tapping into the human capital potential of the social and economic migratory phenomenon (Bednorz, 2024). By allowing dependents to travel along with the pass holder subject to arbitration, these two programs share common authoritative initiatives. Tourism-wise, there are also common angles to provide direct services tailored to digital nomad lifestyles. Estonia's visa program however also draws the same criticism as Malaysia's given the higher barrier to entry in terms of income requirements. (Business Advisory Group on Migration, 2024). On the other hand, Malaysia's program has highly a strict adherence to only allowing certain job sectors of high-skilled digital workers.

Litania & Marsan (2023) provide a detailed analysis of the ASEAN landscape regarding digital nomad policies, emphasizing the region's efforts to attract global talent post-pandemic. ASEAN countries generally recognize digital nomad potential to drive innovation and economic growth, although strategies vary significantly. Singapore leads in attracting high-skilled professionals with its EntrePass, Tech.Pass, and Overseas Network and Expertise (ONE) Pass, targeting tech professionals, entrepreneurs, and top talents. These visas facilitate long-term residency and aim to position Singapore as a global innovation hub. Thailand and Indonesia adopt similar strategies with the respective Long-Term Residence (LTR) Visa and Golden Visa, aiming to attract affluent global citizens and high-skilled professionals. These programs encourage investment and economic growth in advanced technological sectors.

Other ASEAN countries as analysed by Litania & Marsan (2023), like Brunei and Cambodia, offer alternative visa options for business owners and professionals that are not directly aimed at digital nomads who prefer long short-stay durations, for example, up to more than 3 months at a time. The Philippines and Vietnam, meanwhile, focus on fostering entrepreneurial ecosystems through startup and investment visas, targeting business creators and innovators. The varied approaches highlight ASEAN's recognition of global talent mobility. In summary, from the ASEAN perspective, successful implementation of digital nomad and talent visa schemes depends on creating supportive ecosystems that integrate digital nomads into local communities, ensuring mutual benefits through knowledge exchange, and fostering innovation and entrepreneurship.

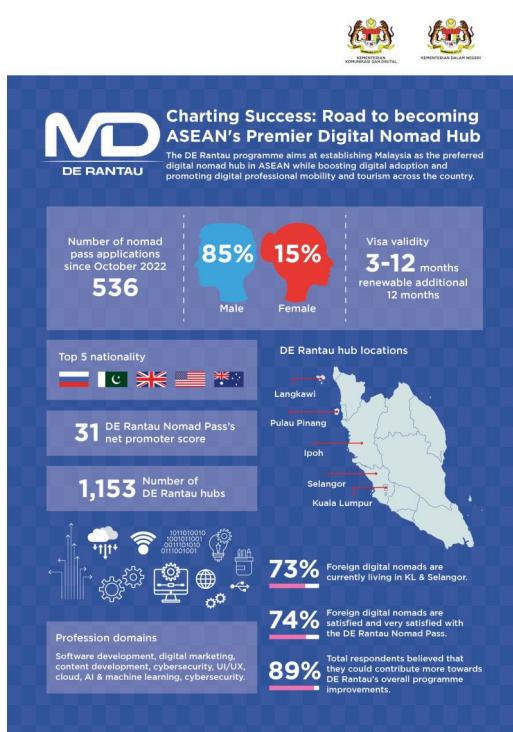


Figure 2.2: Statistical infographic of the DE Rantau Nomad Pass Applications up to July 2023 (Aziz, 2023a)

Up to July 2023, Malaysia's emergence as a favoured destination for digital nomads can be supported by statistical data seen in Figure 2.2 which explores various facets of its DE Rantau program. Key details included Malaysia's expansive network of 1,153 DE Rantau hubs and increasing, providing a glimpse at the nation's commitment to providing infrastructure conducive to digital nomadic lifestyles. Moreover, the growing interest in Malaysia is evidenced by the 1,435 applications for the Nomad Pass in just over a year since the program's inception (Aziz, 2023b), indicating a rising trend in digital nomadic migration to the country. This influx is then strengthened by the diverse nationalities of the digital nomads, with Russia, Pakistan, Britain, the US, and Australia leading the charge. The gender balance within Malaysia's digital nomad community, though still skewed towards males, demonstrates a gradual shift towards greater diversity, with females accounting for 15% of the demographic. This trend reflects the evolving composition of digital nomadic populations worldwide and also an opportunity to incorporate efforts of Sustainable Development Goals like gender equality.

The trend conveys the significance of digital infrastructure and urban development in attracting and retaining digital nomadic talent. Notably, however, despite being in the first few states to

be prioritized in the program rollout, Penang has yet to show a formidable presence relatively. Nonetheless, the high satisfaction rates (73%) and positive Net Promoter Score (NPS) of 31 highlight the effectiveness so far associated with the overall DE Rantau program in meeting the needs of nomads. These metrics reflect the program's success in facilitating a seamless transition for digital nomads. Furthermore, the receptiveness of digital nomads towards contributing to program improvements (89% of respondents) proves the collaborative ethos within Malaysia's digital nomadic community (Aziz, 2023b). This willingness to engage in feedback loops and iterative enhancements bodes well for the continuous evolution and refinement of the DE Rantau program.

In addition to the demographic and satisfaction statistics, Malaysia's attractiveness to digital nomads is highlighted by the calibre of professionals it attracts and the flourishing ecosystem it fosters. With an average salary of USD 70,000 (~RM 330,000 as of 2024) that surpasses the global average (Aziz, 2023a) and a diverse array of skillsets encompassing software development, digital creative content, and digital marketing, Malaysia continues to emerge as a suitable destination for highly skilled digital professionals seeking a conducive environment for remote work and freelance entrepreneurship, especially considering the relatively strong purchasing power of global economies in Malaysia. Overall, the statistical landscape surrounding Malaysia's digital nomad scene has a dynamic and inclusive ecosystem, poised to thrive in an increasingly digital-reliant economy as the workforce's composition expands to require external skills and talents.

To summarize, the DE Rantau program has provided a soft landing for DNs eager to integrate into the thriving nomadic ecosystems across Malaysia that also supports the economic prospects of local tourism-related business owners. This has included the establishment of certified nomad-ready accommodations with the creation of DE Rantau Hubs to benefit both nomads and local accommodations and coworking space service providers, contributing to accelerating the expansion of the digital tourism sector.

2.2.2 Existing Systems – DE Rantau by HostAStay



Figure 2.3: DE Rantau Mobile Application Features

The DE Rantau mobile app as seen in Figure 2.3, is an existing initiative between HostAStay, a short-stay platform company, and MDEC that was launched as part of the broader effort to enhance the digital nomad ecosystem in Malaysia, DE Rantau, under the program of the same name, is a mobile application available on both the Apple App Store and Google Play Store. This partnership leverages HostAStay's expertise in short-stay hospitality and MDEC's new-found commitment to promoting digital initiatives, especially in the digital economic aspect, resulting in a platform that caters to the mutually beneficial needs of digital nomads and short-stay accommodation providers. In essence, HostAStay acts as the pivot in the ecosystem by providing accommodation accessibility through means like consolidating the searching and booking activities and handling integration with the official DE Rantau verification system, therefore being a good example of tapping into the existing DE Rantau and MDEC infrastructure.

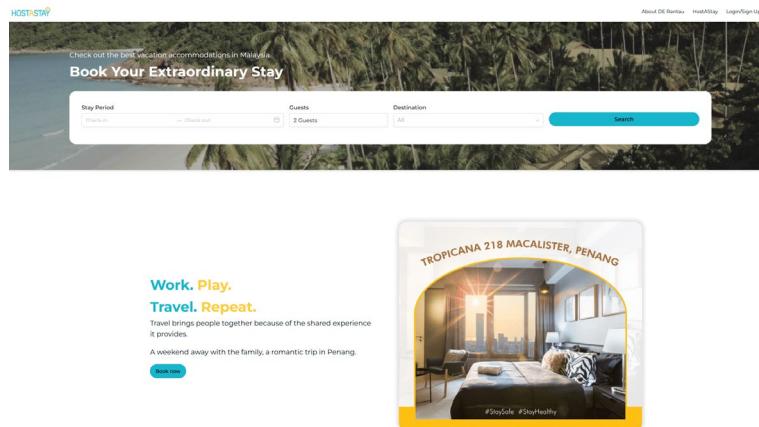


Figure 2.4: HostAStay, a DE Rantau-based short-stay accommodation booking platform.

On HostAStay, a similar web platform seen in Figure 2.4 is offered to ensure maximal support for various devices and platforms. Running alongside business operations is the availability of extra services on these platforms providing instruments for digital nomads to further expand operations through means like membership privileges, seamless booking and discovery of exciting deals. In summary, the DE Rantau mobile app is a crucial system in Malaysia's short stay and digital nomad sector. By combining HostAStay's industry expertise with MDEC's digital innovation capabilities, the app provides a consolidated and user-authenticated platform that supports both property owners and travellers.

2.2.3 Citizen Remote

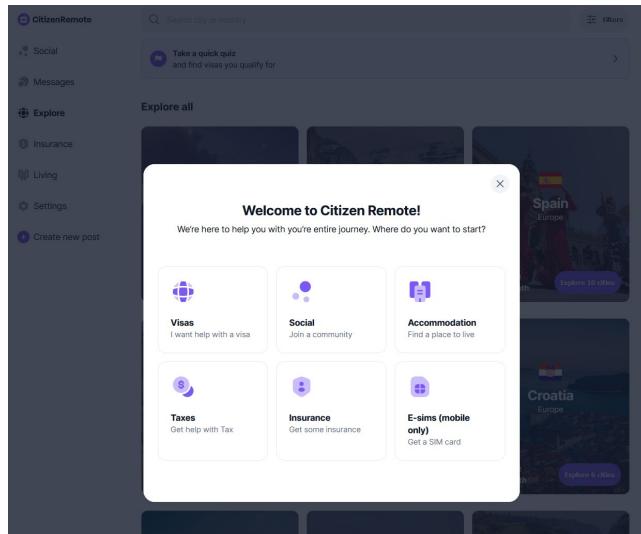


Figure 2.5: Citizen Remote Onboarding Screen (Citizen Remote, n.d.)

Citizen Remote which is pictured in Figure 2.5 is a platform designed to support digital nomads, expatriates, and remote workers by offering a suite of tools and resources that streamline the complexities of working and living abroad. This platform integrates features that cater to the specific needs of remote workers, providing a centralized source for information needed to manage various aspects of remote work and international living. Citizen Remote distinguishes itself by offering a wide range of features and services aimed at enhancing the digital nomad lifestyle in supported countries. These features are designed to address the challenges faced by digital nomads, from visa management to community engagement.

A composite image showing two parts of the Citizen Remote platform. On the left, a promotional card for visa applications features the headline "Take The Stress Out Of The Visa Application Process" and a subtext about specialized tools. It includes a "Get Started" button and a "Learn more" link. Below this are three service cards: "Explore Destinations" (determine visa qualifications and country fit), "Chat With Experts" (book consultations with experts), and "Hire Advisors" (work with immigration lawyers). On the right, a woman with glasses is smiling while looking at her phone. A floating overlay on the phone screen shows an "Application status" card with "Approved" in large letters, "25/25 items completed", and a message: "It looks like your application was approved! Personal Advisor is typing...".

Figure 2.6: Citizen Remote guides digital nomad visa application (Citizen Remote, n.d.)

One of the key features captured in Figure 2.6 of Citizen Remote is its comprehensive visa and immigration support. The platform provides detailed information on visa requirements for various countries including Malaysia, helping users understand the necessary steps for obtaining visas and permits. Additionally, Citizen Remote offers tools to track visa applications, ensuring that users stay on top of legal statuses and avoid potential issues. Citizen Remote centralizes the process of finding accommodation by offering a curated list of housing options although not officially partnered with local platforms.

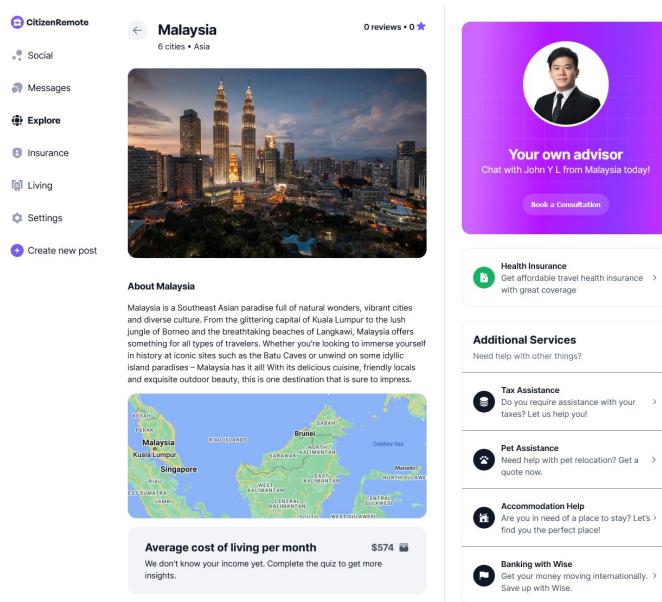


Figure 2.7: Citizen Remote is partnered with local advisors (Citizen Remote, n.d.)

The platform partners with local entities as seen in Figure 2.7 to offer services like information from legal authorities, ensuring that users have access to reliable channels of communication (Citizen Remote, n.d.). Citizen Remote fosters a sense of community by connecting users with other digital nomads and remote workers in an area. The platform features forums, events, and social meetups, allowing users to share experiences, seek advice, and build professional connections. Health and insurance are critical considerations for anyone living abroad. Citizen Remote offers resources and tools to help users find appropriate health insurance plans that meet certain needs. The platform also provides information on local healthcare services and emergency contacts, ensuring that users have access to essential healthcare support.

Managing finances and legal matters in a foreign country can be a complex process. Citizen Remote provides guidance on opening bank accounts, managing currency exchange, and understanding local tax regulations. The platform also offers legal resources to help users

navigate issues such as contracts, employment laws, and residency requirements. Citizen Remote is designed with a user-friendly interface that makes it easy for users to navigate and access its features. The platform is available on both the web and mobile devices, ensuring that users can manage remote work needs on the go. Citizen Remote also provides a comprehensive knowledge base and FAQs, ensuring that users have access to the information needed to make informed decisions. Citizen Remote stands out as a comprehensive solution for digital nomads and remote workers, offering a wide range of features that address the unique challenges of working and living abroad. Its focus on visa and immigration support, accommodation, community building, health services, and financial assistance makes it a valuable resource for anyone looking to streamline the experience of remote work.

2.2.4 Product Requirements of PRAGmadic

Table 2.1: Feature and Requirements Comparison of Existing Systems

Features/ Requirements	DE Rantau / HostAStay	Citizen Remote	PRAGmadic Implementation Readiness
Technical Requirements			
Platform Availability	Mobile (iOS, Android), Web	Mobile (iOS, Android), Web	Web-based (Mobile is Nice-to-Have)
Accommodation Booking	Yes, via HostAStay	Yes, curated list	Low Priority
Local Integration	Yes, through HostAStay	Only in groups	Networking, events from the community
Community Engagement	None / very limited	Yes, slowly expanding groups feature	Yes, a community platform for coworking spaces
Local Economic Integration	Yes, through accommodation services	Yes, local advisors and partnerships	Yes, enhancing local economic impact through networking and info dissemination
Healthcare Information	No	Yes, insurance plans & local healthcare info	Yes, basic personalization of info via profile
Financial & Legal Guidance	No	Yes, banking, tax, and legal resources	Yes, using LLMs to summarize officially uploaded documents
Membership Privileges	Advertised	Yes, various membership benefits/premium features	Not planned

Real-time Information	Not a focus	Yes, via local partnerships	Yes, via AI and embedding technology
Visa Application Process Related			
Visa & Immigration Support	Included authorization and authentication with DE Rantau verification, no support for process	Yes, detailed visa information & tracking	Yes – personalized information provided, using RAG-based Answer Engine
Digital Nomad Onboarding Support			
Target Audience	Digital nomads and short-stay accommodation providers	Digital nomads, expatriates, remote workers	Digital nomads, DE Rantau applicants, local businesses under DE Rantau
Onboarding Experience	No	Yes, comprehensive guidance	Yes, seamless onboarding via AI & LLMs

Based on Table 2.1, by comparing DE Rantau/HostAStay, Citizen Remote, and the proposed PRAGmadic platform, several potential requirements and features for PRAGmadic can be identified. DE Rantau/HostAStay and Citizen Remote provide a range of functionalities aimed at supporting digital nomads, such as accommodation booking, visa and immigration support, community engagement, and local economic integration. However, PRAGmadic aims to narrow the system scope according to feasibility constraints and instead enhance these offerings with a focus on AI-driven tools and seamless onboarding experiences.

Table 2.2: Classification of Feature Priority

Feature Category	Feature Description
Must-Have Features	
Platform Availability	Web-based (Mobile is Nice-to-Have)

Visa & Immigration Support	Personalized information is provided using a RAG-based Answer Engine
Onboarding Experience	Seamless onboarding via AI and LLMs
Community Engagement	Community platform for coworking spaces, networking, and events
Real-time Information	Up-to-date information via AI and embedding technology
Should-Have Features	
Healthcare Information	Basic personalization of healthcare info via user profiles
Financial & Legal Guidance	Summarizing official documents using LLMs to provide relevant information
Local Integration	Networking events and community activities to enhance user engagement
Nice-to-Have Features	
Accommodation Booking	Integrating accommodation options (low priority)
Membership Privileges	Considering membership benefits (if time allows)
Local Economic Integration	Detailed networking and information dissemination to enhance local economic impact

According to the classification in Table 2.2, PRAGmadic will prioritize must-have and should-have features, with nice-to-have features being considered if there is adequate time for development. This structured approach ensures that the platform will effectively meet the needs of digital nomads while leveraging advanced AI tools for a superior user experience.

2.3 Overview of JavaScript Meta-Frameworks

	Angular	Astro	Next.js	Remix	Nuxt	SolidStart	SvelteKit
Common features	●	●	●	●	●	●	●
	●		●	●	●	●	●
	●	●	●	●	●	●	●
	●	●	●	●	●	●	●
		●	●	●	●	●	●
	●	●	●	●	●	●	●
	●		●	●	●	●	●
			●	●	●	●	●
Recent trends	● (Signals)		● (Forget)	● (Forget)	● (shallowRef)	● (Signals)	● (Runes)
	●	●	● (partial prerendering)			●	
			●		●		
	●	●	●		●		●

Figure 2.8: Comparison of JavaScript Meta-frameworks in 2024 (Hempenius & Osmani, 2024)

In this section, the crucial step of choosing a suitable stack of technologies will be carried out based on the official first-party documentation provided by the technologies' stakeholders. In the context of the project's frontend tech stack, a JavaScript meta-framework is a higher-level framework that is built on top of a core JavaScript framework (such as React, Vue, or Angular) to provide additional features, abstraction, and capabilities that streamline the development process and especially dictate the potential performance, scalability, and maintainability of web applications. These meta-frameworks typically offer important conveniences for developers evolved from modern web development innovation. In this comparison, some of the features focused on will be as follows in Table 2.3:

Table 2.3: Overview of JavaScript Meta-framework Criteria

Aspect	Description
Complexity of Setup and Configuration	The degree to which these meta-frameworks handle boilerplate and configuration to get a project up and running, often based on highly opinionated and web convention-abiding approaches.
Support for Modern Rendering Patterns	Meta-frameworks follow different rendering patterns that stand out for improving performance and user experience on the client-facing

(i.e.: Server-side Rendering (SSR) and Static Site Generation (SSG))	side while providing search engine optimization capabilities through pre-rendering pages.
Routing	Routing comes out of the box with modern meta-frameworks, which in and of itself is a challenge to setup and optimize, therefore built-in robust support for features like dynamic routes, nested routes and code-splitting have emerged. The ability of a meta-framework to conform to newly established standards such as the introduction of React Server Components is also a welcome addition.
Build tooling optimizations	Modern web frameworks are highly competitive and often innovate on solutions focused on developer tooling, ensuring the final build output of websites is production-optimized.
Developer experience and deployment ergonomics-focused features	Developer ergonomics is crucial for testing, hot module reloading, error/warning handling, linting and debugging tools. The stack chosen requires mature deployment principles that can leverage different kinds of cloud computing providers.
Platform-specific tool maturity and rate of adoption of cutting-edge features	Proprietary experimental features and the stability of these implementations greatly influence specific use cases. Therefore, mature tooling should be well-documented, thoroughly tested, and widely adopted with ample long-term support.

2.3.1 Next.js

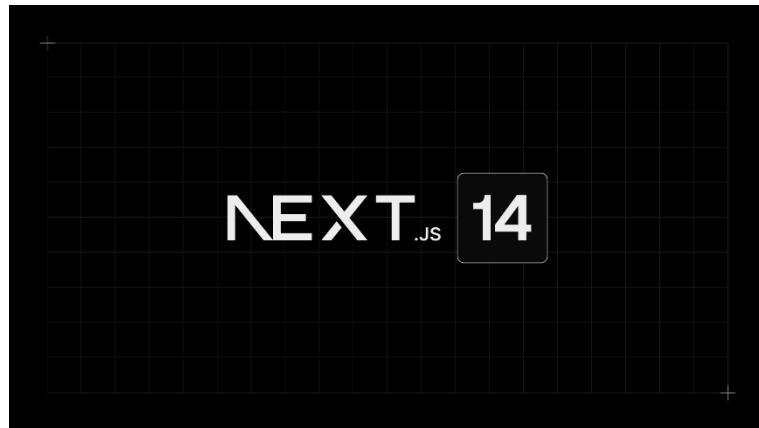


Figure 2.9: Logo of Next.js 14, the latest stable iteration of Next.js as of writing

Because of it being open-sourced since 2016 and backed by pioneering edge cloud platform infrastructure as a service company Vercel, Next.js stands out as a robust React-based full-stack JavaScript meta-framework. The angle that Next.js took in its trajectory to garner its widespread community mindshare is through offering an almost “magical” approach with its comprehensive seamless abstraction of features including supporting every rendering strategy both server-side and client-side (SSR, SSG, ISR, CSR) and support for React Server Components through the adoption of the App Directory and its file-based App Router, making it a strong candidate for the project.

Developers often utilize Next.js for its edge-readiness, despite being known for vendor lock-in (Noor, J.H., 2024) as the process of deploying and ensuring proper middleware functionality elsewhere is often cumbersome, due to being built for configurability for hosting on the Vercel edge network. Regardless of the deployment method, client-side interactivity is achieved through hydration, where a JavaScript bundle is sent to the client. Another contentious feature of Next.js is its caching patterns, which may often lead to unforeseen behaviour between development and production environments. Aside from early adoption of cutting-edge React features like the React Compiler and Partial Prerendering, the Next.js team also helps drive the web development movement forward by providing optimized primitives like `next/image` and `next/font` that ensure the experience of web applications are continuously optimized. Turbopack is the in-house build system tooling developed by Next.js for improved bundling performance and developer experience. All in all, this combination of mature tooling, rapid

adoption of new features, and a thriving ecosystem makes Next.js a versatile choice for modern web development.

2.3.2 Remix

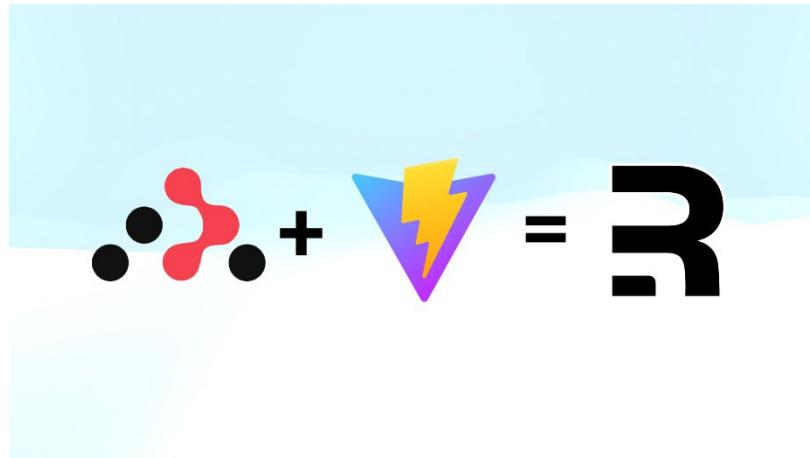


Figure 2.10: In essence, Remix is a framework built upon React Router and Vite (Lybrand, B., 2024)

Remix is another modern advanced web framework built atop React Router as seen in Figure 2.10 to enable a full-fledged framework designed to streamline both server-side and client-side development. It integrates four primary components: a compiler, a server-side HTTP handler, a server framework, and a browser framework (Remix, 2024). This blend of functionalities allows Remix, which placed a substantial bet on Vite, to optimize application performance, improve developer productivity, and enhance user experience.

Strong at the core of Remix is a strict focus on adhering to web standards. Its compiler, invoked via `remix vite:build`, uses Vite to generate key build artefacts that ensure efficient server-side rendering and resource management. These artefacts include a server HTTP handler, a browser build with automatic code splitting and fingerprinted asset imports, and an asset manifest for preloading resources (Remix, 2024). This process allows for optimized application performance and seamless deployment on various JavaScript-hosting platforms, contributing to a smoother user experience.

Another benefit is that the server-agnostic HTTP handler and versatile adapters of Remix provide significant flexibility. Built on the Web Fetch API, Remix can run on numerous environments, including Node.js servers like Vercel, Netlify, and Architect, as well as non-Node.js environments such as Cloudflare Workers and Deno Deploy (Remix, 2024). This

adaptability allows Remix to be progressively adopted in existing JavaScript servers, simplifying integration and offering developers the freedom to choose any preferred deployment platform without being constrained by server-specific limitations.

Remix's server framework combines the responsibilities of the view and controller while leaving the model up to the developers, promoting a UI-focused approach. Route modules handle URL segments, enabling nested layouts in the UI and co-locating UI and controller code for better developer ergonomics. The browser framework's progressive enhancement philosophy allows developers to start with basic HTML forms and scale up to sophisticated interactions seamlessly.

In terms of the future of Remix, the framework's strong vision is to push the ecosystem forward by reducing challenges in the adoption of automatic code splitting, simplified data loading, form actions, server actions, simplified pending states, optimistic UI, server rendering, static pre-rendering and soon, React Server Components (Lybrand, B., 2024). This ensures long-term ecosystem robustness and scalable user experiences that remain resilient under varying network conditions. Overall, Remix's comprehensive integration of server and client functionalities based on web standards makes it a versatile and efficient solution for building high-performance web applications though it is important to take into consideration factors like being battle-tested and mature.

2.3.3 SolidStart



Figure 2.11: SolidStart 1.0 is a JavaScript Framework designed to build SolidJS apps

SolidStart 1.0 is a major release as the de facto JavaScript meta-framework designed to build and deploy SolidJS applications that are an alternative option to React-based frameworks. The

framework offers an array of modularity-focused features that set it apart in the modern web development landscape. Built on Vite 2.0, SolidStart focuses on simplicity, performance, and flexibility, providing a streamlined setup that allows developers to get started quickly without being locked into specific technologies (SolidJS Core, 2024). The framework supports a minimal initial setup with a "Hello World" example weighing in at only ~5kb JavaScript Minified and GZipped. This design principle ensures that as a framework, additional libraries and features can be added "à la carte", avoiding the burden of unnecessary bloat and allowing developers to tailor the setup based on certain needs (SolidJS Core, 2024).

One of the essential features of SolidStart is its advanced routing capabilities. It supports both filesystem-based routing and the flexibility to integrate custom routers. This granular approach allows a more ergonomic definition of routing structures in a way that suits the application's architecture, with options for lazy loading, route groups, named index routes, and more (SolidJS Core, 2024). The introduction of "Server Functions", SolidStart's version of React Server Components, enables seamless server-side operations. Simply adding "use server" to a function ensures that the function is executed on the server therefore supporting advanced serialization methods like Async Iterables, Streams, and Promises. This capability enhances the performance and efficiency of server-side rendering (SSR), static site generation (SSG), and client-side rendering (CSR).

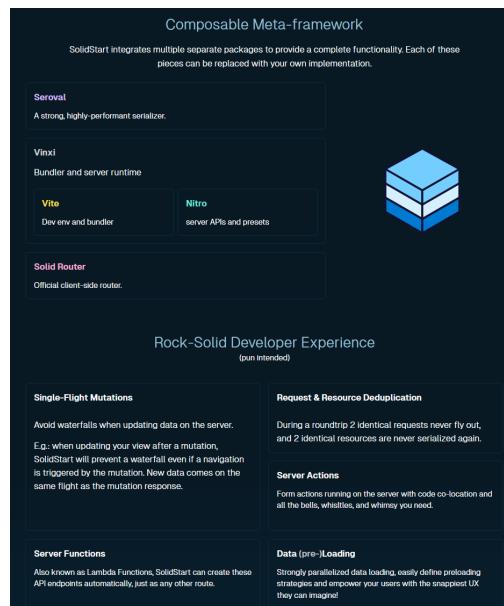


Figure 2.12: SolidStart describes itself as a Composable Meta-framework with Rock-Solid Developer Experience. (SolidStart, 2024)

SolidStart's developer experience is also further enhanced by its integration with a suite of robust open-source battle-tested tools and technologies. It leverages Vinxi for bundling and server runtime, Nitro for server APIs and presets, and Solid Router for client-side routing. Features such as single-flight mutations, request and resource deduplication, server actions, and parallelized data loading contribute to a smoother and more efficient development process. SolidStart's commitment to web standards, combined with its flexibility in deployment across various platforms like Netlify, Cloudflare, Vercel, Bun, Deno, AWS, and Azure, makes it a decent ergonomic and powerful alternative for building high-performance web applications. The release marks a significant milestone in its future, showcasing SolidStart's carved-out niche in the future of web development with its innovative and developer-friendly approach cutting through the bloat of the React ecosystem. SolidStart is the newest technology in the comparison made so far.

2.4 Analysis of Backend-as-a-Service Providers

The following review based on extensive amounts of documentation analysis is to decide the backend provider for the project. The backend provider plays a crucial role in ensuring authentication, storage and database management are handled properly and securely. The backend provider should also support some generative AI features, including vector embedding support. Additionally, the cost and complexity of implementation are crucial factors as the developer resources allocated for the project are small-scale.

1.1.1 Supabase

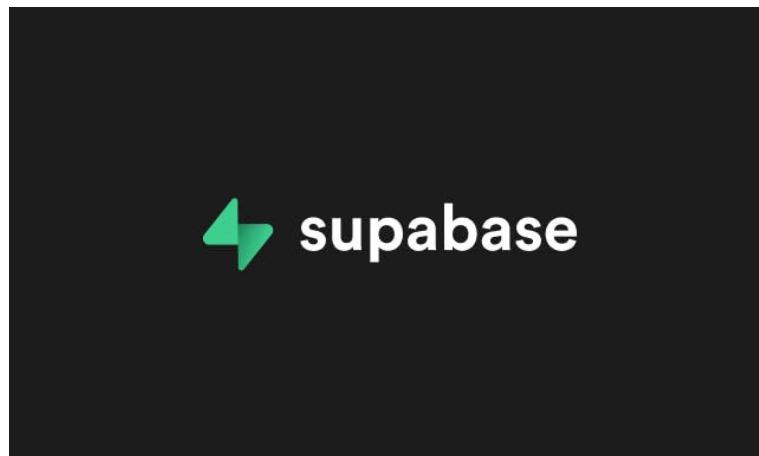


Figure 2.13: Logo of Supabase

Supabase is a rapidly growing backend-as-a-service (BaaS) platform designed to simplify and enhance backend integration. Leveraging open-source technologies as its company ethos, Supabase aims to provide a scalable and user-friendly alternative to Firebase (Supabase, n.d.,). The core of Supabase is open-source and built on PostgreSQL, which offers a rich set of functionalities and scalability differing from those found in NoSQL databases. The choice to build around a battle-tested relational database management service software provider allows developers to utilize the full power of SQL and the extensive ecosystem of tools and extensions available for PostgreSQL. Supabase's commitment to open source extends to its entire stack, ensuring transparency, community support, and the ability to tailor the platform to specific needs.

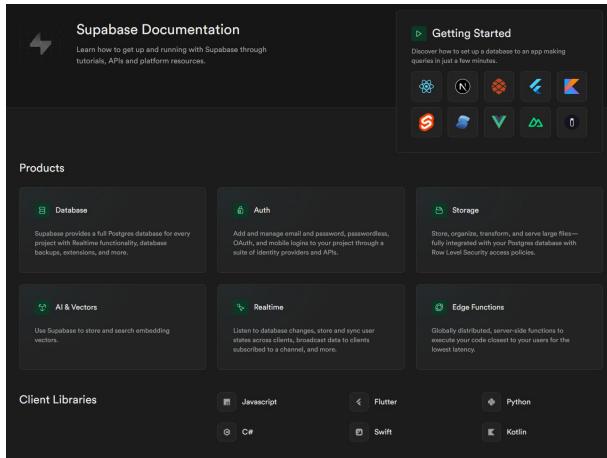


Figure 2.14: Supabase has entered general availability on multiple platforms

The Supabase architecture comprises various integrated yet standalone tools that cater to different aspects of backend development (Supabase, n.d.,). Key components include GoTrue for Supabase Auth, PostgREST for converting PostgreSQL into a local-environment RESTful API, and Realtime, a WebSocket engine for handling live updates and broadcasting changes. Additionally, Supabase includes an S3-standard-compatible storage API for managing large files, a dashboard for database management, and a suite of edge functions powered by Deno. Each of these tools can operate independently or in conjunction with others, providing flexibility in deployment and usage. The integration is seamless and highly community-coveted, enhancing the overall developer experience by reducing complexity and improving productivity through established best practices like row-level security.

Supabase's product principles emphasize scalability, portability, and extensibility (Supabase, n.d.,). The platform is designed to be easily adopted by both indie developers and large-scale

enterprises. Supabase ensures that every component is modular, allowing developers to start with a simple setup and progressively integrate more advanced features as needed. This modularity also facilitates migration in and out of Supabase, avoiding vendor lock-in because local deployment is permitted under open source. By supporting existing standards and fostering an open-source community, Supabase also enhances its ecosystem through extension support, like the pgvector extension for vector storage. This approach ensures that Supabase remains a competitive and forward-thinking choice for backend development with compatibility for major Postgres updates.

2.4.1 Firebase



Figure 2.15: Firebase

Firebase, developed by Google, is an industry-standard comprehensive BaaS platform that provides a wide range of services, including real-time NoSQL databases, cloud storage, authentication, and hosting. Firebase's integration with the Google Cloud Platform (GCP) ensures an expected degree of high performance, scalability, and reliability, making it a popular choice among developers at all small, medium and big enterprise scale.

Besides database solutions like Firebase Realtime Database and Firestore, the newer, more flexible, scalable database, some features of interest are yet to enter general availability as of May 2024, but there is a clear interest by the Firebase and Google Cloud Providers team to provide the backend tooling necessary to kickstart generative AI web integrations, for example, Firebase App Hosting, Firebase GenKit and the Firebase Prompt Structure SDK using GraphQL (Firebase Genkit, 2024). Firebase thus in the future may be also well-equipped for the generative AI era of tooling,

Based on the documentation (Firebase, n.d.), Firebase Authentication simplifies the process of securing user data by providing pre-built UI libraries and supporting various authentication methods including social logins and email/password on various platforms. The platform's cloud

storage solution is designed to store and serve user-generated content, leveraging Google's infrastructure for security and performance. The integration with Firebase Cloud Functions allows for serverless execution of backend code, providing a scalable solution for executing complex logic without managing servers. Firebase's analytics and performance monitoring tools offer insights into user behaviour and application performance, helping developers optimize metrics. The strong ecosystem and extensive documentation make Firebase a staple for many developers seeking a robust BaaS platform supported on many platforms.

2.4.2 AWS Amplify

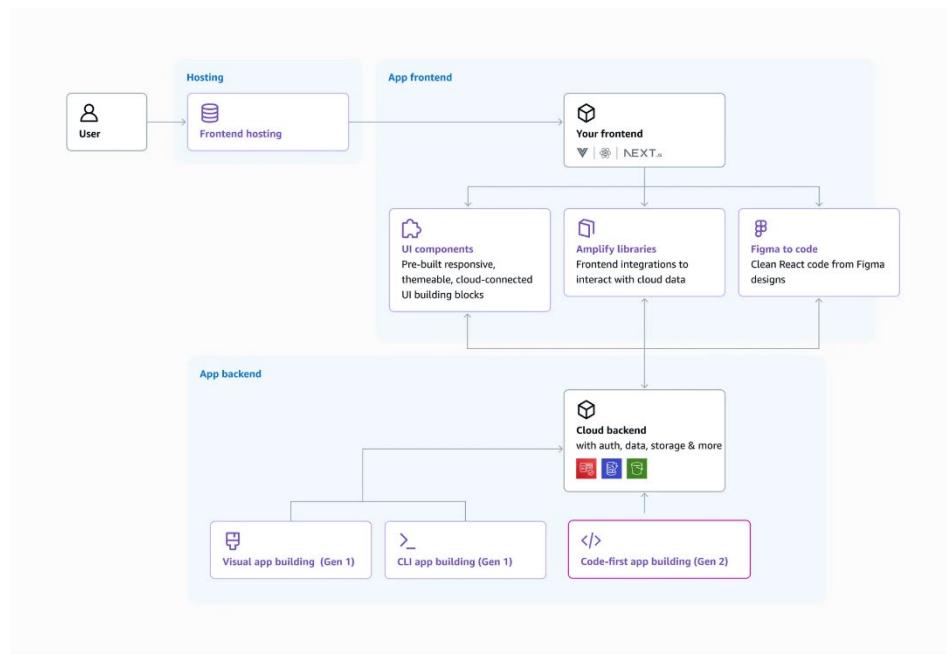


Figure 2.16: AWS Amplify

AWS Amplify, as part of Amazon Web Services (AWS), is a highly versatile and powerful BaaS platform that provides the glue for a comprehensive suite of tools and services for building scalable and secure applications. Amplify both supports and is supported by a wide range of end-to-end, frontend to backend functionalities, including authentication, storage, APIs, and analytics, all seamlessly integrated with the broader AWS ecosystem (Amplify Docs, n.d.).

Amplify's authentication service, powered by Amazon Cognito, supports various authentication methods, including customized email/password, social logins, and enterprise federation, ensuring secure and flexible user management with multi-factor authentication (MFA) (Amplify Docs, n.d.). The platform's storage service leverages Amazon S3, providing

scalable and secure file storage with advanced features like versioning and access control. Amplify also simplifies API creation with AWS AppSync, which offers a fully managed GraphQL service, enabling developers to build real-time and offline-first applications effortlessly.

One of Amplify's standout features is its seamless integration with other AWS services, allowing developers to leverage the full power of the AWS cloud. This includes using AWS Lambda for serverless computing, Amazon DynamoDB for a highly scalable NoSQL database, and Amazon API Gateway for creating, deploying, and managing APIs. Amplify's CLI and admin UI tools streamline the development and deployment process, providing a robust and efficient workflow (Amplify Docs, n.d.). The platform's comprehensive documentation and strong community support further enhance its appeal, making AWS Amplify a versatile and powerful choice for modern web applications.

2.5 Production-scale AI Infrastructure in Large Language Models for Embeddings and Retrieval-Augmented Generation

The selection of the underlying artificial intelligence-related tech stack boils down to factors including model workflow availability, pricing, compute and more. The selection made in this area will down the line account for a sizeable investment of time and resources, but architectural decisions made with portability in mind can ensure a strong and modular foundation under the umbrella of AI products. A few years removed as an emerging technology; generative AI has seen a semblance of maturity in its meteoric rise. Generative AI solutions became an arms race for major big tech players like Google, OpenAI, Anthropic and more in hopes of profiting from providing the shovels in the gold rush. Large language models, specifically those made with the transformer architecture that saw its landmark research established in 2017 have become the cornerstone of many products. Products like embeddings, chat completions, and context windows capabilities of each solution will be summarised and clarified amidst all the chaos and noise.

1.1.1 OpenAI API

OpenAI's API platform is a prominent provider of generative artificial intelligence technologies for both enterprises and small-scale developers allowing the integration of advanced AI capabilities that have taken the world by storm. OpenAI's business model relies on offering access to a variety of large language models built on generative pre-trained

transformers, namely the established GPT series, which are optimized for a range of tasks in multimodalities such as natural language understanding, generation, and more (OpenAI, n.d.). OpenAI's software development kit supports various programming languages and focuses on providing a standardized robust API service that is easily portable between domains.

The screenshot shows a dark-themed interface for the OpenAI Embeddings API documentation. At the top, it says "How to get embeddings". Below that, a text box explains: "To get an embedding, send your text string to the [embeddings API endpoint](#) along with the embedding model name (e.g. `text-embedding-03-small`). The response will contain an embedding (list of floating point numbers), which you can extract, save in a vector database, and use for many different use cases:". Below this is a "Example: Getting embeddings" section with Python code:

```

1 from openai import OpenAI
2 client = OpenAI()
3
4 response = client.embeddings.create(
5     input="Your text string goes here",
6     model="text-embedding-03-small"
7 )
8
9 print(response.data[0].embedding)

```

Below the code, it says "The response will contain the embedding vector along with some additional metadata." Then there is an "Example embedding response" section with JSON code:

```

1 {
2   "object": "list",
3   "data": [
4     {
5       "object": "embedding",
6       "index": 0,
7       "embedding": [
8         -0.000929283495992422,
9         -0.00533642207930928,
10        ... (omitted for spacing)
11        -4.547132266452536e-05,
12        -0.024047505110502243
13      ],
14    }
15  ],
16  "model": "text-embedding-03-small",
17  "usage": {
18    "prompt_tokens": 5,
19    "total_tokens": 5
20  }
21 }

```

Figure 2.17: OpenAI Embeddings API

OpenAI's state-of-the-art models like GPT-4 have been leveraged in highly demanding use cases in enterprise settings, navigating challenges like model alignment and safety. Platforms like OpenAI are built on a pay-as-you-use sales structure, allowing developers to experiment and deploy AI features without needing to worry about infrastructure management including exhaustive compute costs. OpenAI's head start in the shipping of LLM products ensures continuous updates and improvements to the availability of cutting-edge models, offering the latest advancements in human-AI interaction through vision and speech (OpenAI, n.d.). The embeddings API seen in Figure 2.17 suits highly attractive use cases in terms of AI-augmented search capabilities.

The screenshot shows a dark-themed code editor window. At the top, it says "Chat Completions API". Below that, a note states: "Chat models take a list of messages as input and return a model-generated message as output. Although the chat format is designed to make multi-turn conversations easy, it's just as useful for single-turn tasks without any conversation." An example call is shown in Python:

```

1 from openai import OpenAI
2 client = OpenAI()
3
4 response = client.chat.completions.create(
5     model="gpt-3.5-turbo",
6     messages=[
7         {"role": "system", "content": "You are a helpful assistant."},
8         {"role": "user", "content": "Who won the world series in 2020?"},
9         {"role": "assistant", "content": "The Los Angeles Dodgers won the World Series in 2020."}
10        {"role": "user", "content": "Where was it played?"}
11    ]
12 )

```

Figure 2.18: OpenAI Chat Completions API (OpenAI, n.d.)

OpenAI provides alongside its powerful embeddings used in various applications such as semantic search, clustering, and recommendation systems the support for retrieval-augmented Generation (RAG), which enhances the quality of chat responses (pictured in Figure 2.18) by leveraging external data sources retrieved during the generation process. This capability allows for more accurate and contextually relevant outputs, making it suitable for complex applications like customer-facing solutions.

2.5.1 Amazon Bedrock



Figure 2.19: Amazon Bedrock

Amazon Bedrock is Amazon Web Services' offering of a fully managed service that simplifies the integration of highly performant generative AI capabilities into applications. The service provides access to a variety of high-performing foundation models (FMs) from leading AI companies through a single API (Amazon Bedrock, n.d.). Bedrock emphasizes ease of use, customization, and security, making it a robust choice for enterprises. Leveraging its

preexisting infrastructure, Amazon Bedrock is serverless, thus eliminating the need for in-house infrastructure management and ensuring on-demand scalability to handle varying workloads. Developers can quickly deploy AI capabilities into applications using the already familiar and mature AWS services, enhancing effectiveness through generative AI and reducing time to market.

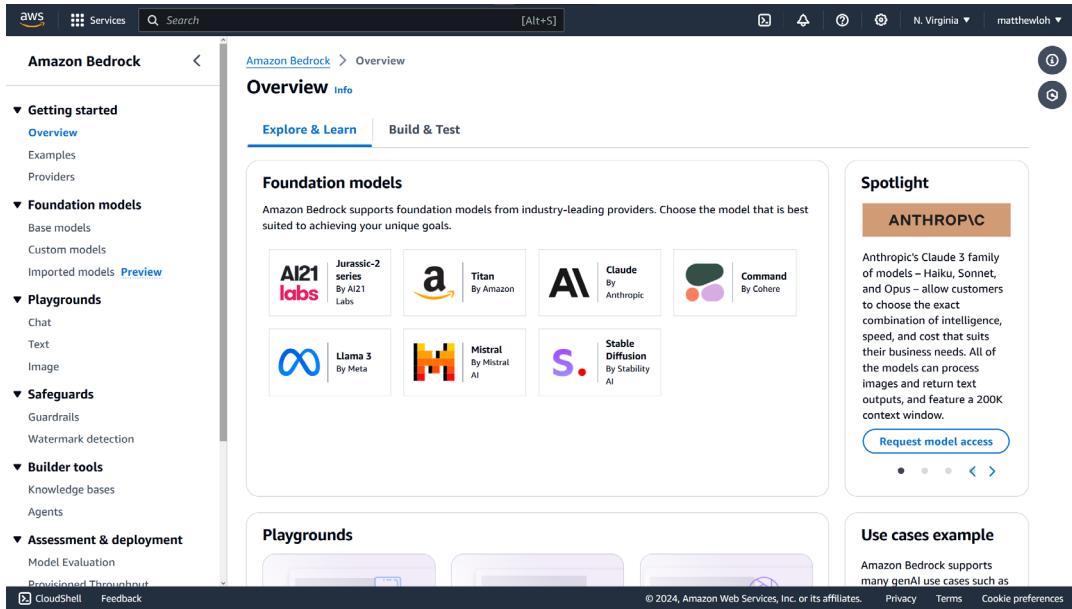


Figure 2.20: AWS Console for Amazon Bedrock and provided Foundational Models

Because of Bedrock's first-hand access to industry-leading generative AI companies such as those seen in Figure 2.20, Bedrock supports a wide range of foundation models that can excel in generating embeddings for tasks such as search, recommendation, and personalization. The platform's developer support for RAG allows teams of developers to seamlessly customize models with in-house data sources and adapt models to specific tasks and domains with said training data (Amazon Bedrock, n.d.). Bedrock provides robust customization options, allowing developers to fine-tune models with proprietary datasets securely. The platform ensures that data used for customization is transferred through Amazon Virtual Private Cloud (VPC) and is not used to train the original base models, maintaining high standards of privacy and security.

Get started with key use cases quickly

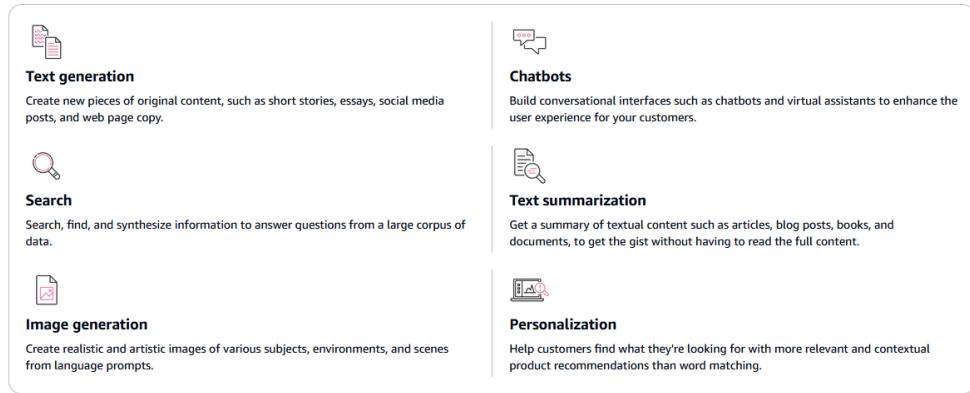


Figure 2.21: Key use cases for Amazon Bedrock including integration into preexisting infrastructure

2.5.2 Open-source Large Language Models as Infrastructure –Locally Hostable Ollama Services

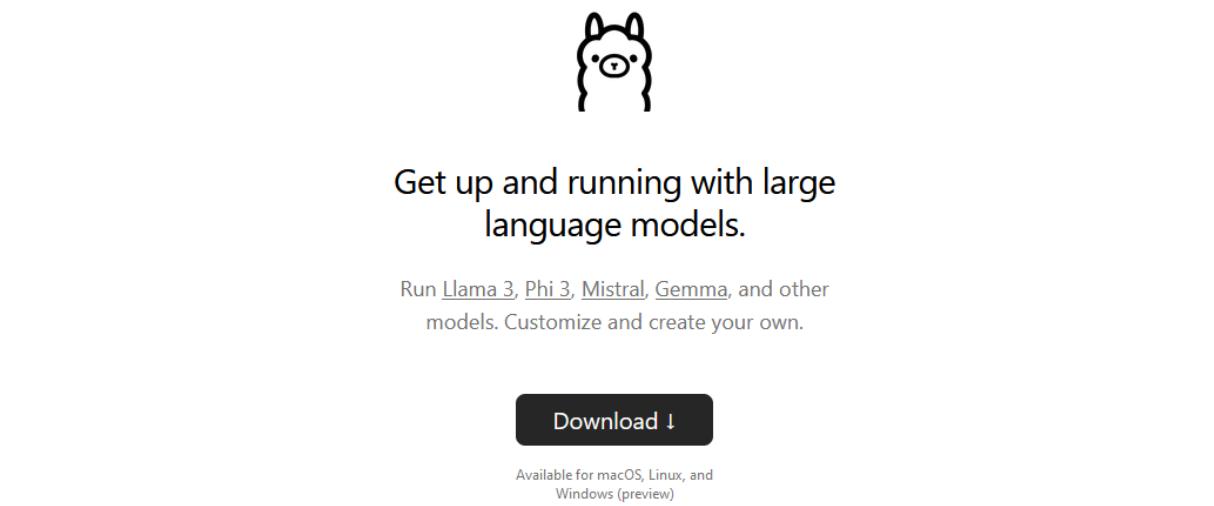


Figure 2.22: Ollama is an open-source project to make running LLMs on local machine seamless

Ollama is an open-source project built around the Llama line of large language models released by Meta. The platform and development tooling, namely the ollama command line interface provided allows locally hostable infrastructure for community-released large language models. These models cater to developers who need robust AI capabilities without relying on cloud-based services. Ollama especially focuses on flexibility, privacy, and control, making it an attractive option for organizations with specific data security requirements (Ollama, n.d.).

Ollama allows for the in-house deployment of AI models on local servers or private clouds, giving developers full control over compute infrastructure. This approach provides flexibility in deployment, making it suitable for environments where data sovereignty and security are paramount or in scenarios where fine-tuning for prompt research is used (Ollama, n.d.). Ollama supports the deployment of open-source large language models that in tandem with the active community carries out the plumbing necessary for models fragmented and fine-tuned for different workflows that may include the ability to generate high-quality embeddings in various applications. The platform's local deployment model and RESTful API endpoint setup enhance local RAG capabilities by allowing direct integration with proprietary data sources, ensuring that the generated outputs are precise and contextually accurate.

Ollama offers extensive customization and support options in terms of configurability, enabling developers to easily pull and finetune models from the centralized store of models (Ollama, n.d.). This ensures that all models are highly portable within the infrastructure, providing the highest level of model velocity, veracity and control. The ability to host models locally also means that developers can implement additional security measures tailored to any specific requirements.

2.6 Infrastructure as a Service for Web and AI Deployments and CI/CD on the Edge

The comprehensive study of Manner, J., (2023) explores the nuanced distinctions and evolving interpretations of Serverless Computing and Function as a Service (FaaS), uncovering the terminological confusion and conceptual overlap within the domain. The work highlights the trend of considering Serverless as an umbrella term encompassing various cloud computing service models. In this section, these insights provide a valuable framework for examining Infrastructure as a Service (IaaS) providers, particularly in the context of edge computing for web and AI inference deployments. The emphasis on operational abstraction and the shift towards NoOps paradigms in the reviewed work emphasizes the critical role of continuous integration and continuous deployment (CI/CD) practices in enhancing the agility and scalability of IaaS solutions on the edge. Thus, various providers will be reviewed from the perspective of frontend and backend cloud providers that are invaluable components of modern software development and its lower barrier to entry for developers.

1.1.1 Vercel



Figure 2.23: Vercel's logo

Innovations in the frontend web territory like React Server Components bring together escalated frontend complexities and challenges. The necessity for a robust, managed, scalable, and automated frontend infrastructure has become paramount to keep up with modern trends of software as a service web application. Vercel, a leading cloud platform provider, dubs its product the “Frontend Cloud”, designed to provide solutions for these challenges to streamline the provisioning of global web infrastructure to its customers. Vercel leads this paradigm shift from traditional hosting to framework-defined infrastructure that enhances developer velocity and experience, allowing teams to focus on innovation and user experience without being bogged down by backend and DevOps constraints. Furthermore, by seamlessly integrating with popular web meta-frameworks like Next.js, also directly supported by Vercel, the infrastructure adapts to each developer team’s specific needs factoring in a broad scope of technical requirements through its Edge Network, Vercel Functions, Edge Middleware, Image Optimization, Incremental Static Optimization, Edge Cache, Data Cache and Cron Jobs.

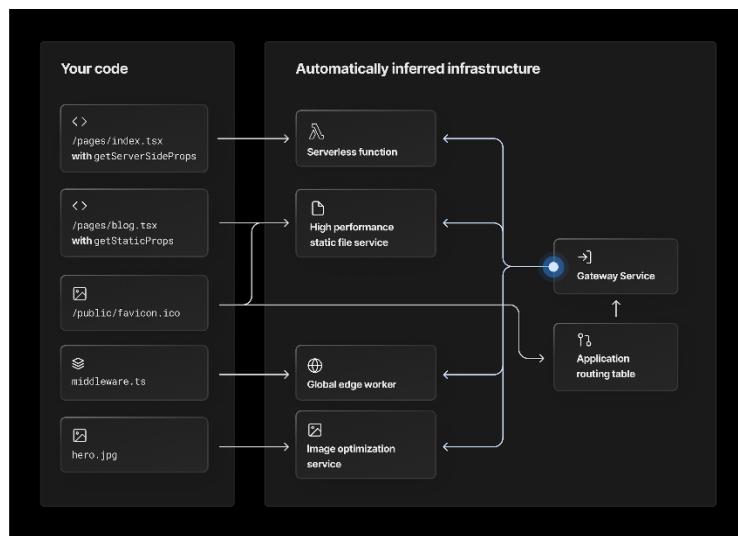


Figure 2.24: Vercel automatically infers project structure to optimize deployments

Vercel’s approach to web development seen in Figure 2.24 is built on the decoupling of frontend and backend deployment architectures, fostering a more agile development environment compared to a monolithic structure whilst still bringing in monitoring and real-

time observability features. This decoupling enables independent scaling, faster iteration cycles, and simplified team collaboration, regardless of developer team sizes. Vercel's serverless architecture is the bedrock for this flexibility, automatically managing resource allocation and scaling based on real-time demand. This serverless model eliminates the overhead associated with traditional server management, providing a streamlined and cost-effective solution that is elastic - being able to scale down and up to match demands. The automatic continuous integration and continuous delivery infrastructure features ensure that even a single developer team can utilize deployments optimized for speed and reliability. Beyond the technical efficiencies, Vercel's overall platform offers a cohesive, user-friendly environment for deploying global applications that adhere to global compliances.

2.6.1 Render



Figure 2.25: Render's logo

Render is a highly versatile unified cloud platform as a service designed to host various web full-stack solutions and services, including web applications, APIs, and static sites. It provides a streamlined experience for deploying and managing backend services, particularly for Python web frameworks ranging from micro-frameworks like FastAPI to full-fledged backend-focused frameworks like Flask and Django. Render offers intuitive documentation on the ins and outs of the deployment process, supporting a wide range of programming languages and frameworks. For such developer requirements as Python functions, developers can deploy the code directly from Git repositories, with Render automatically handling the build and deployment processes using Render's infrastructure-as-code model. Render's autoscaling of instances feature ensures that applications can handle increased loads efficiently, scaling up and down based on traffic.

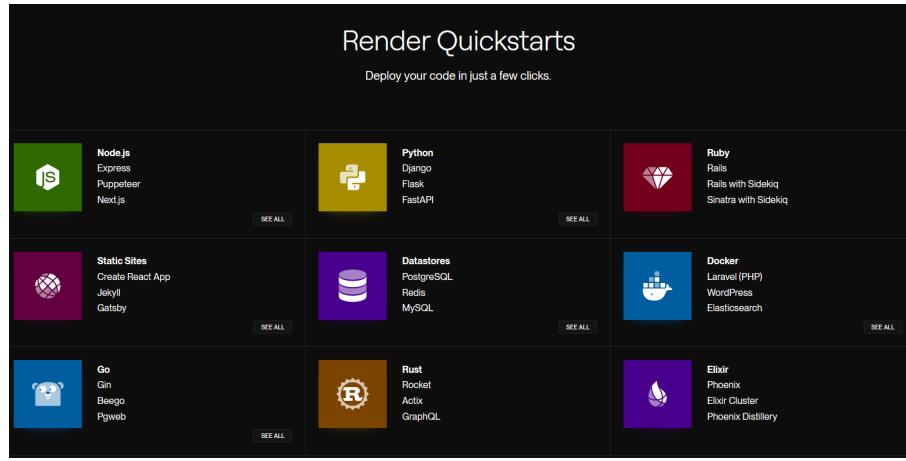


Figure 2.26: Render's built-in support for various technologies

Render optimizes backend performance by providing services seen in Figure 2.26, including managed databases, caching, and background job processing. Its infrastructure is designed to deliver low-latency responses, making it suitable for API services and real-time applications. Render also emphasizes confidence in reliability, providing DDoS protection and compliance with data privacy policies. Therefore, Render is a decent solution for learning toward general-purpose and backend-heavy projects that can serve workflows requiring its always-on services.

2.6.2 Amazon Web Services

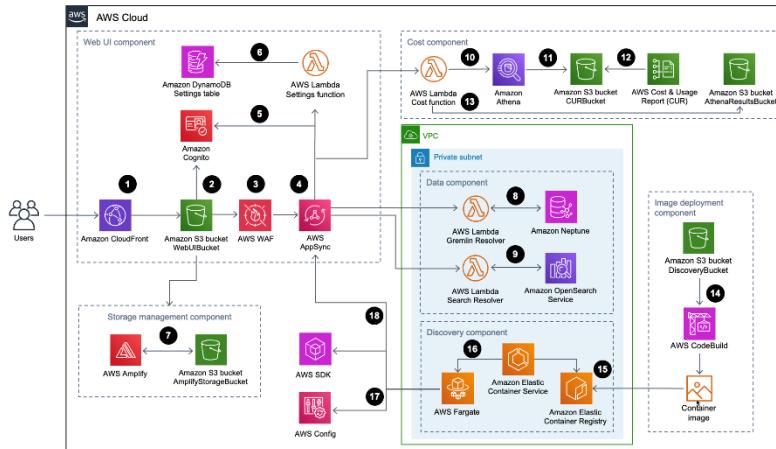


Figure 2.27: Amazon Web Services Example Cloud Service Architecture

Amazon Web Services (AWS) is the de facto cloud platform service offering a vast assortment of managed services for enterprise-ready computing, storage, databases, machine learning, and more. AWS is synonymous with providing a highly scalable and flexible infrastructure that can support both frontend and backend components of web applications. AWS particularly offers

a wide range of deployment options, from EC2 virtual servers on the cloud to serverless computing with AWS Lambda. Developers can deploy applications using AWS Elastic Beanstalk, which automates the provisioning and scaling of web application resources (Amazon Web Services, n.d.). Elastic Beanstalk is an end-to-end solution for auto-scaling groups and load balancers ensure that applications can handle varying levels of traffic, scaling up or down as needed.

AWS provides tools and services to optimize the performance of web applications, such as content-delivery networks, managed databases, and caching. Its global infrastructure can pose an immense challenge for developer experience and resource allocation. Being locked into the AWS comes with its various drawbacks and benefits. For one, properly utilizing AWS is challenging, but it allows low-latency access for users worldwide backed by the vast amounts of resources poured into the underlying cloud computing infrastructure (Amazon Web Services, n.d.), allowing advanced monitoring and logging with certain services. AWS might not be suitable for small teams of developers if needing various specific workflows to track application performance and identify issues. Nevertheless, AWS offers a comprehensive suite of tools for developers, including the AWS Management Console, CLI, SDKs, and extensive documentation (Amazon Web Services, n.d.). It is as versatile of a cloud platform as it comes, providing software development kits for a wide range of programming languages and frameworks, making it versatile for nearly every use case imaginable. AWS's integration with other AWS services facilitates the development of complex applications with minimal operational but extreme developer velocity and experience overhead.

2.7 Proposed Work

An analysis of the technologies is undertaken crucially to understand the context within which a full-stack digital nomad onboarding support platform could be composed. The full-stack nature of the entailing project covers various branches of development, from the frontend to the backend, such as determining on a microservices-oriented architecture versus a monolithic structure. To make sense of the layers of complexity, each technology should have a clear responsibility. To convey this objectively, the following section will provide context behind the selection of key technologies based on the literature review carried out.

1.1.1 Retrieval Augmented Generation System Architecture

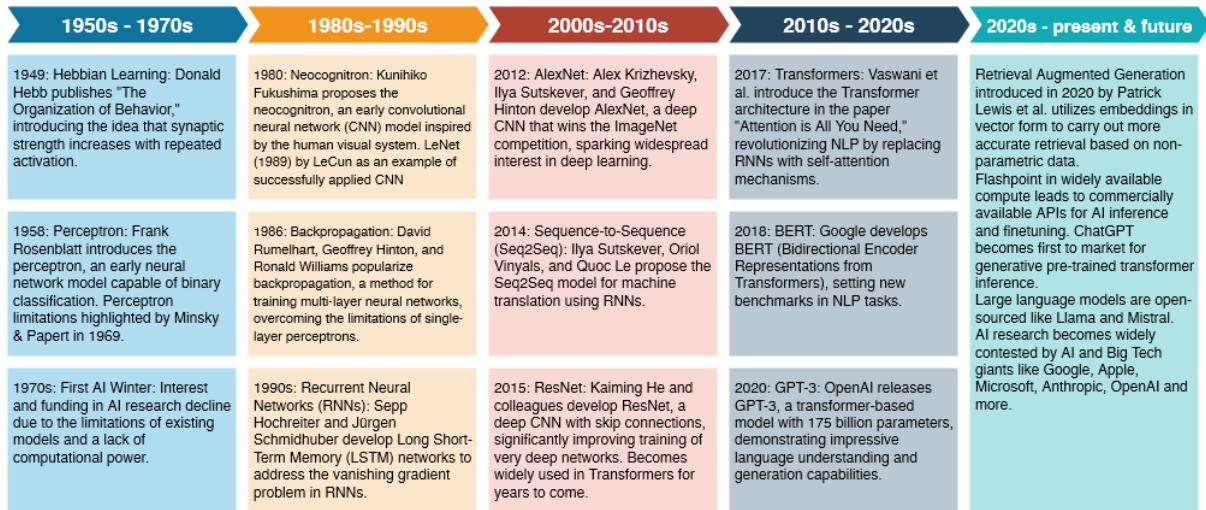


Figure 2.28: A historical analysis of the evolution of early neural networks to the current transformer lightning-in-a-bottle landscape

A crucial component for the success of the project is the due diligence to preface the proposed work through the analysis of historically significant milestones in neural network development. The evolution the field experienced shapes the current iteration of artificial intelligence-based technologies. Figure 2.28 is partially constructed through the annotated work of Schmidhuber, 2022, which breaks down the pioneering works of leading AI researchers. The evolution of neural networks into its current transformer-based iteration was a gradual and transformative journey.

The journey started in the 1940s with Hebbian learning where the foundation for understanding synaptic strength was laid. Then, the 1950s saw the development of the perceptron by Frank Rosenblatt, but its limitations highlighted by Minsky and Papert in 1969 marked something of a start to the first AI winter in the 1970s as the compute available was limited. The 1980s saw rumblings in the field with the backpropagation algorithm introduced by Rumelhart, Hinton, and Williams, enabling the training of multi-layer neural networks. The conception of the convolutional neural networks (CNNs) by Fukushima and later advancements like LeNet by LeCun in the late 1980s demonstrated the potential of deep learning for image recognition. Slowly but surely, as hardware began to develop with Moore's law, the 1990s and early 2000s saw the development of recurrent neural networks (RNNs) and LSTMs by Hochreiter and Schmidhuber, addressing the vanishing gradient problem, which is a major roadblock in the training process of the neural networks.

Branching into modern times, one real breakthrough was seen in 2012 with the success of AlexNet, which leveraged GPUs for deep learning. This set the stage for further innovations, including the introduction of sequence-to-sequence models and ResNet which are now widely used in modern transformer architecture. The hierarchical structure of neural network primitives led each fundamental concept to snowball into the turning point of 2017, where the landmark paper by Vaswani et al. gave rise to a tangibly linear growth of effectiveness of these NLP given the compute invested by utilizing self-attention mechanisms. The writing was on the wall when Google then pioneered BERT while OpenAI, when the company was still primarily an AI research organization, developed the seminal concept of generative pre-trained transformers (GPT). The power of transformers now spearheads the almost unavoidable integration in various applications.

Now in the tumultuous 2020s, entire industries are at a flashpoint of innovation and widespread availability. The landmark work of Lewis et. al., (2020) which coined the term retrieval-augmented generation (RAG) spurred on a branch of state-of-the-art large language model-based applications that can harness pre-existing data to improve context relevance and accuracy. In the grand scheme of the project, efforts are primarily made to condense and implement the concepts from an application development standpoint. The constraints of the project are such that it leverages abstractions from the underlying rigorous linear algebra and theoretical fundamentals based on the open work of the active research and development communities. This is an important understanding to possess to enable the project to feasibly explore the capabilities of large language models as endless amounts of methodologies continue to be invented at a rate of knots. The technologies used in the project like OpenAI's API and the open-source Llama models by Meta AI for utilities like LLM inference, embedding generation and chat completions are proof of the new era of availability of AI.

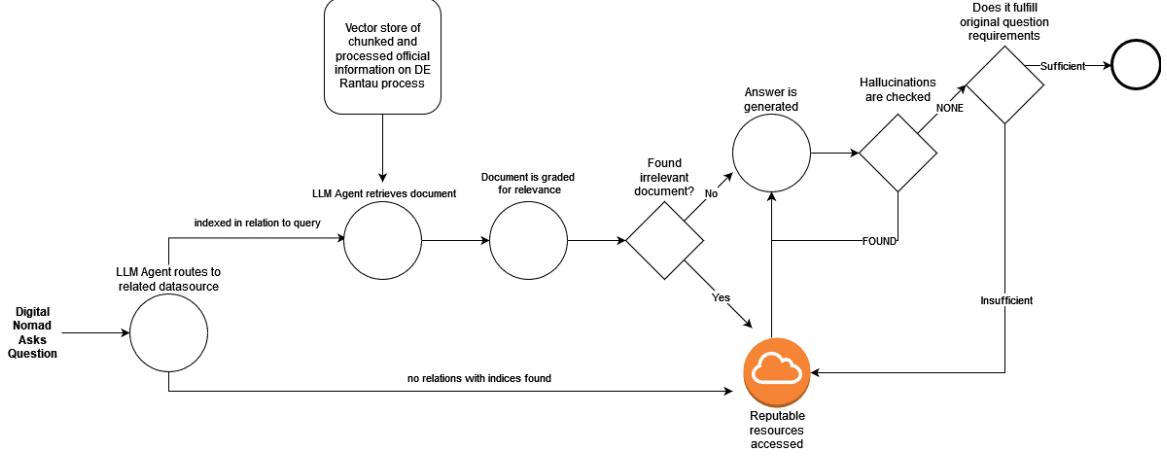


Figure 2.29: Proposed RAG Architecture

Therefore, the modelling of the process flow as seen in Figure 2.29 for the self-reflective Retrieval-Augmented Generation (RAG) is implemented based on LangGraph resources provided by LangChain, (n.d). The underlying process implements several key stages to enhance the quality and relevance of generated content. Initially, a user query is embedded into the system, triggering a retrieval phase where relevant documents are fetched from a variety of sources (Jeong et al., 2024). These documents undergo an evaluation process; if the quality is deemed insufficient, the system iteratively rewrites the query and retrieves new documents until satisfactory content is obtained. In this section, safety and security measures are researched to ensure that the data handled in the application is treated with best practices like salting, hashing and other cryptographic methods learned to ensure proper authentication and authorization behind sensitive document handling. Additionally, row-level policies help in the backend layer to secure database access.

LangGraph's graph architecture facilitates this iterative process, managing transitions between retrieval, document grading, query rewriting, and generation stages. For instance, in the Corrective RAG (CRAG) implementation, a lightweight evaluator assesses the retrieved documents' quality, supplemented by web-based retrieval if necessary (Yan et al., 2024). Query rewriting optimizes search queries, while additional prompting tools enable decision modelling for conditional logic. This method ensures that the final generation is grounded in the most relevant and accurate context available.

Overall, LangGraph's flexible architecture supports the development of complex RAG flows, as demonstrated by implementations inspired by recent papers like Self-RAG and CRAG (Asai et al., 2023; Yan et al., 2024). This modern prototype, “self-reflective” RAG showcases the

importance of iterative feedback loops and adaptive decision-making based on a graph structure in enhancing the capabilities of large language models to generate hopefully more accurate and contextually relevant content.

2.7.1 Overall Frontend and Backend Technology Selection

In selecting the technology stack for the project, Next.js 14 is opted for as the JavaScript meta-framework for several key reasons. Firstly, Next.js is a battle-tested framework proven to operate on a blend of performance, ease of use, and flexibility. These terms are given because the framework offers built-in support for different rendering paradigms such as server-side rendering (SSR) and static site generation (SSG). The framework ensures a balance of fast initial page loads and also improved SEO, crucial for delivering a responsive user experience and greater user visibility. Moreover, Next.js is also a pioneer in the thriving JavaScript & TypeScript library ecosystem, providing excellent documentation for integrating with popular technologies like authentication libraries, ORMs and AI SDKs, allowing it to be scaled and maintained over time. Additionally, for beginners, Next.js has a less steep learning curve because of its vast amounts of tutorials due to its maturity in the full-stack web framework world. The choice also considers the outstanding developer communication by the team behind Next.js in bug fixes and features that are quickly released in backwards-compatible versions.

For the backend, Supabase is chosen due to its easy integration and full-stack capabilities, especially on the web. Supabase leverages PostgreSQL under the hood for its database layer and a plethora of community-developed extensions for features like user authentication. SQL is at its core offering, allowing both full control and seamless integration with web frameworks with its JavaScript client that provides sane abstractions. Supabase works on best practices, enabling rapid development without compromising on security with features like Row Level Security. With Supabase, the backend logic has a simple monolithic workflow that scales as the application logic layer continues to develop, requiring less time on infrastructure management.

To power the intelligent features of the application, the proposed project aims to integrate the OpenAI API and Ollama into the tech stack. OpenAI's powerful language models like GPT-4 and GPT-4o provide an easy-to-implement API, such as for chatbots and embedding generation. Choosing to go down the paid pathway enhances the interactivity and utility of the application because compute is handled by OpenAI. Ollama, on the other hand, can provide a

rapid to iterate command line interface to serve up locally hosted REST APIs, essential for prototyping using local resources. These integrations allow an architecture of modern AI technologies to take shape without investing heavily in compute resources.

Finally, Vercel is a potential platform to seamlessly deploy and host the frontend of the application. Vercel's first-party integration with Next.js greatly simplifies the deployment process, offering instant deployments on the edge with CI/CD, serverless functions, and global CDN distribution out of the box. This ensures that the application is not only performant but also scalable, with minimal operational overhead assuming the database and deployments are within vicinity of each other. By choosing Vercel, development can focus on delivering features and improving user experience because the application is hosted on a reliable and efficient platform.

2.7.2 High-level Overview of Technology Integration

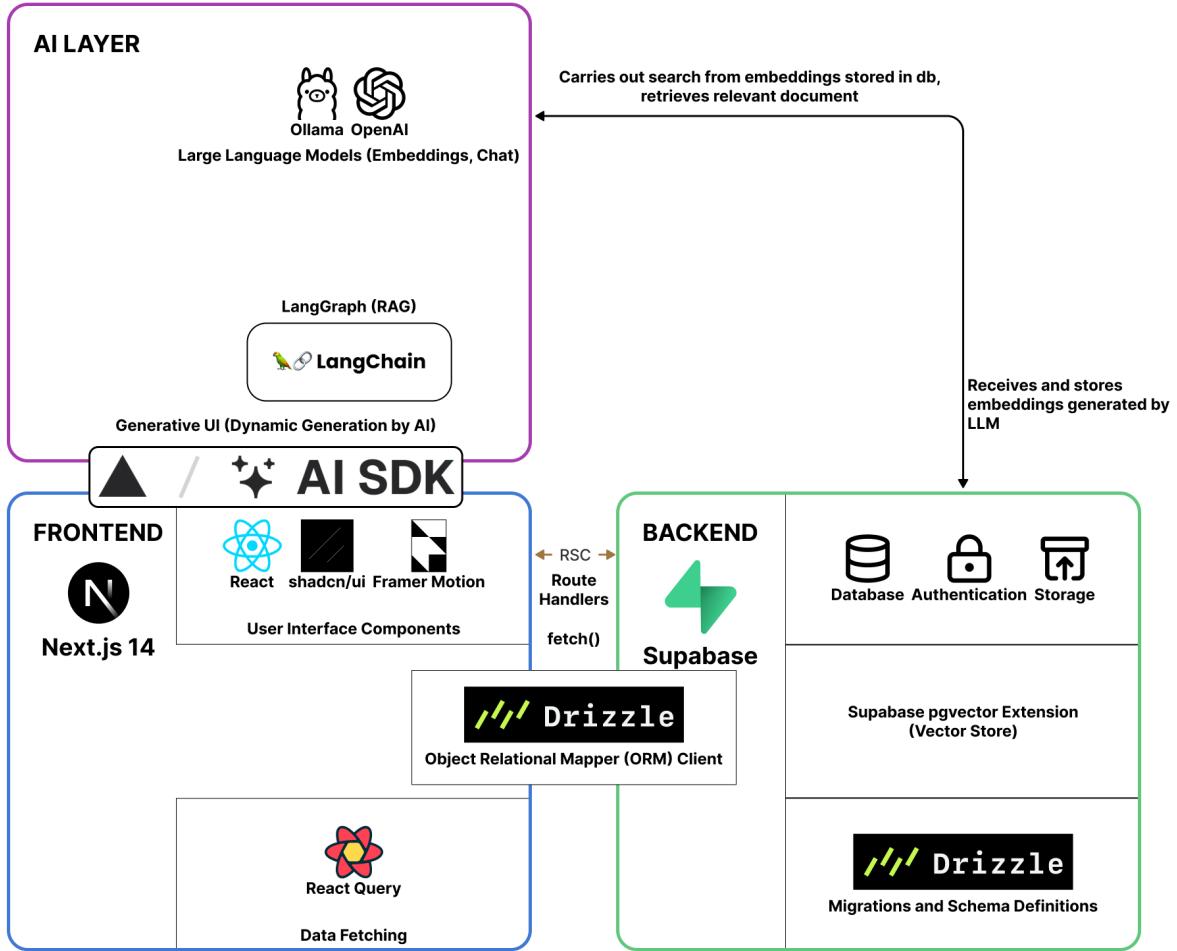


Figure 2.30: Proposed Project Implementation Architecture

Figure 2.30 is a high-level overview of the proposed project. In the frontend layer, Next.js 14 serves as the JavaScript meta-framework. The frontend UI/UX experience is complimented by a comprehensive suite of battle-tested libraries for building smooth web applications, including React, shadcn/ui and Framer Motion. These libraries are highly optimized to ensure responsiveness and smooth animations. This is crucial because initial load times due to compute overhead may occur. Next.js also supports React Server Components, which facilitate the rendering of UI components on the server, ensuring a smooth and responsive client-side user experience. Additionally, route handlers and the `fetch()` API are to be employed where necessary to manage client-server communication, allowing best practices for data retrieval and mutation. React Query is a library potentially to be used to ensure caching is solved for long queries to ensure data received on the frontend is up-to-date.

For the backend layer, Supabase is selected as the backend provider. Leveraging PostgreSQL, Supabase offers a robust and scalable database solution with built-in authentication, storage, real-time support, edge functions and pgvector support. Right out of the box, Supabase's CLI and JavaScript SDK ensure synchronization across clients, crucial for applications requiring a cohesive backend. The backend stores embeddings generated by the AI layer from the requests made client-side and manages vector stores in the database, facilitating efficient retrieval and storage of data for AI operations like semantic search.

In the AI layer, Ollama and OpenAI are integrated as the primary providers of large language models. Ollama supports local development and rapid API setup, while OpenAI's LLMs enable offloaded compute, such as rapid RAG-based chat conversations and embedding generation. The AI layer communicates with the backend through embedding searches from vector stores in the database, optimizing data retrieval and analysis for RAG. To interface with the frontend, the Vercel AI SDK is utilized, providing UI functions that allow the LLMs to directly return components based on the workflow. This integration enhances the interactivity and functionality of the application, supporting complex AI-driven experiences seamlessly across the entire tech stack. This layered approach ensures that the application is performant and scalable. The tech stack leverages cutting-edge AI technologies to deliver a robust and interactive user experience. By utilizing Next.js, Supabase, Ollama, OpenAI, and Vercel AI SDK plus additional libraries, the project achieves a well-rounded integration of frontend, backend, and AI capabilities.

2.8 Conclusion

In conclusion, the proposed project architecture integrates a well-researched blend of modern technologies to create the fundamental tech stack for a digital nomad onboarding support platform with community support. By leveraging Next.js for frontend development, Supabase for backend services, and OpenAI and Ollama for advanced AI capabilities, the application ensures a clear definition of the concerns and responsibilities of the technologies used. Vercel potentially serves as a reliable deployment platform, enabling instant and global distribution of the application, but this comprehensive tech stack is tied to development efficiency and capabilities. Therefore, through careful consideration of the integration of these technologies, the project aims to deliver on the essentials first, catering to basic user needs with cutting-edge features and optimal performance.

CHAPTER 3: RESEARCH METHODOLOGY

3.1 Introduction

This chapter includes details on the overall implementation of the project's research methodology, which is the Prototyping Model, a type of Software Development Life Cycle (SDLC) model. The Prototyping Model is selected for its iterative nature considering the project's development which is to be completed over two formal semesters. Factoring in the nature of which the technologies are rapidly evolving, the research methodology used requires that the definitions of software requirements clearly follow the Prototype Model that aids in adapting without a clear direction and rather expediting the development process to deepen the stakeholder understanding through rapid iterating of application versions. Therefore, the following sections will go into detail about an analysis of the questionnaire, project management tools including a Gantt chart and Kanban board of the project, development tools used, version control strategy, testing methodologies following paradigms including functional and non-functional testing, and most importantly, that supports the problem statement and idea development and the overall system flowcharts in the form of activity diagrams.

3.2 Prototyping Model

Prototyping is a valuable process model in software development, following the Prototyping paradigm seen in Figure 3.1, which is especially useful when general objectives are defined but detailed requirements are unclear. It allows for a better understanding and refinement of software requirements through iterative feedback and improvement. While prototyping can be a standalone model, it is often integrated within other process models to address specific needs or uncertainties (Roger & Bruce, 2015). This methodology applies to this project due to the underlying technology's reliance on bleeding-edge innovation in the AI and LLM environment. Furthermore, the DE Rantau program's stakeholder requirements may also grow in parallel with the development process. With these factors taken into consideration, a model that adapts through prototype iteration focuses on reducing friction between stakeholder interests and development speed. Essentially, it makes it the research methodology of choice going forward for the development of the project.

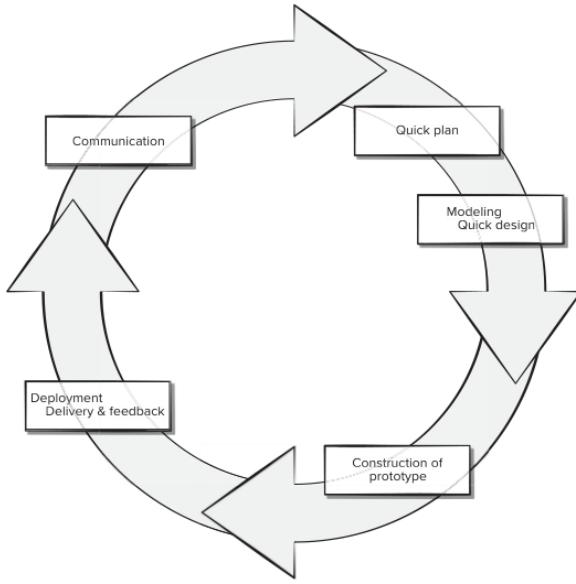


Figure 3.1: The Prototyping Paradigm (Roger & Bruce, 2015)

3.2.1 Communication / Requirements Gathering

Effective communication is the foundation of successful software development in the Prototype Model. Following the prototyping paradigm, initial communication involves gathering high-level requirements from stakeholders, including digital nomads, local businesses, and government entities involved in the DE Rantau program. This phase aims to define the overall objectives, identify key features, and understand the pain points and needs of various stakeholders. Throughout the course of this project, a questionnaire was carried out that targeted digital nomads familiar with the DE Rantau program to elicit qualitative and quantitative data to simultaneously help shape the problem statement and features to be delivered after gauging the response towards the preexisting solutions in the digital nomad environment. Efforts are continuously made to attain correspondence with the relevant official government authorities to keep the system grounded within practical reality and resource constraints.

The act of regular communication is maintained throughout the development process to ensure that all stakeholders are aligned and any changes in requirements are promptly addressed. Potentially for the digital nomad onboarding support system, this involves continuous engagement with users to gather feedback on prototypes using different models that can be integrated with different modalities supported. Therefore, this phase discusses potential improvements and ensures that the system evolves in a direction that suits potential needs.

Leveraging some principles from the Spiral Model methodology, this phase is crucial to help mitigate risks associated with misaligned expectations and changing requirements, ensuring a stakeholder risk-accommodating strategy.

3.2.1(a) Questionnaire Analysis

The following questionnaire was carried out anonymously in accordance with the expected academic standards. This questionnaire was permitted out of the goodwill of the administrators of the Facebook group Malaysia's DE Rantau Digital Nomad Member Community. The questionnaire's target audience consisted of digital nomads interested in applying for the pass, digital nomads who were approved and in the pre-arrival portion of the process and also digital nomads who already entered Malaysia under the DE Rantau Nomad Pass or had already completed the allotted duration of stay. The questionnaire was carefully crafted with essential instruments like Likert scales, multiple and single choice and open-ended questions to gauge the difficulties and pain points of the processes that go into researching and applying for the DE Rantau Nomad Pass program. The questionnaire was carried out using Google Forms. The questionnaire's full contents with questions will be attached appropriately in the appendix as Appendix B. The questionnaire will be continuously evaluated as responses continue to be addressed.

Firstly, which of these statements best describes your status currently as a digital nomad?

6 responses

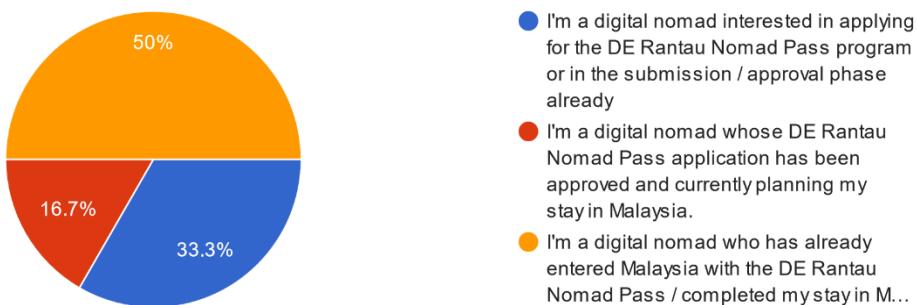


Figure 3.2: Percentage of respondents based on category of applicant / digital nomad

Based on Figure 3.2, half of the respondents were digital nomads who had already entered the country or completed the allotted stay while the remainder were digital nomads who were

interested in applying or were yet to enter the country post-approval. Based on this, the digital nomads are streamed based on the category to differentiate one another.



Figure 3.3: Demographic Questions (Pre-application & Pre-arrival Digital Nomads)

Based on Figure 3.3, the respondents were from an age group of 25-44 years old, consisting of Canadian and Pakistani nationalities of the same countries of residence. These respondents were highly educated, possessing master's degrees and working in information technology and digital marketing/content creation. The digital nomads also consisted of both the new and

highly experienced (5+ years of experience). Finally, the respondents also show a resounding likelihood to travel to destinations along with dependents including spouses, children and/or parents.

In terms of personal importance, rank the following factors that motivates you to pursue a digital nomad lifestyle in certain countries.

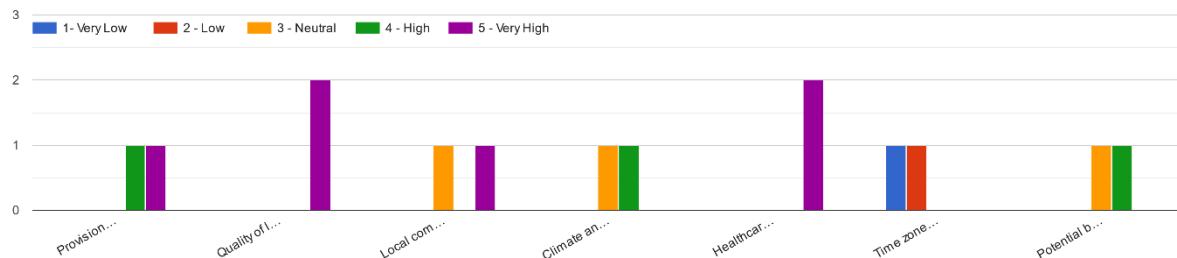


Figure 3.4: Motivation to Pursue DN Lifestyle (Pre-application & Pre-arrival Digital Nomads)

Based on Figure 3.4, the factors that gave the respondents the highest motivation to pursue a DN lifestyle were the quality-of-life improvements found abroad relative to local spending power and the healthcare, medical facilities and services factor. Additionally, this category of respondents also ranks the provision of government-issued digital nomad-tailored visas and potential business opportunities of equal if not just slightly less importance.

What methods do you typically use to search for information related to digital nomad visa programs? Select one or more that apply.

2 responses

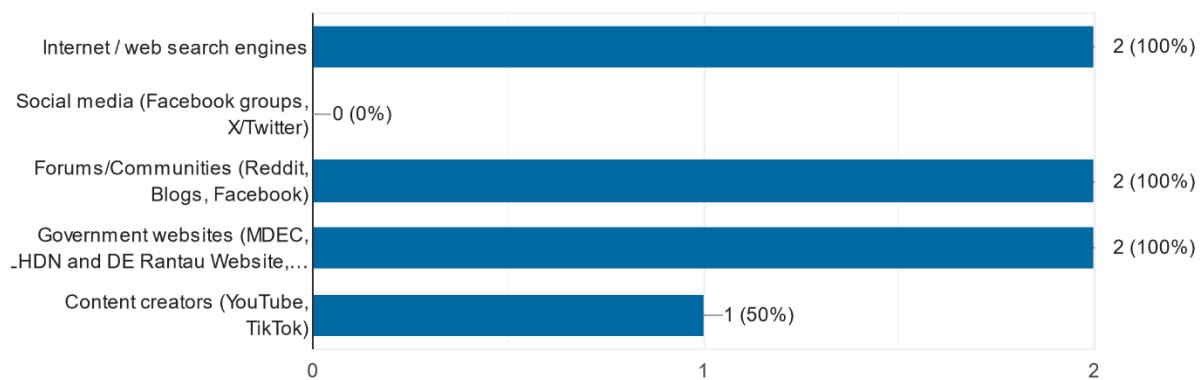


Figure 3.5: Information Gathering Methods (Pre-application & Pre-arrival Digital Nomads)

In Figure 3.5, the digital nomads primarily rely on web search engines, online communities, and government websites for information, with lesser use of content creators and no use of social media platforms like Facebook groups or Twitter.

Based on your experience, indicate the degree of perceived difficulty you faced in the following aspects while researching the DE Rantau Nomad Pass Program:

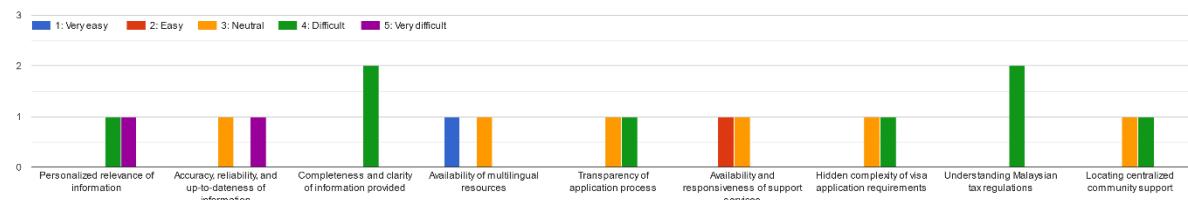


Figure 3.6: Degree of Difficulty Faced in Various Aspects of Researching (Pre-application & Pre-arrival Digital Nomads)

In Figure 3.6, the categories that reported a "Very difficult" were in terms of personalization, accuracy, reliability and up-to-dateness of information. Two other aspects that stood out in terms of difficulty were the completeness and clarity of information, especially concerning understanding Malaysian tax regulations. In general, the perceived experience is skewed towards difficulty in various aspects of research.

Based on the following categories of information, rank the difficulty experienced carrying out research and finding support applicable to you

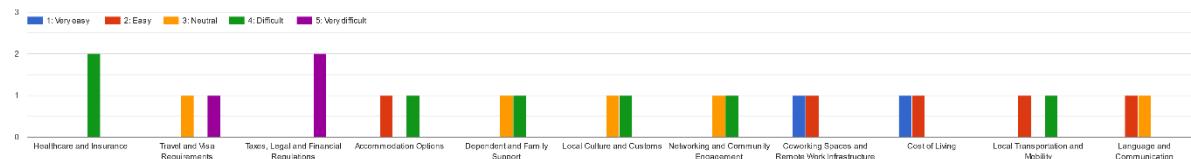
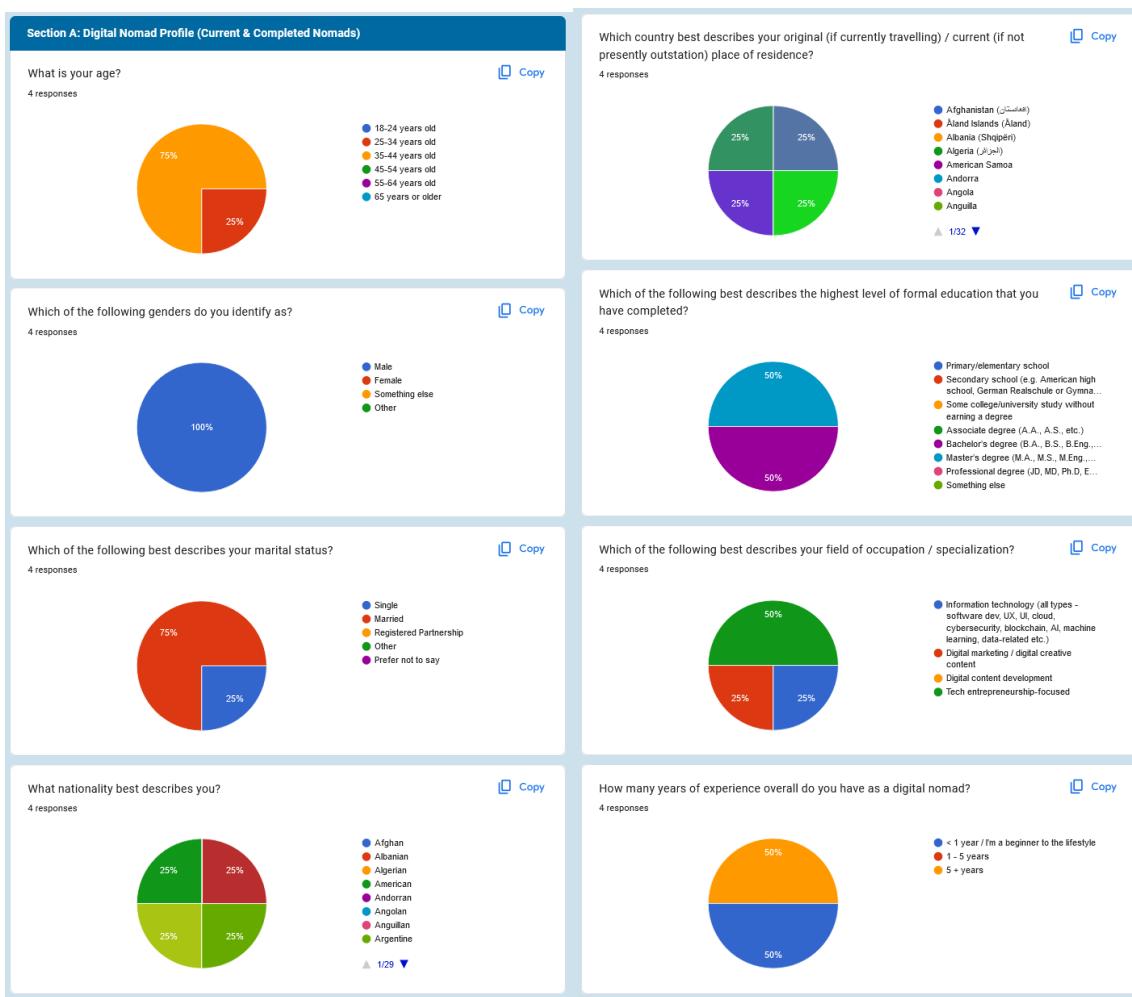


Figure 3.7: Degree of Difficulty Faced in Various Categories of Information (Pre-application & Pre-arrival Digital Nomads)

Based on Figure 3.7, the categories with the most significant challenges are taxes, legal, and financial regulations, with the respondents finding it "Very difficult," while categories like cost of living and coworking space / remote work infrastructure were easier. Healthcare and insurance stand out too as an outlier of difficulty. Other notable categories were generally rated as neutral or easy, indicating a mixed level of difficulty across various information types.



How would you describe the likelihood for you to travel and live abroad in a destination with dependents (spouse, children, parents)?

4 responses

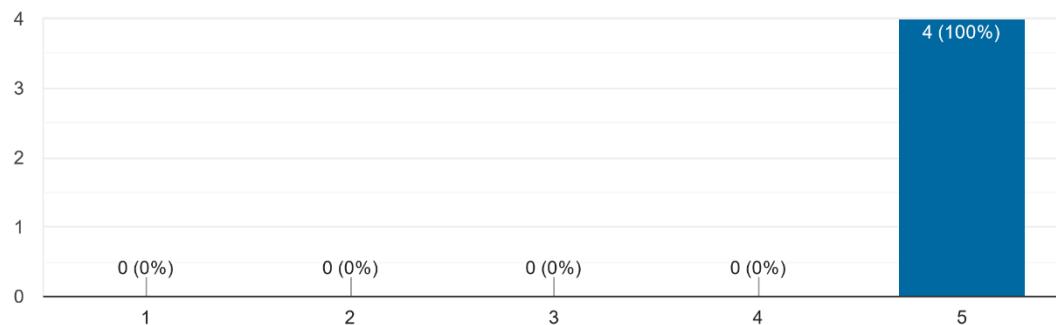


Figure 3.8: Demographic Information (Current & Completed Digital Nomads)

According to Figure 3.8, which compiles the entire demographic information of currently active and completed DNs in Malaysia under the DE Rantau Nomad Pass visa, the majority of

these digital nomads currently in Malaysia are males aged between 25-34 years, representing 75% of the respondents, with the remaining 25% aged between 35-44 years. All respondents identify as male. Regarding marital status, 75% are single, while 25% are in committed partnerships. Nationality-wise, the respondents are evenly split among Polish, Argentine, British, and Indian, each comprising 25%. In terms of original and/or current place of residence, there is an equal distribution with respondents coming from Poland, Singapore, the United Kingdom, and the United Arab Emirates. Educationally, the respondents display a wide range, with 50% having completed a bachelor's degree and the remaining holding a master's degree. Professionally, a quarter of these respondents specialize in tech entrepreneurship, while the rest are divided between digital marketing / creative content and information technology roles, each representing 25%. In terms of experience as digital nomads, 50% have 5+ years of experience, and 50% are just beginning. Additionally, all respondents express a high likelihood (rating 5 out of 5) of travelling and living abroad with dependents.

In terms of personal importance, rank the following factors that motivates you to pursue a digital nomad lifestyle in certain countries.

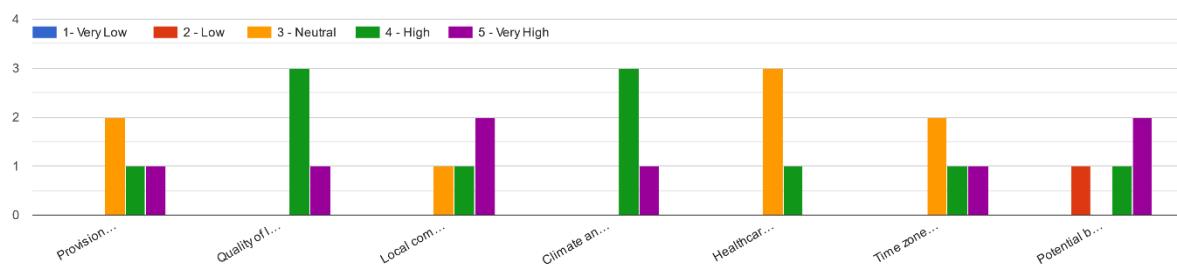


Figure 3.9: Motivation Information (Current & Completed Digital Nomads)

Figure 3.9 describes the motivation behind these current digital nomads in choosing to pursue a digital nomad lifestyle in Malaysia under the DE Rantau program. The key takeaway from the motivation is the ability to interact with the local community spirit and to broaden cultural horizons. Similarly, the quality of life still rings out relevantly to the motive of choosing a digital nomad lifestyle, which also ranks as important as climate and natural environment. Potential business opportunities are also highlighted in the interests of these digital nomads.

What methods do you typically use to search for information related to digital nomad visa programs? Select one or more applicable.

4 responses

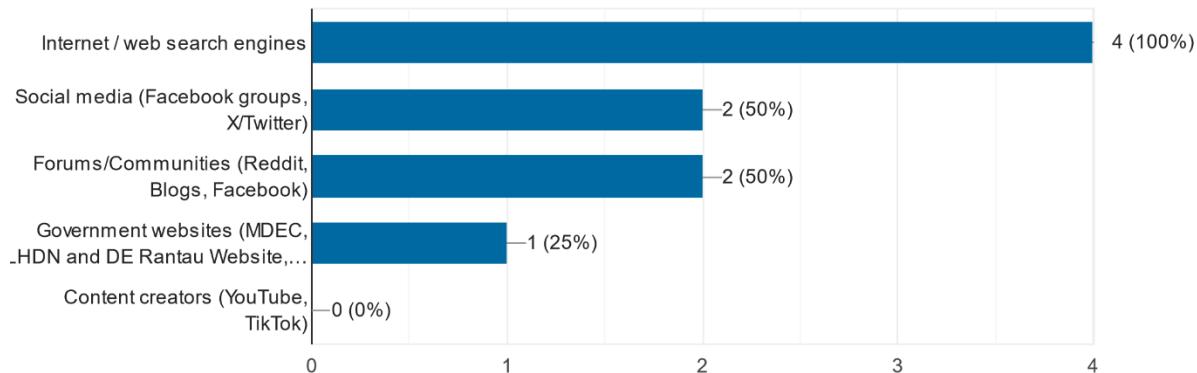


Figure 3.10: Information Searching Methods (Current & Completed Digital Nomads)

Based on Figure 3.10, the most used methods to search for information related to digital nomad visa programs were Internet/web search engines followed by social media and forums. Government websites were a lower priority for current and completed digital nomads, and none were interested in content creators.

Based on your experiences, indicate the degree of perceived difficulty you faced in the following aspects while researching the DE Rantau Nomad Pass Program:

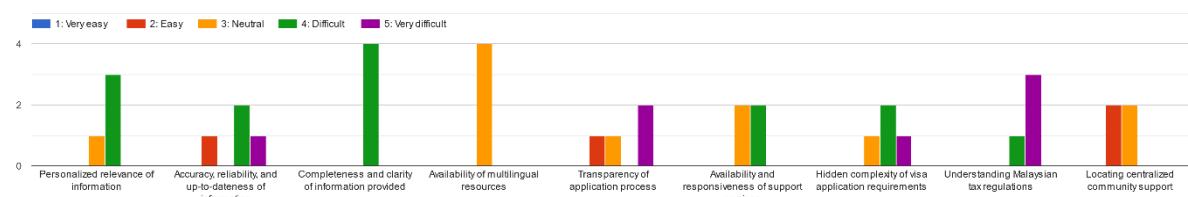


Figure 3.11: Perceived Difficulty of Research Aspects (Current & Completed Digital Nomads)

Figure 3.11 illustrates the aspects of researching the program ranked in terms of the difficulty of current and completed digital DE Rantau DNs. The clear most difficult aspects included the understanding of Malaysian tax regulations, especially once officially in Malaysia for longer durations. Then, the transparency of the application process, completeness and clarity of the information together with the personalization aspect also tell a similar story. The accuracy and availability aspects were also labelled as challenges albeit of lower magnitude.

Based on the following categories of information, rank the difficulty experienced carrying out research and finding support applicable to you

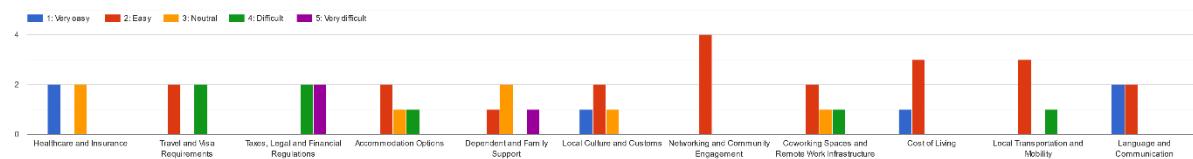


Figure 3.12: Perceived Difficulty of Categories of Information (Current & Completed Digital Nomads)

Figure 3.12 shows the difficulty levels experienced by digital nomads in obtaining various types of information. Healthcare and insurance were generally easy to find, with most ratings between "Very easy" and "Neutral." Travel and visa requirements had mixed responses, from "Easy" to "Difficult." Taxes and legal/financial regulations were notably challenging, with some finding it "Very difficult." Accommodation options, destination safety, and support were mostly rated as "Easy" or "Neutral." Local culture, networking, and community engagement also had moderate difficulty ratings. Language and communication barriers and cost of living were generally accessible, with ratings from "Very easy" to "Easy". Local transport and mobility were similarly manageable albeit with one difficult rating. Overall, while some categories were easy to navigate, taxes and legal/financial regulations and dependent information posed significant challenges.

Based on your experiences either from Malaysia and/or other countries, what are some larger challenges you might face while living as a digital nomad? Select one or more that apply.

4 responses

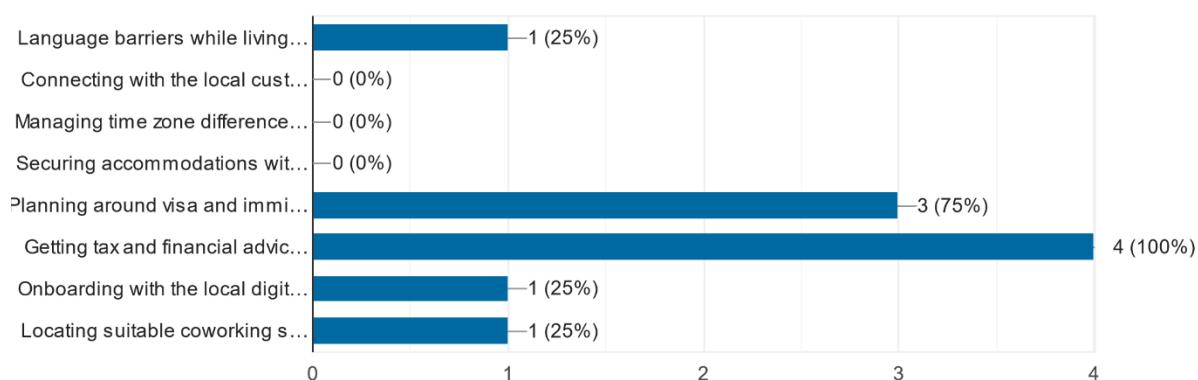


Figure 3.13: Challenges Experienced as Digital Nomads (Current & Completed Digital Nomads)

Figure 3.13 reports the challenges experienced by digital nomads in Malaysia under the DE Rantau program. An outstanding challenge was the act of receiving tax and financial advice with 100% of respondents selecting this category. Similarly, sentiment is shown regarding the uncertainty of visa and immigration plans in the longer term, extending up to exiting the country.

From the following, are there any existing platforms or services that you currently use and find particularly valuable for your nomadic lifestyle? Select all that apply.

4 responses

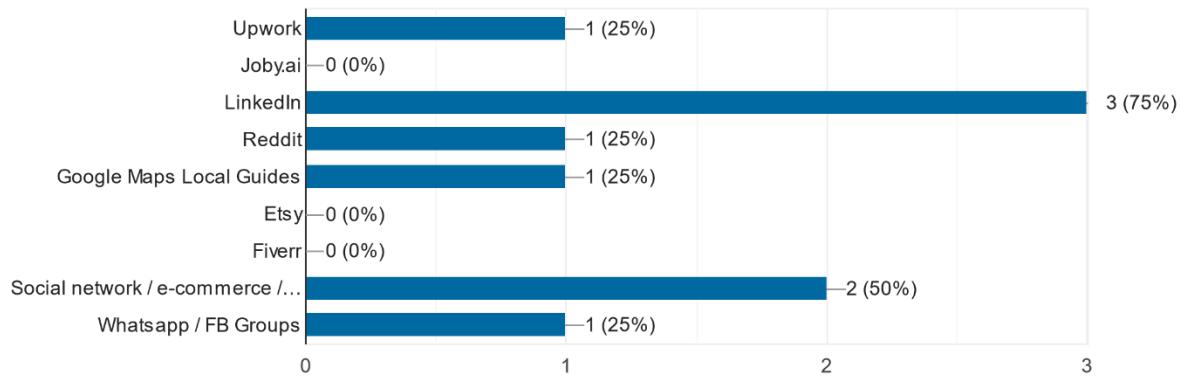


Figure 3.14: Existing Systems and Platforms Used (Current & Completed Digital Nomads)

Figure 3.14 reports on data collected that aims to go towards features for a community platform with networking and onboarding integrations. The respondents show a clear usage of LinkedIn as a platform that helps to connect similar-minded professionals from a wide range of industries and pathways. Some respondents also use Upwork, a freelancing platform. Some staples of the digital nomad lifestyle are also reported, such as Reddit, a social news platform, Google Maps Local Guides, social network platforms and finally WhatsApp / Facebook groups. This is important information to be able to understand the scope of the tools used by the current and completed digital nomads.

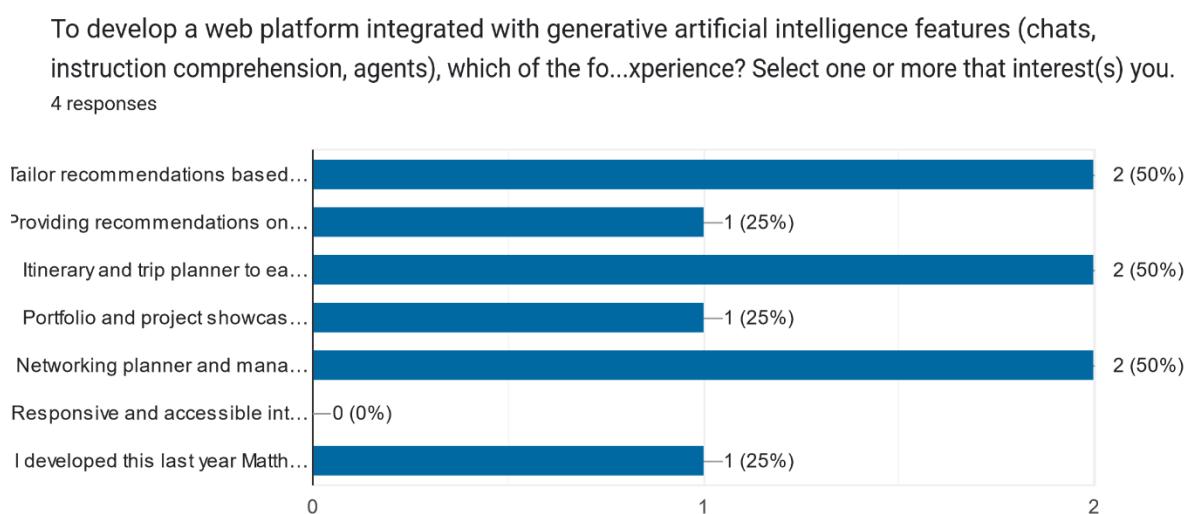


Figure 3.15: Potential Generative AI Features (Current & Completed Digital Nomads)

Figure 3.15 describes the highly dynamic nature of digital nomads wanting generative artificial intelligence to augment the staple workflows of digital nomadism. From the results, it is noted that these digital nomads seek tailoring recommendations when it comes to coworking spaces and accommodations, itinerary and trip planner to easily onboard onto local attractions and sightseeing opportunities and a planner for networking and managing connections with the local happenings. Other items of note are recommendations on local cuisines, portfolio showcase feature

Would you be interested in features that tailor to community incentivization and centralization for example, discussion forums, with other digital nomadities / businesses under the DE Rantau program?

4 responses

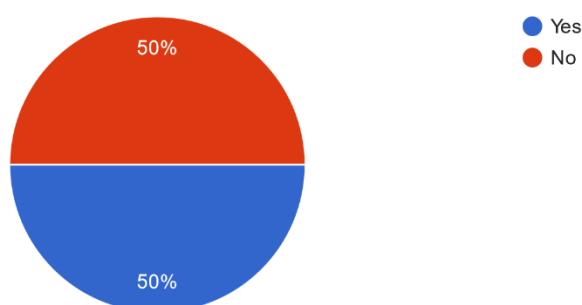


Figure 3.16: Interest in Community Platform (Current & Completed Digital Nomads)

The response towards a community platform is divisive, with equal amounts of scepticism and optimism. There is a noticeable lack of commitment by these digital nomads concerning participating in the agendas of MDEC.

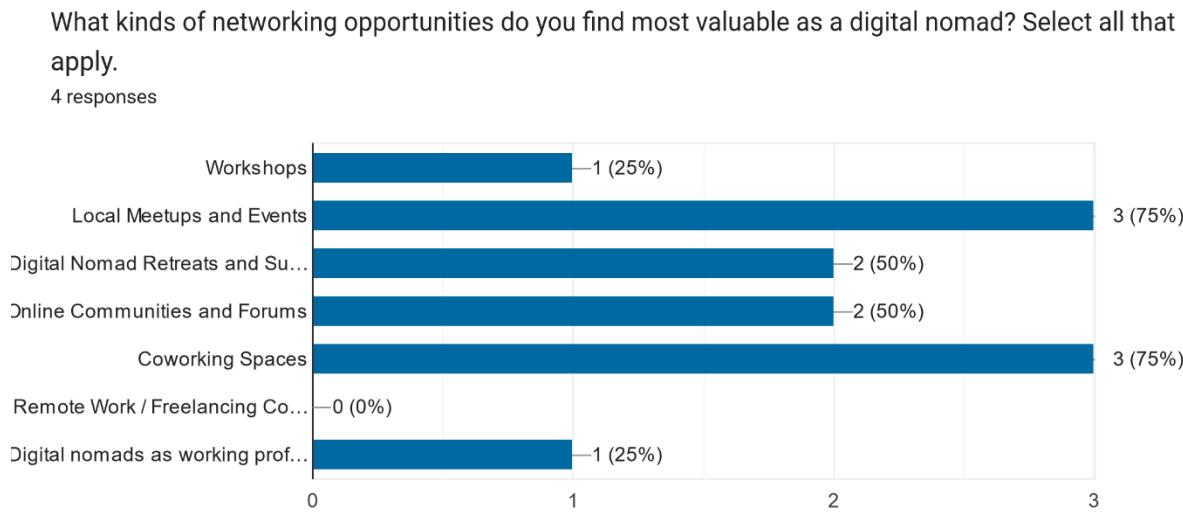


Figure 3.17: Key Networking Opportunity Options (Current & Completed Digital Nomads)

Based on Figure 3.17, the kinds of networking opportunities that current and completed digital nomads are interested in are local meetups and events, digital nomad retreats and summits and opportunities at coworking spaces. One respondent noted that digital nomads as working professionals with families are less interested in meeting other nomads (particularly those more like backpackers). It is therefore important to understand that a majority of the platform's potential users care more about baseline responsibilities about family and loved ones.

Feedback

In your opinion, what are some positive features / characteristics that MDEC and its DE Rantau Program including its Hub program that could be further developed and expanded, if any? (Open-ended)

0 responses

No responses yet for this question.

In your opinion, what are some negative qualities / characteristics possessed by MDEC and its DE Rantau Program including its Hub program that could be changed or improved, if any? (Open-ended)

1 response

There's way too much focus on KL area and nothing going on in other major cities (Johor, Penang etc.)

Figure 3.18: Key Feedback on DE Rantau Program (Current & Completed Digital Nomads)

The final figure, Figure 3.18, depicts that a notable negative quality/characteristic of the current iteration of the DE Rantau program is the focus on the KL area with not much going on in Johor and Penang. This important insight is crucial to validate the area scope of the PRAGmadic platform. This question marks the end of the analysis of responses based on the input from both categories of DE Rantau DNs, which are the applicants and also the current active DNs in Malaysia. These insights prove to be valuable reality checks in the digital nomad ecosystem as it is. Key takeaways include being first and foremost cognizant of the dependent-focused digital nomads consisting of older generations of the male demographic. These digital nomads are also highly educated and skilled in various IT and technology sectors. Finally, there is some evidence of an interest in choosing Penang as a hub requiring more attention by the program.

3.2.2 Planning

Planning in the prototyping model involves outlining the iterative cycles of prototype development, testing, and refinement. Each iteration is carefully planned to focus on specific features or functionalities based on the communication phase carried out prior, allowing the development team to tackle complex aspects of the system incrementally. Taking the digital nomad onboarding support platform as an example, planning includes scheduling user feedback outlets to be released immediately, allocating resources upfront for development and testing, and setting timelines for each iteration. The key to the Planning phase in the prototyping paradigm is speed. Therefore, it is critical to break things fast to identify potential risks in the system. As such, choosing to go with service providers in aspects of the platform such as the backend and AI API providers helps to reduce the planning necessary. There is also the developer human factor, which considers risks related to the capabilities of development teams.

Executing and modelling risk analysis is a critical component of the planning phase. Potential risks, such as delays in receiving user feedback, integration challenges with third-party services, like Supabase or OpenAI, and evolving user requirements, are identified and mitigation strategies are developed. By anticipating and planning for these risks, the development team can proactively address issues, ensuring that the project stays on track and meets its objectives. Then, the plan can be executed with the proper management and manpower allocated. However, in the context of this project, seeing that the overall development team consists of just one developer, foresight and proper software artefacts are required to ensure the complexity does not outgrow the capabilities at hand.

Therefore, on the development side, certain artefacts to be used in the foreseeable future in the scope of project management include a Gantt Chart and Kanban Board using Trello.

3.2.2(a) Gantt Chart

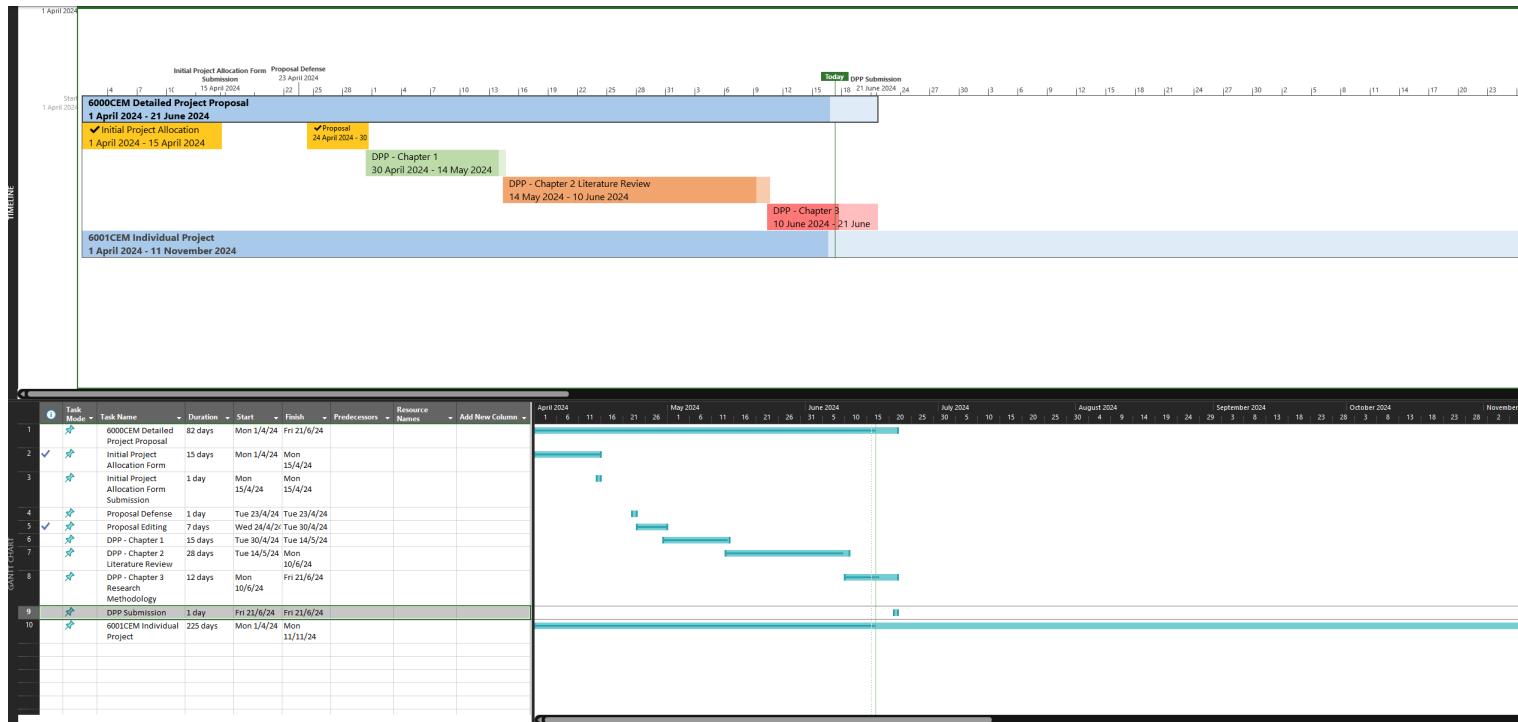


Figure 3.19: Gantt Chart with Overview of Detailed Tasks

The Gantt chart in Figure 3.19 is a valuable project management tool that provides a visual timeline of tasks and milestones, allowing for efficient planning and scheduling. It highlights the start and end dates of each activity, enabling supervisors to track progress and identify any potential delays. By clearly outlining dependencies between tasks, it ensures that all team members of a development team are aware of designated responsibilities and the sequence of events. Additionally, the Gantt chart facilitates better resource allocation and helps in maintaining a balanced workload, ultimately contributing to the successful completion of the project.

3.2.2(b) Kanban Board

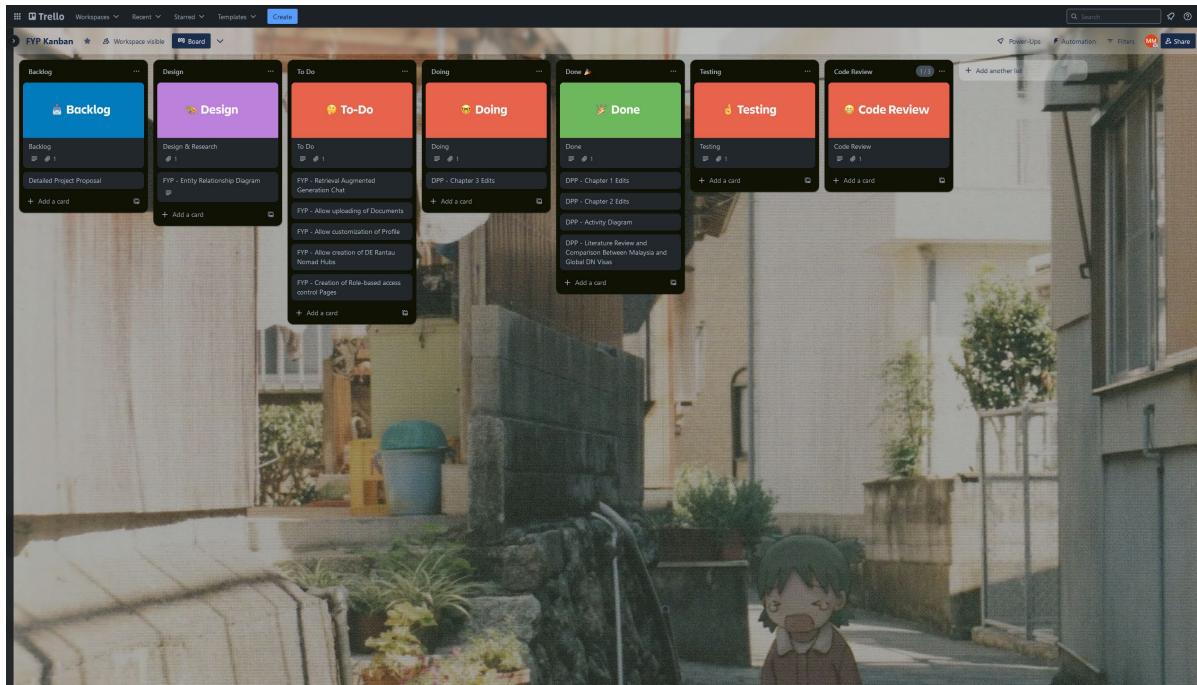


Figure 3.20: Kanban Board of Pragmatic

The Kanban board shown in Figure 3.20 depicts a comprehensive and detailed development process view, illustrating the flow of tasks from inception to completion. Each column represents a different stage of the workflow, allowing team members to visualize the progress of individual tasks. This real-time tracking aid promotes transparency, collaboration and accountability in the mental model of the development process. The Kanban board ensures that all objectives are clearly designated and can swiftly address any new changes in the requirements of the project.

3.2.3 Modelling

Modelling in the prototyping process involves creating a quick design that focuses on the visible aspects of the system, such as the user interface and user experience. This quick design serves as the blueprint for building the initial prototype, allowing stakeholders to visualize the system and provide early feedback. Modelling in the context of Pragmatic includes designing the user interface for key features like the RAG-based Answer Engine, user profile, networking tools, and onboarding processes. Given the platform support extending to the web browsers of the intended users' devices, the scope of the modelling required is understood in nature. The diagrams that are included in this section therefore also need to match the degree of the upfront development expectations. As of writing, diagrams for modelling the user experience include items like activity diagrams for various parts of the system functionality.

The quick design must balance simplicity with functionality, ensuring that the prototype can demonstrate the core features effectively. In this phase, evaluating the potential feasibility of integrating various components will be carried out, such as AI-driven search capabilities and real-time data synchronization. The process will require a suite of tools like Figma and Draw.io to help ensure that the design can be iteratively refined without significant overhead. The key is to stick to tools that are already familiar to keep the mental overhead reserved for designing the user stories and use cases.

3.2.3(a) Activity Diagram

An activity diagram is designed using UML standards and terminologies. This is depicted in the activity diagrams below using activity nodes that represent various actions taken by users like digital nomads, MDEC, and the Malaysian Immigration Department. Decision nodes are included to indicate decision points in the process, such as the approval or denial of the visa application. Meanwhile, swimlanes diagrammatically differentiate the responsibilities of the various entities.

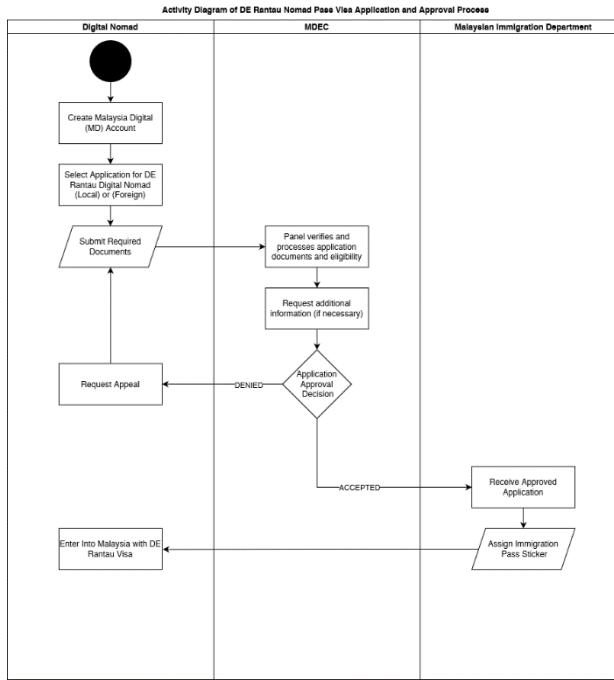


Figure 3.21: Activity Flow Diagram of DE Rantau Nomad Pass Visa Application and Approval Process

The activity diagram in Figure 3.21 delineates the application and approval process for the DE Rantau Nomad Pass Visa, highlighting the distinct roles and responsibilities of the applicant (digital nomad), MDEC, and the Malaysian Immigration Department. The figure illustrates a streamlined and systematic approach where the digital nomad initiates the process, MDEC evaluates and decides on the application, and the Immigration Department finalizes the process by issuing the visa. The digital nomad swimlane identifies the boundaries of the steps taken by the digital nomad, including creating the Malaysia Digital Nomad Account, selecting the application, submitting required documents, and possibly requesting an appeal if the application is denied. The MDEC swimlane outlines the responsibilities of the Malaysia Digital Economy Corporation (MDEC), which include verifying and processing the application, documents, and eligibility, and making the final application approval decision. The Malaysian Immigration Department swimlane illustrates the steps taken upon receiving approved applications which include assigning the immigration pass sticker. The diagram attempts to depict a high-level overview of the structured workflow and decision-making points, ensuring transparency and clarity in the visa application process for the proposed system.

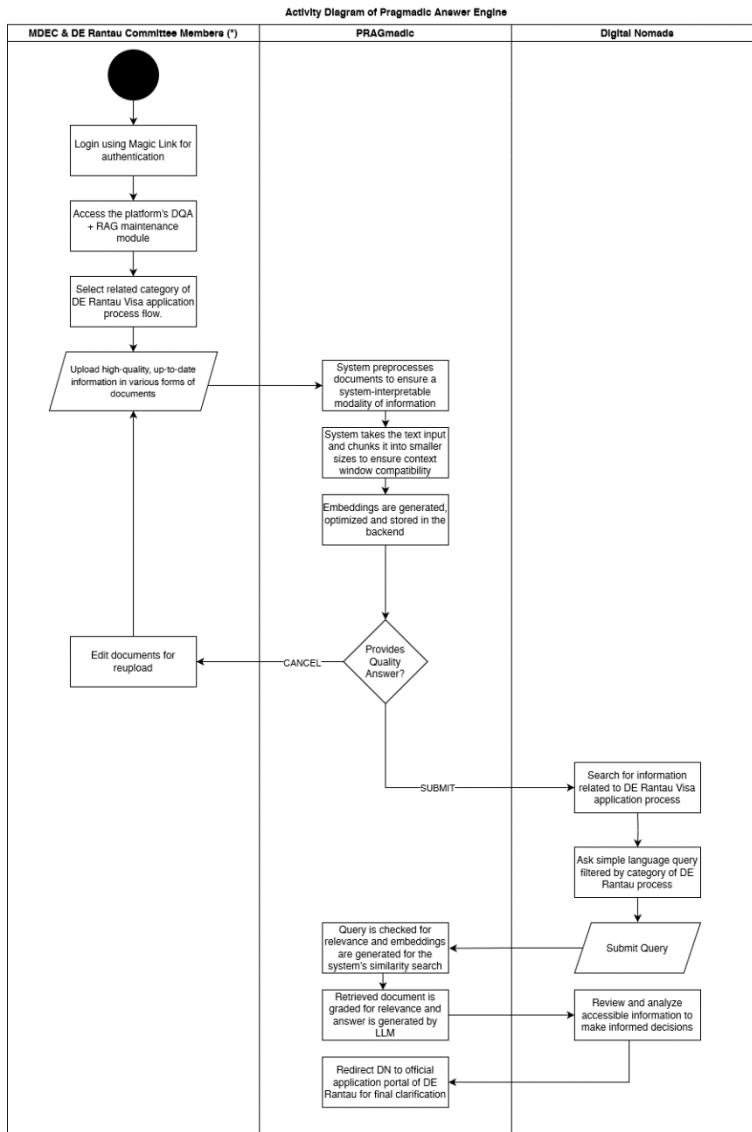


Figure 3.22: Activity Diagram of Pragmatic Answer Engine

The activity diagram in Figure 3.22 of the Pragmatic answer engine illustrates a streamlined process for providing more accurate and contextually relevant responses to digital nomads using retrieval augmented generation (RAG). It starts with MDEC and DE Rantau Committee members logging in via Magic Link for authentication, accessing the FAQ module, and selecting the relevant DE Rantau visa application category. These accounts emulated by the system are responsible for uploading high-quality, up-to-date information, which is then processed to generate potential answers based on the user's query. If necessary, documents can be edited for improved accuracy before submission. Digital nomads carry out querying using plain language which is then checked for relevance and accuracy through embeddings and graded by a knowledge graph of agents. The final responses are sent back to the digital nomads, ensuring that precise and reliable information is conveyed regarding the DE Rantau visa

application process. The diagram models the collaborative effort between automated systems and human oversight to maintain high standards of information quality and user satisfaction.

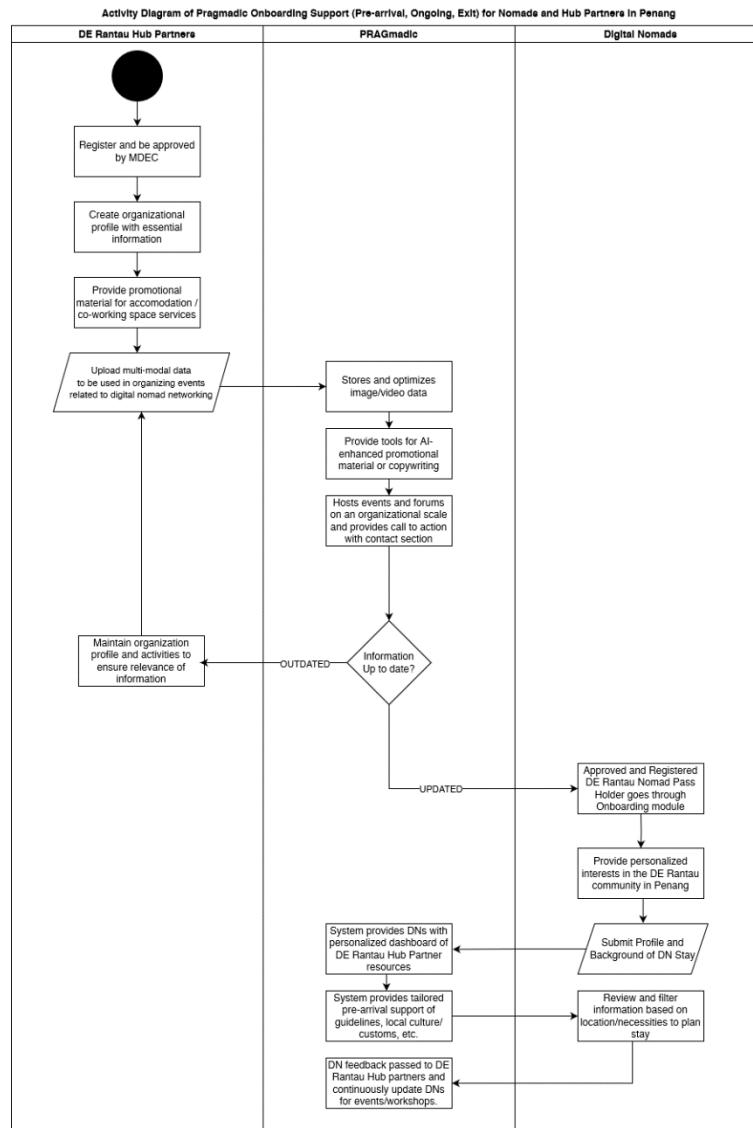


Figure 3.23: Activity Diagram of Pragmatic Onboarding Support Platform

The activity diagram in Figure 3.23 represents the community platform aspect of the Pragmatic outlining a comprehensive onboarding, ongoing support, and exit process for nomads and nomad hubs in Penang. It begins with the registration and approval of hub partners by MDEC, followed by the creation of organizational profiles and uploading of promotional and operational data. This data is then stored and optimized by Pragmatic, ensuring it remains current and relevant. Digital nomads access the platform to find and register for hubs, receiving personalized support and community engagement opportunities. Then, digital nomads customize the respective profiles with stay details, which are then reviewed and approved by

hub partners into relevant business offerings. These DE Rantau Nomad Hubs can then provide tailored resources and integration into the local community for the digital nomads. The system also provides ongoing updates and support, ensuring both nomads and hub partners have the necessary tools and information for a successful experience. This process emphasizes continuous interaction, up-to-date information, and personalized support to enhance the digital nomad experience in a hypothetical setting.

3.2.3(b) Use Case Diagram

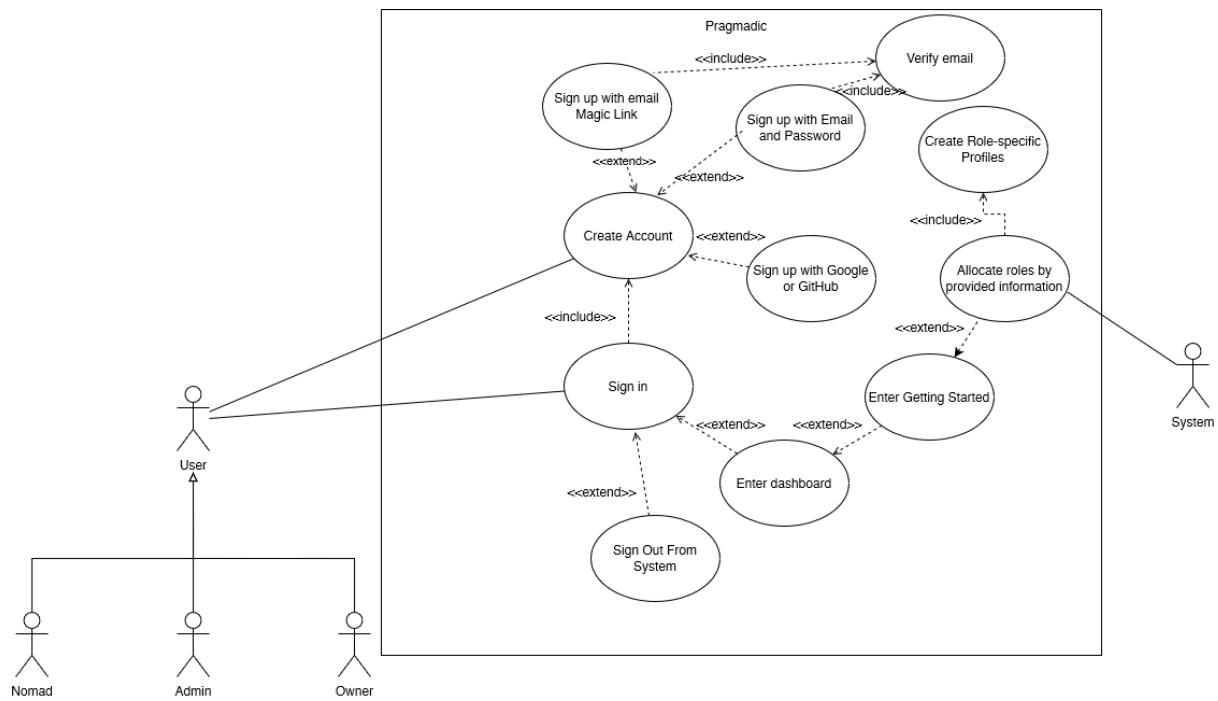


Figure 3.24: Use Case Diagram for RBAC and Registration

The use case diagram seen Figure 3.24 partly depicts the Pragmadic platform provided functionality. The diagram depicts essential interactions between users and the system, highlighting the core functionalities available to several/different user roles. At the heart of the diagram is the "User" actor, which serves as a general representation of all users interacting with the platform for this Use Case Diagram. This actor can be further specialized into three distinct roles: Nomad, Admin, and Owner, each with specific responsibilities and capabilities. The diagram encapsulates the primary use cases within one of the system boundaries of Pragmadic. These use cases include account creation, sign-in, and sign-out processes, which are fundamental to user interaction. The "Create Account" use case is extended by various registration methods, such as signing up with an email magic link, email and password, or

through social platforms like Google and GitHub. This flexibility ensures that users can choose their preferred method of registration, enhancing user experience and accessibility. The "Sign In" use case is as well central to user access, allowing users to authenticate and gain entry to the platform. Once signed in, users can enter the dashboard or access the "Getting Started" section, which provides guidance and initial setup for new users. The inclusion of "Verify Email" and "Allocate Roles by Provided Information" included use cases ensures that user accounts are secure, and roles are appropriately assigned based on user input. The system actor represents automated processes within the platform, such as role allocation and profile creation, which are included in the account setup process. This automation streamlines user onboarding and ensures consistency across user profiles.



Figure 3.25: Additional Use Case Diagrams for RAG Chat and Onboarding

More use case diagrams can be created to help support and contextualize the system's development. In Figure 3.25, flows like RAG Chat and Onboarding are clearly depicted within the system boundaries according to the actions of the respective roles as actors. The system is depicted as an abstraction of the large language model-based automation and systematic pipeline that helps contextualize LLM generations for users who wish to seek support for DE Rantau related information.

3.2.3(c) Entity Relationship Diagram (ERD)

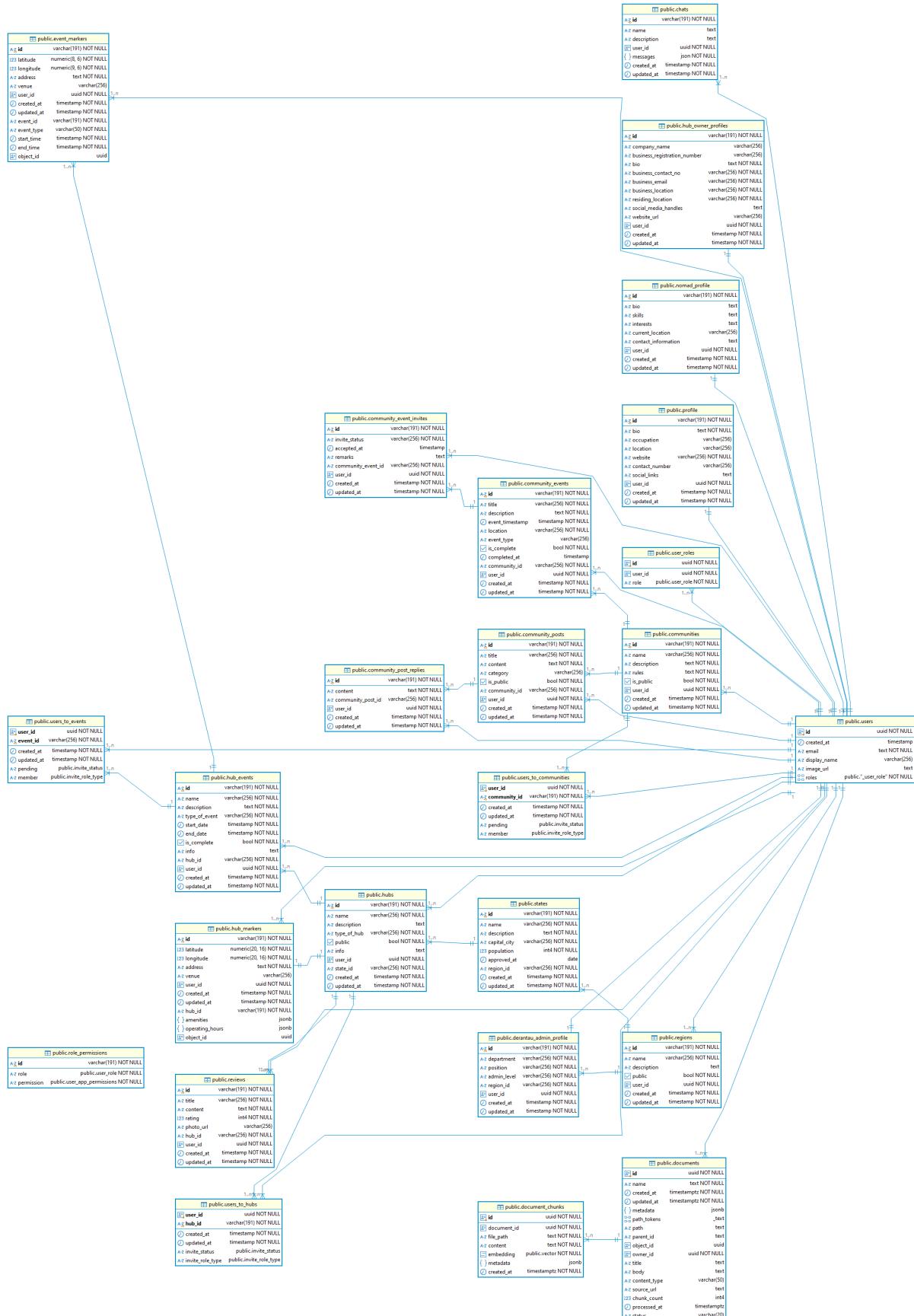


Figure 3.26: Entity Relationship Diagram of Pragmatic

The ERD seen in Figure 3.26 represents the database schema for the Pragmatic platform, modelling the relationships between various entities involved in managing digital nomad activities, events, and user interactions. The diagram is structured to support the platform's core functionalities, including user management, event organization, chats, and community engagement.

Table 3.1: Brief Overview of Key Entities and Relationships of Pragmatic

Entity	Description
Users (user)	The central entity in the schema, representing individuals using the platform. Each user has attributes such as id, email, display_name and roles. The role attribute is crucial for implementing role-based access control, distinguishing between admins, nomads, hub owners, and regular users. This is done through an enum called “user_role” to enforce strict operations.
Profiles (“profile”)	Linked to users, profiles store additional information like bio, occupation, and more. This separation allows for flexible user data ownership, management and supports profile customization. It is an optional relationship, where one and only one user can have zero to one profiles.
Role Profiles (“nomad_profile”, “hub_owner_profile”, “derantau_admin_profile”)	Similar relationships between “profile” and user. Contains varying fields.

Hubs (“hubs”)	Represent DE Rantau recognized businesses or locations. Each hub has attributes like name, location, and description. Hubs are associated with events and can be managed by hub owners. One hub owner can manage multiple hubs. A user can be given membership through the corresponding (“user_to_hubs”) table that handles statuses and role of a non-owner user in that specific hub.
Communities (“communities”)	Represent an entity that can have posts, events and members. Similar relationships between users and communities as with hubs. Posts can have one author, and an author can have many posts in one community. Then similarly with replies.
Hub and Community Events	These are central entities to the platform's functionality, events are organized by hubs and can be attended by users. Community events also are similar. Attributes include all that are expected. The relationship between events and hubs is one-to-many, indicating that a hub can host multiple events. The associative entity of join tables for events links users to events, capturing participation details. It includes attributes like user_id and event_id, facilitating the tracking of user engagement in events.

Reviews	Users can leave reviews for hubs, providing feedback and ratings. This entity includes attributes such as rating, comment, and user_id, linking reviews to both users and hubs.
Regions and States	These entities define the geographical hierarchy, with regions containing multiple states. This structure supports location-based filtering and management of hubs and events.
Chat	Specially used to accommodate Vercel AI SDK core messages format. Stores chat messages in JSON and is consumed appropriately using helper types on the client.
Documents and Document_Chunks	Documents are closely tied to the object_id in which references Supabase Storage's schema. Documents are then chunked and stored into document_chunks table which has special pgvector types known as embeddings of dimensions 1536. This corresponds to OpenAI's ada embedding model. This is used in the system for vector search.

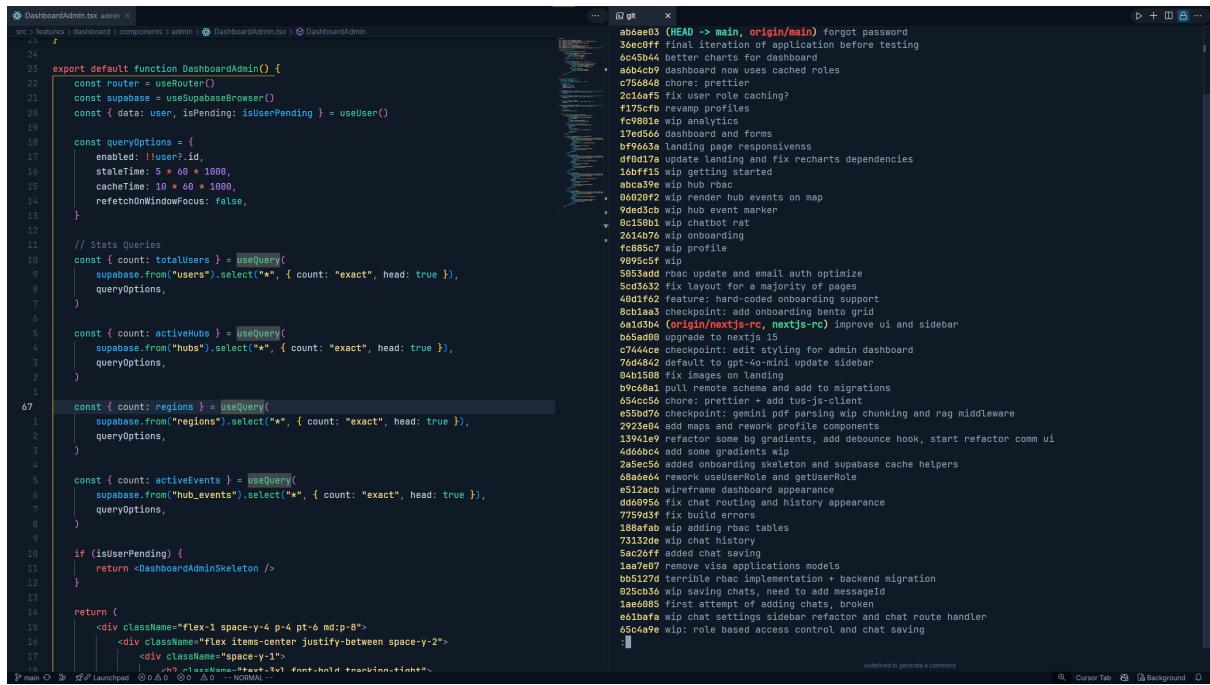
The ERD effectively models the complex interactions and data requirements of the Pragmatic platform. By organizing entities and relationships logically using PostgreSQL foreign keys, it supports the platform's objectives of facilitating digital nomad activities, enhancing community engagement, and providing robust user management. The schema's design ensures scalability

and flexibility, accommodating future enhancements and additional features especially with the DE Rantau system itself spanning far more data business logic requirements.

3.2.4 Construction

The construction phase places an emphasis on ensuring the development resources required are clearly laid out. The rapid timeline of the overall software development life cycle requires building the initial prototype based on the quick design. This phase emphasizes rapid development to produce a working model that stakeholders can interact with. This outlet for development requires familiar development environment tools such as a code editor, version control and assistance through means like generative AI up to a certain extent for questions that are not easily searched for. In PRAGmadic's development this involves implementing core functionalities right from the get-go, such as user authentication tied to user management directly in the backend, setting up proper security policies on data storage, and the basic RAG-based Answer Engine, using technologies like Next.js, Supabase, and OpenAI APIs.

3.2.4(a) Developer Environment



The screenshot shows a code editor window for a file named `DashboardAdmin.tsx`. The file content is a TypeScript component named `DashboardAdmin` that uses various hooks and queries from `useSupabase` to fetch data. The right side of the screen displays a detailed commit history for the `main` branch of the repository, listing numerous commits with their hash codes and descriptions, such as "forgot password", "better charts for dashboard", and "fix user role caching". The commit list is scrollable, indicating a large history of changes.

```

1 // DashboardAdmin.tsx admin
2
3 export default function DashboardAdmin() {
4   const router = useRouter()
5   const supabase = useSupabaseBrowser()
6   const { data: user, isPending: isUserPending } = useUser()
7
8   const queryOptions = {
9     enabled: !user?.id,
10    stateTime: 5 * 60 * 1000,
11    cacheTime: 10 * 60 * 1000,
12    refetchOnWindowFocus: false,
13  }
14
15  // Stats Queries
16  const { count: totalUsers } = useQuery(
17    [supabase.from("users").select("*", { count: "exact", head: true })],
18    queryOptions,
19  )
20
21  const { count: activeHubs } = useQuery(
22    [supabase.from("hubs").select("*", { count: "exact", head: true })],
23    queryOptions,
24  )
25
26  const { count: regions } = useQuery(
27    [supabase.from("regions").select("*", { count: "exact", head: true })],
28    queryOptions,
29  )
30
31  const { count: activeEvents } = useQuery(
32    [supabase.from("hub_events").select("*", { count: "exact", head: true })],
33    queryOptions,
34  )
35
36  if (isUserPending) {
37    return <DashboardAdminSkeleton />
38  }
39
40  return (
41    <div className="flex-1 space-y-4 p-4 pt-6 md:p-8">
42      <div className="flex items-center justify-between space-y-2">
43        <div className="space-y-1">
44          <div style={{ height: 15, width: 15 }}></div>
45        </div>
46      </div>
47    </div>
48  )
49}

```

Figure 3.27: Cursor with Pragmadic Main Branch Open

The code editor used for development purposes is Cursor pictured in Figure 3.27, a free VS Code fork code editor that provides extensive support for extensions and language server

protocols (LSP) for multiple languages. Cursor builds on VS Code's flexibility and powerful features make it the ideal choice for a developer working in diverse coding environments, especially when bouncing around the frontend and backend. Its full suite of conveniences and tools like debuggers, built-in GitHub Copilot support, and customization options help to enhance and streamline the development process. Moreover, the vast library of extensions available in the VS Code marketplace allows developers to tailor linting, type hints, code autocomplete, refactoring actions and other kinds of helpful developer tooling.

3.2.4(b) Version Control Strategy – Feature Branching

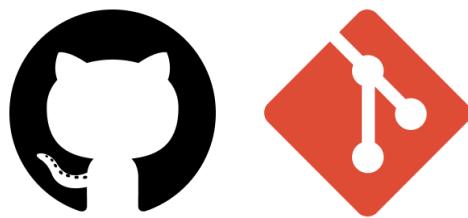


Figure 3.28: GitHub and Git as Version Control Tools

The version control utilized in the construction phase revolves around GitHub and Git seen in Figure 3.28, focusing on feature branching within the scope of a single developer team. This approach ensures that new features are developed in isolated branches, allowing for better management of code changes and reducing the risk of conflicts. Once a feature is complete, it can be merged back into the main branch after thorough testing and review. This method enhances collaboration and clarity of separated areas of concern, even in a single-developer setup, by maintaining a clean and organized codebase. It also facilitates managing to leverage continuous integration and deployment, ensuring that new features are seamlessly integrated into the production environment by being tested as a unit. The blast radius of unexpected and unpredictable bugs can be reduced to ensure Git commands can easily be applicable to solve common software engineering problems.

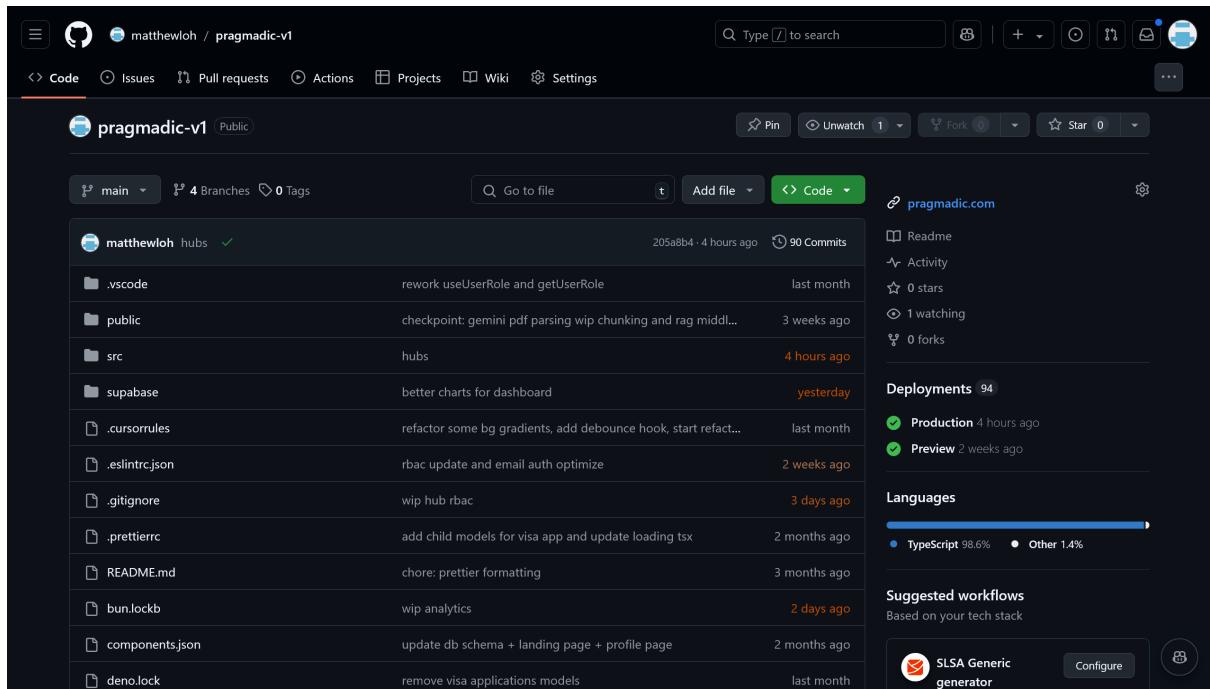


Figure 3.29: GitHub Repository of Pragmadic

Figure 3.29 demonstrates that GitHub is used to serve as the remote origin of the entire source code. GitHub offers developers an arsenal of powerful collaborative functionality that includes developer-friendly pages for issue discussions, pull requests, project wikis and other insights that integrate into the development workflow. The reliable availability of the source code helps to ensure that fixes can be performed where and when necessary. GitHub provides extremely important security features and foolproof means to help avoid catastrophic mistakes when dealing with sensitive code or information like private keys.

```

origin/main
└─ git branch -a
* main
remotes/origin/main
└─ git branch rag-chat
└─ git checkout rag-chat
Switched to branch 'rag-chat'
└─ git branch -a
  main
* rag-chat
remotes/origin/main
└─ git push -u origin rag-chat
Total 0 (delta 0), reused 0 (delta 0), pack-reused 0
remote: This repository moved. Please use the new location:
remote: https://github.com/matthewloh/pragmadic.git
remote:
remote: Create a pull request for 'rag-chat' on GitHub by visiting:
remote: https://github.com/matthewloh/pragmadic/pull/new/rag-chat
remote:
To https://github.com/matthewloh/supabase-next-project.git
 * [new branch]   rag-chat -> rag-chat
branch 'rag-chat' set up to track 'origin/rag-chat'.
└─ git

```

Figure 3.30: Git CLI used to create feature branches

In practice, the terminal such as in Figure 3.30 is going to primarily be used to leverage the immense power that comes built-in with Git as a version control solution. In tandem with VS Code, Git can be provided abstractions to help reduce the mental burden of figuring out the extensive amounts of commands provided in Git.

3.2.5 Testing

Further down the line of development, some principles of test-driven-development should be included, such as basic functional testing and non-functional testing, to help ensure the development team is cautious of implementation compromises made for the sake of speed. These shortcuts can lead to technical debt if not properly managed. Another extremely important aspect is the handling of security and safety when it comes to LLM output, something the field has yet to come up with a robust solution for. The foresight of the project in this phase involves identifying areas where technical debt may accumulate and planning refactoring efforts to ensure that the system remains testable, maintainable and scalable as it evolves especially in response to the feedback from users, which is discussed later on.

3.2.5(a) Black Box Testing

Black Box Testing is a crucial functional testing methodology used to evaluate the system's functionality without examining its internal code structure. This method focuses on testing the inputs and outputs of the software, ensuring that the system behaves as expected under various conditions. For Pragmatic, Black Box Testing involves creating test cases based on the software requirements and specifications. This includes scenarios such as successful user registration, registration with invalid email formats, and registration with already-used email addresses. Additionally, it involves testing the RAG-based Answer Engine by providing various queries and verifying that the responses are relevant and accurate. The functionality of navigation links, buttons, and forms across the platform is also tested to ensure all work as intended. The benefits of Black Box Testing are that it ensures the system meets the functional requirements, helps identify discrepancies between the expected and actual system behaviour, and does not require knowledge of the internal code structure, making it easier for testers to create comprehensive test cases.

3.2.5(b) Integration Testing

Integration Testing focuses on verifying the interactions and interfaces between different modules or components of the system. This testing methodology is essential for ensuring that the integrated parts of the software work together seamlessly. In the context of Pragmatic, Integration Testing will involve testing the communication between the frontend, backend, and AI layers. For the Frontend-Backend Integration, tests will be conducted to check the data flow between the user interface (Next.js) and the backend (Supabase), including scenarios such as data submission through forms and data retrieval for display. The Backend-AI Layer Integration will verify the interaction between the backend database (Supabase) and the AI layer (OpenAI and Gemini), focusing on testing the retrieval of embeddings from the vector store and the generation of responses by the AI models. End-to-end workflows will also be tested, covering complete workflows like a user querying the RAG-based Answer Engine and receiving accurate and relevant responses based on stored embeddings. The benefits of Integration Testing include ensuring that individual modules work together correctly, identifying interface defects and integration issues early in the development process, and validating the overall system functionality and data flow across different components.

3.2.5(c) Security Testing in Web and LLM-based Applications (Non-functional)

OWASP Top 10 for Large Language Model Applications

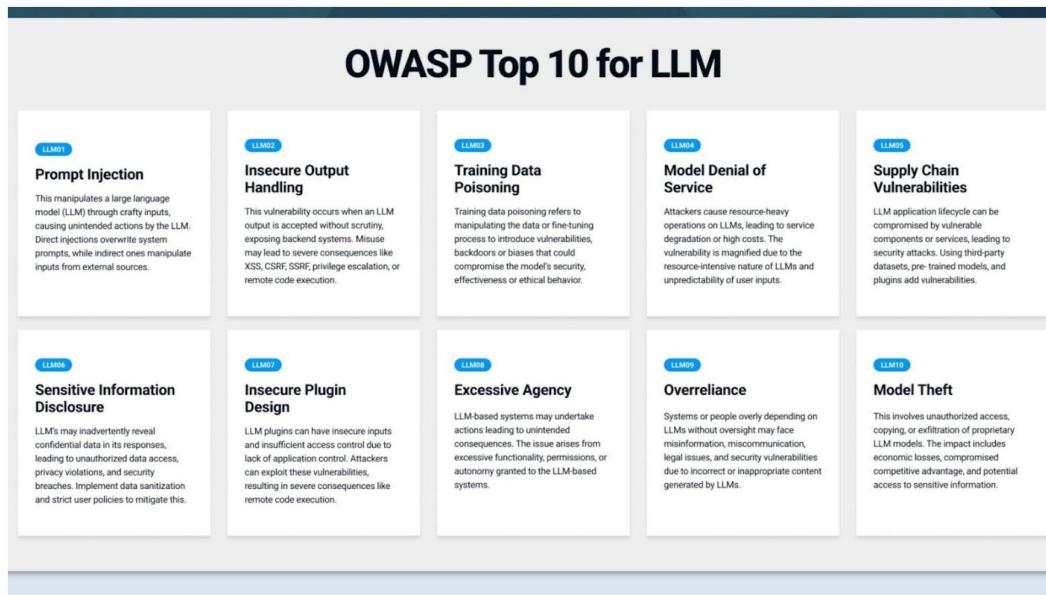


Figure 3.31: OWASP Top 10 for Large Language Model Applications (OWASP, 2024)

Based on Figure 3.31, OWASP has provided an extensive threat analysis and Top 10 threat vectors / considerations when using LLM-integrated solutions. Following in a similar vein, security testing within the scope of non-functional testing will be implemented in this project. In order to do this, OWASP (2024), recommends threat modelling play a crucial role in preemptively identifying vulnerabilities in LLM-integrated solutions. Therefore, it is important to follow OWASP's suggestions for a systematic, repeatable approach to threat modelling that specifically addresses GenAI-targeted attacks. This includes evaluating whether LLM solutions are equipped to detect and mitigate harmful inputs, unauthorized data access, and malicious query submissions. Additional areas to prioritize is Model Denial of Service and Prompt Injection, which will be covered in the testing section.

3.2.6 Deployment and Maintenance

Deployment in the prototyping model involves making the prototype available to stakeholders for preliminary evaluation. This phase allows users to interact with the system, test its functionalities, and provide feedback. The project's deployment phase will include setting up the prototype both on localhost and deployment on a platform like Vercel, ensuring that it would be accessible to users including digital nomads and digital nomad business owners for

testing and feedback. The advantages of using a deployment platform are the ability to leverage analytics built-in to the platform and handle scaling in both directions to meet demands in traffic. Eventually, if the project were to be successful in attracting stakeholder interest, it would be a seamless transition to hand off the technical implementation by simply adding the relevant parties as administrators on the platform. Considerations crucially need to be made regarding validating the expected functionality in the different environments of development and production.

During deployment, it is essential to monitor system performance and gather user metrics systematically, something that is difficult to get just right in accordance with data privacy and protection regulations, especially on the Internet. Risk analysis includes identifying these customer identity protection issues ahead of time akin to potential deployment issues, such as server downtime or performance bottlenecks, and ensuring that adequate monitoring and support mechanisms are in place to address these issues promptly. Because of the nature of the prototyping model, the developer experience of integrating a continuous integration / continuous deployment pipeline is crucial to meet project deadlines whilst also ensuring application uptime and availability.

After being developed, the cost structure of running a platform like Pragmatic has been revealed. The managed IaaS and BaaS providers in Vercel and Supabase were used in this project in order to provide easy prototyping in accordance with the necessitated software methodology efforts. Additionally, the deployed web application utilizes various other services for features like rate-limiting, LLM inference as detailed in the previous sections and email providers. Therefore, it is important to understand within the wider project context that some fees may grow exponentially in a real-world environment as the consumed services could scale through serverless means to an extreme degree with greater userbases. In total, considerations should be made to explore self-hosted solutions that use open-source software while still relying on industry standard technologies to abide with regulations and expectations in modern web development.

3.2.7 Feedback and Iteration

The feedback and iteration phases are where the prototype undergoes continuous refinement based on user feedback. The prototyping paradigm is inherently iterative, meaning that feedback received is mission-critical for the application to fulfil expectations. Stakeholders like

mock DE Rantau nomads should provide input on the prototype's usability, functionality, and overall performance, highlighting areas for improvement in tailoring the application features to certain workflows. For the application, this involves each iteration aiming to bring the prototype closer to what is needed instead of what is wanted by the developers. Feature enhancements on the large language model used in the RAG system are examples of feedback based on user testing and stakeholder input can be collected. This iterative cycle continues until the prototype is sufficiently refined to meet the initial goals set during the communication phase. This iterative nature means that models from other AI API providers at the forefront of innovation can be integrated without being locked in. Each iteration cycle involves revisiting the planning, modelling, construction, and deployment phases to implement the necessary changes. Risk analysis in this phase includes managing scope creep, ensuring that feedback is actionable and prioritized, and maintaining a balance between adding new features and refining existing ones. By iteratively refining the prototype, the development team can ensure that the final product meets user needs and expectations effectively.

In conclusion, the prototyping model is a highly effective approach for the system and its development resource constraints, such as team size and expertise, allowing for iterative refinement based on continuous stakeholder feedback. By sticking to prototyping paradigm fundamentals like communication/requirements gathering, planning, modelling, construction and testing, deployment, and feedback and iteration, the development process remains flexible and responsive to accommodate user needs. Through implementing some careful risk analysis and management at each phase, the project can address potential challenges in adapting the Prototype model for developing an AI application proactively, ensuring that the final product is accurate, robust, scalable and aligned with the objectives of the DE Rantau program and the needs of the stakeholders.

3.3 Conclusion

In this chapter, the overall research methodology was discussed. The selected Prototype model was discussed in terms of its stages, including communication/requirements gathering, planning, modelling, construction, testing, deployment, feedback and iteration. Throughout these topics, key artefacts of the Pragmadic system were included, such as the full results and analysis of the questionnaire given out to carefully shape the contextual information of the intended audience of the application. An assortment of activity diagrams, use case diagrams and an entity relationship diagram related to the problem domains of the project were also

included to model the deeper relationships between entities in the system. A Gantt chart and Kanban board are used to help in project management efforts while in terms of developer environment tooling, Cursor is used as a code editor along with Git and GitHub as version control software solutions. The version control strategy used is Feature Branching. In terms of testing, two functional testing methods were discussed, Black Box Testing and Integration Testing. Simultaneously, Security Testing for LLM Applications following OWASP recommended principles was selected as non-functional testing to ensure the front-facing and backend experience on various platforms and browsers is as expected for security, accessibility and other related standards.

CHAPTER 4: PROJECT IMPLEMENTATION

4.1 Introduction

This chapter presents the culmination of what was accomplished during the Pragmatic web platform's development, detailing the implementation of each core module and architectural decision made across a full-stack application that attempt to fulfil the project's objectives. The following sections together with respective code snippets and feature screenshots will be provided with analysis and explanations. Some sections also will provide information on the manner in which certain technologies were employed throughout the development process such as in higher level web development, deployment and production service principles.

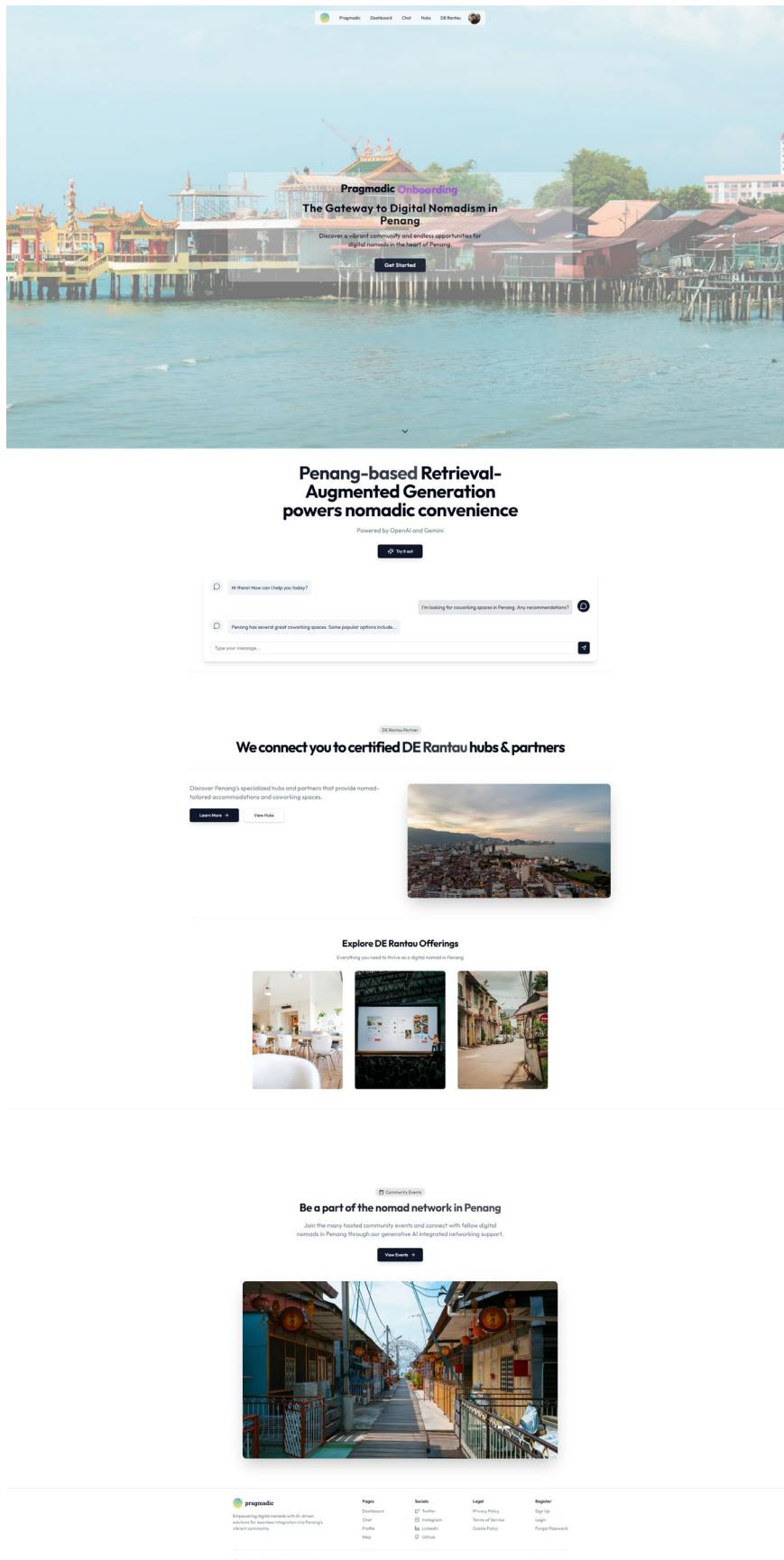


Figure 4.1: Pragmatic Landing Page

4.2 Login and Registration Module

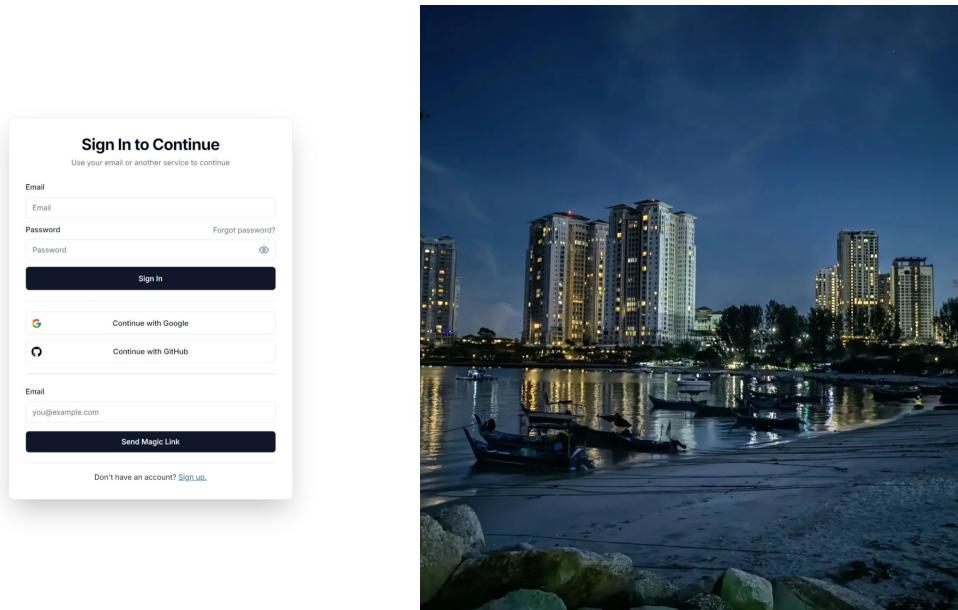


Figure 4.2: Login Page

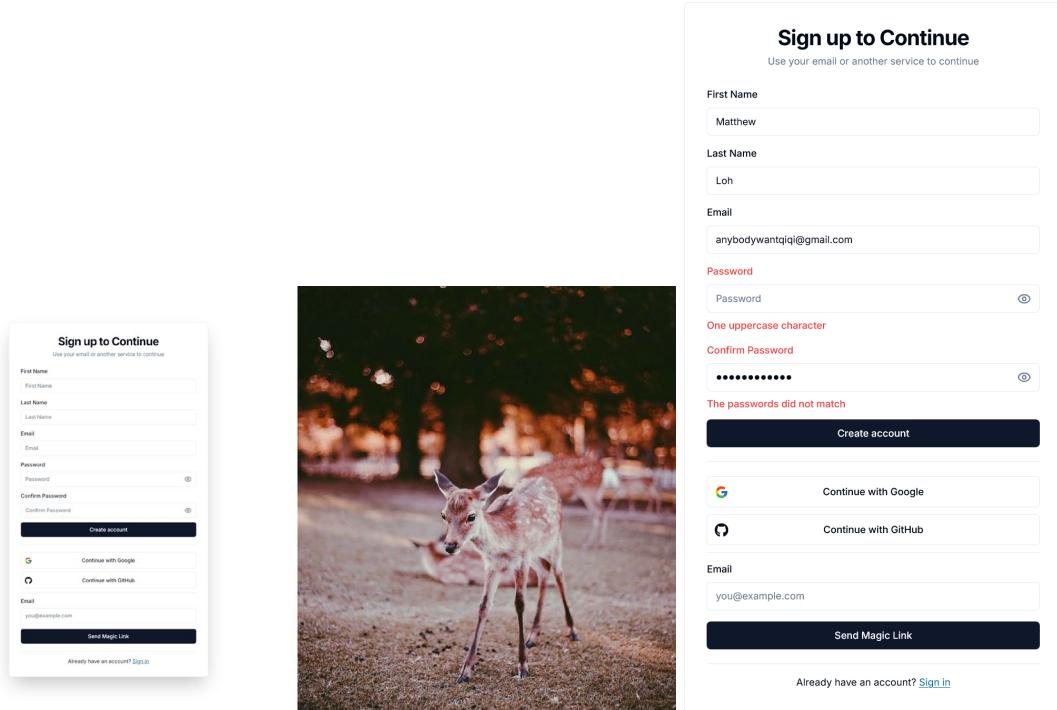


Figure 4.3: Registration Page

The Login and Registration module shown in Figure 4.2 and Figure 4.3 builds the foundation for user authentication and authorization in the Pragmatic platform. This module of the system leverages Supabase Auth, integrated through its Next.js-tailored `@supabase-js` and

@supabase/ssr JavaScript/TypeScript libraries, to provide a highly customizable and extensible authentication system that supports various standard authentication methods while ensuring best practices security and user session management.

4.2.1 Login and Registration with Supabase - OAuth, Magic Link, Email and Password

The login and registration flows are implemented using Next.js Server Actions, ensuring type-safe handling and proper server-side auth management. The framework offers the ability to implement proper redirect handling for both successful authentication and error cases, maintaining a smooth user experience while ensuring security.

```

pragmatic-v1 - Signinform.tsx
<Form {...form}>
  <form onSubmit={form.handleSubmit(onSubmit)} className="space-y-3">
    {error && (
      <p className="text-center text-destructive">{error}</p>
    )}

    <FormField
      control={form.control}
      name="email"
      render={({ field }) => (
        <FormLabel>
          <FormControl>
            <input
              placeholder="Email"
              type="email"
              {...field}
            />
        </FormControl>
        <FormMessage />
      </FormItem>
    )}
  </FormField>
  <FormField
    control={form.control}
    name="password"
    render={({ field }) => (
      <FormLabel>
        <FormControl>
          <PasswordInput
            placeholder="Password"
            {...field}
          />
        </FormControl>
        <FormMessage />
      </FormItem>
    )}
  </FormField>

  {/* Hidden input value for next */}
  {next && <input type="hidden" name="next" value={next} />}
  <LoadingButton
    loading={isPending}
    type="submit"
    className="w-full"
    size="lg"
  >
    Sign In
  </LoadingButton>
</form>
<Separator />
<div className="flex flex-col gap-y-2.5">
  <Button
    disabled={false}
    onClick={() => {
      handleLoginWithOAuth("google")
    }}
    variant="outline"
    size="lg"
    className="relative w-full"
  >
    <FaGoogle className="absolute left-2.5 top-2.5 size-5" />
    Continue with Google
  </Button>
  <Button
    disabled={false}
    onClick={() => {
      handleLoginWithOAuth("github")
    }}
    variant="outline"
    size="lg"
    className="relative w-full"
  >
    <FaGitHub className="absolute left-2.5 top-2.5 size-5" />
    Continue with GitHub
  </Button>
</div>
<Separator />
<SignInMagicLink />
</Form>

```

```

pragmatic-v1 - auth-actions.ts
export async function login({
  credentials,
  next,
}: {
  credentials: LoginValues
  next: string
}): Promise<{ error: string }> {
  const supabase = await createClient()
  const { email, password } = loginSchema.parse(credentials)
  // try {
  // } catch (error) {}
  // type-casting here for convenience
  // in practice, you should validate your inputs
  const data = {
    email: email,
    password: password,
  }
  console.log(next)
  const { error } = await supabase.auth.signInWithEmailAndPassword(data)

  if (error) {
    if (isAuthApiError(error)) {
      return { error: "Invalid credentials entered." }
    }
    console.log(error)
  }

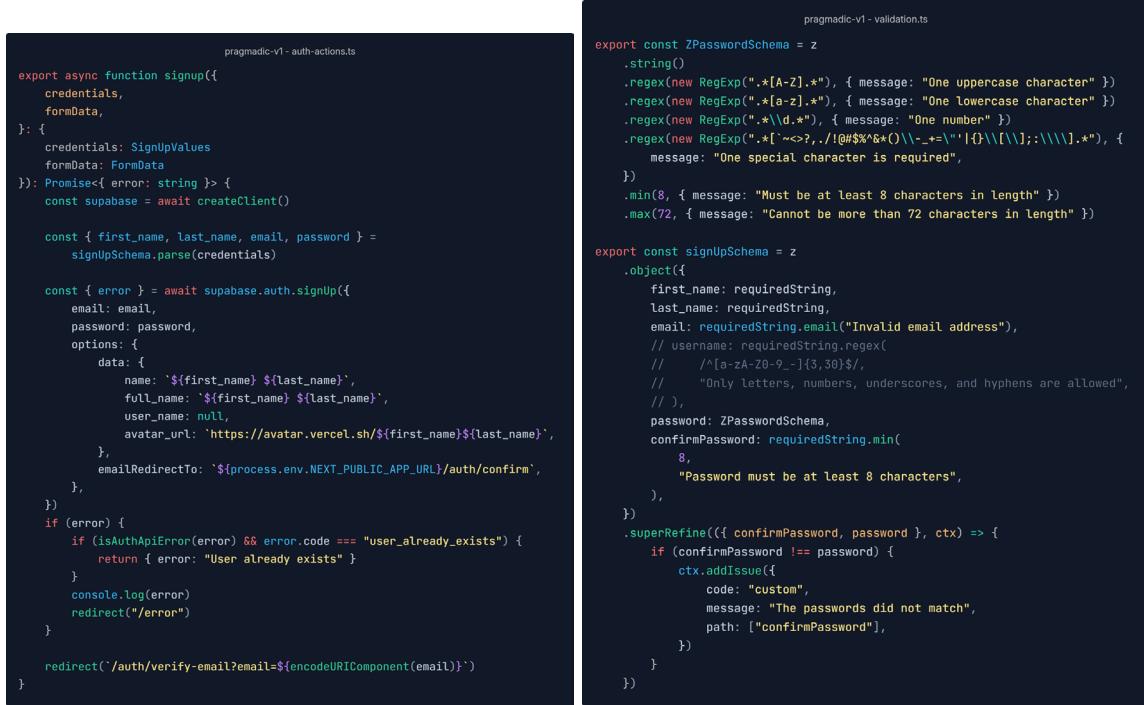
  revalidatePath("/")
  if (next) {
    redirect(`.${next}`)
  } else {
    redirect("/")
  }
}

pragmatic-v1 - validation.ts
export const loginSchema = z.object({
  email: requiredString.email("Invalid email address"),
  password: requiredString,
})

```

Figure 4.4: auth-actions.ts: login function uses signInWithEmailAndPassword from supabase.auth

The email/password authentication process shown in Figure 4.4 involves the login function in auth-actions.ts, which handles user login by validating credentials using a Zod schema. It utilizes Supabase's `signInWithPassword` method to authenticate users. Upon successful login, users are redirected to the specified path or the homepage.



```

Pragmatic-v1 - auth-actions.ts
export async function signup({
  credentials,
  formData,
}: {
  credentials: SignUpValues
  formData: FormData
}): Promise<{ error: string }> {
  const supabase = await createClient()

  const { first_name, last_name, email, password } =
    signUpSchema.parse(credentials)

  const { error } = await supabase.auth.signUp({
    email: email,
    password: password,
    options: {
      data: {
        name: `${first_name} ${last_name}`,
        full_name: `${first_name} ${last_name}`,
        user_name: null,
        avatar_url: `https://avatar.vercel.sh/${first_name}${last_name}`,
      },
      emailRedirectTo: `${process.env.NEXT_PUBLIC_APP_URL}/auth/confirm`,
    },
  })
  if (error) {
    if (isAuthApiError(error) && error.code === "user_already_exists") {
      return { error: "User already exists" }
    }
    console.log(error)
    redirect("/error")
  }
  redirect('/auth/verify-email?email=${encodeURIComponent(email)})'
}
}

pragmatic-v1 - validation.ts
export const ZPasswordSchema = z
  .string()
  .regex(new RegExp(".*[A-Z].*"), { message: "One uppercase character" })
  .regex(new RegExp(".*[a-z].*"), { message: "One lowercase character" })
  .regex(new RegExp(".*\d.*"), { message: "One number" })
  .regex(new RegExp(".*[^<>?,./!@#$%^&()\\-+=\"!|{}\\[\\]\\:\\\\\\].*"), {
    message: "One special character is required",
  })
  .min(8, { message: "Must be at least 8 characters in length" })
  .max(72, { message: "Cannot be more than 72 characters in length" })

export const signUpSchema = z
  .object({
    first_name: requiredString,
    last_name: requiredString,
    email: requiredString.email("Invalid email address"),
    // username: requiredString.regex(
    //   /^[a-zA-Z0-9-_]{3,30}$/,
    //   "Only letters, numbers, underscores, and hyphens are allowed",
    // ),
    password: ZPasswordSchema,
    confirmPassword: requiredString.min(
      8,
      "Password must be at least 8 characters",
    ),
  })
  .superRefine(({ confirmPassword, password }, ctx) => {
    if (confirmPassword !== password) {
      ctx.addIssue({
        code: "custom",
        message: "The passwords did not match",
        path: ["confirmPassword"],
      })
    }
  })
}

```

Figure 4.5: auth-actions.ts signup Server Action for registration by Email and Password

For registration, the `signup` function in `auth-actions.ts` seen in Figure 4.5 manages user registration by validating input data and using Supabase's `signUp` method to create a new user account. This function also sets user metadata and handles email verification redirects. Client-side validation is implemented on both ends for the server-side by using react-hook-form and Zod schemas to validate form inputs from the user to avoid common form pitfalls when developing in the browser such as weak passwords. In Supabase, various authentication providers can be easily integrated. Pragmatic uses Email verification which is hooked up to Resend as an SMTP provider. Otherwise, it also supports social logins through OAuth with GitHub and Google login.

```

pragmatic-v1 - SigninMagicLink.tsx

async function onSubmit(values: z.infer<typeof formSchema>) {
  setIsLoading(true)

  try {
    const supabase = createClient()
    const { error } = await supabase.auth.signInWithOtp({
      email: values.email,
      options: {
        emailRedirectTo: `${getURL()}/auth/confirm`,
      },
    })

    if (error) {
      throw error
    }

    toast.success("Check your email for the magic link!")
    form.reset()
  } catch (error) {
    toast.error("Failed to send magic link. Please try again.")
    console.error("Magic link error:", error)
  } finally {
    setIsLoading(false)
  }
}

pragmatic-v1 - SignInForm.tsx

const handleLoginWithOAuth = async (provider: "github" | "google") => {
  const supabase = createClient()
  toast.info("Signing in...", {
    description: `Signing you in into Pragmatic`,
    duration: 3000,
    icon: provider === "google" ? <FcGoogle /> : <FaGithub />,
  })
  const { error } = await supabase.auth.signInWithOAuth({
    provider,
    options: {
      redirectTo: getURL() + "auth/callback?next=" + next,
    },
  })
  if (error) {
    toast.error("Error signing in with " + provider, {
      description: error.message,
    })
  }
}

pragmatic-v1 - route.ts

export async function GET(request: NextRequest) {
  const { searchParams } = new URL(request.url)
  const token_hash = searchParams.get("token_hash")
  const type = searchParams.get("type") as EmailOtpType | null
  const next = searchParams.get("next") ?? "/"
  const redirectTo = request.nextUrl.clone()
  redirectTo.pathname = next

  if (token_hash && type) {
    const supabase = await createClient()

    const { error } = await supabase.auth.verifyOtp({
      type,
      token_hash,
    })

    if (!error) {
      return NextResponse.redirect(redirectTo)
    }
  }

  // For error cases, redirect to error page
  const errorRedirect = request.nextUrl.clone()
  errorRedirect.pathname = "/auth/auth-code-error"

  return NextResponse.redirect(errorRedirect)
}

```

Figure 4.6: Magic Link and OAuth logins use API routes for Proof Key for Code Exchange (PKCE) flows

Figure 4.6 is the client side and server side code that allows the usage of social login providers and also magic link sign up, which allows users to create an account without needing to manage login credentials. Supabase Email/Password utilizes bcrypt to hash passwords.

Table Editor

Viewing protected schema
This schema is managed by Supabase and is read-only through the table editor.

Search tables... ▼

schema: auth

Filter Sort

id	encrypted_password	email_confirmed_at	invited_at	confirmation_token	conf
2d7c06cc-a575-45ba-b8ae-b0c762e35bb	NULL	2024-11-03 19:05:34.853773+00	NULL	EMPTY	NUL
363d45e4-21e6-4a62-a2a8-87ca06b7eb9	\$2a\$10\$c0UfwxxEej.SST38lUt9t.O44Me.	2024-11-06 08:35:24.039478+00	NULL	EMPTY	2024
4d580477-6f19-45af-b3e4-1d66f413d0e3	\$2a\$10\$c0UQWaZOjihMdMc4ppg0Tunr	NULL	NULL	pkce_0c8cb0858ee9da7b748e85f4c2ae2f	2024
d18de68d-7a26-42c5-bc40-0442e39290f	NULL	2024-11-05 15:57:18.852325+00	NULL	EMPTY	NUL
d2192fb-fad3-436a-a0e3-fb4719ca9bab	\$2a\$10\$YXDgjOr3KxqgrNt/03.EsYIiUU	2024-11-06 12:40:30.055468+00	NULL	EMPTY	2024
fa6742c3-b2f0-45f0-b3e7-6468a6fd2542	\$2a\$10\$DcG2lM1HVC.yAw9fz4Dh.utExgl	2024-11-06 10:21:34.614615+00	NULL	EMPTY	2024
fc076d78-ce49-48e6-9f3a-5db4a524487c	NULL	2024-11-03 14:20:09.355185+00	NULL	EMPTY	NUL

Page 1 of 1 100 rows 7 records

Refresh Data Definition

Users

Search email, phone or UID All users Provider All columns Sorted by created at

Refresh Add user

UID	Display name	Email	Phone	Providers	Provider type	Created at	Last sign in at
fa6742c3-b2f0-45f0-b3e7-6468a6fd2542	-	gi[REDACTED]	-	✉ Email	-	Wed 06 Nov 2024 18:16:19 GMT+0800	Wed 06 Nov 2024 18:21:34 GMT+0800
363d45e4-21e6-4a62-a2a8-87ca06b7eb95	-	gi[REDACTED]	-	✉ Email	-	Wed 06 Nov 2024 16:35:02 GMT+0800	Wed 06 Nov 2024 16:38:24 GMT+0800
d18de68d-7a26-42c5-bc40-0442e39290f	Matthew Loh Yet Man	p[REDACTED]nti.edu.my	-	Google	Social	Tue 05 Nov 2024 23:57:18 GMT+0800	Tue 05 Nov 2024 23:57:19 GMT+0800
2d7c06cc-a575-45ba-b8ae-b0c762e35bbe	anybody want diluc lol	a[REDACTED].ll.com	-	Google	Social	Mon 04 Nov 2024 03:05:34 GMT+0800	Mon 04 Nov 2024 03:05:35 GMT+0800
fc076d78-ce49-48e6-9f3a-5db4a5244870	Matthew Loh	m[REDACTED].om	-	Google	Social	Sun 03 Nov 2024 22:20:09 GMT+0800	Wed 06 Nov 2024 18:36:45 GMT+0800

Figure 4.7: Users are successfully created into the Supabase instance using OAuth and Magic Link

4.2.2 Role-based Access Control (RBAC) Implementation using Custom Claims

```

pragmatic-v1 - 2024105203801_remote_schema.sql

CREATE OR REPLACE FUNCTION public.handle_new_user () RETURNS trigger LANGUAGE plpgsql SECURITY DEFINER
SET search_path TO 'public' AS $function$
BEGIN
    INSERT INTO public.users (id, email, display_name, image_url, roles)
    VALUES (
        new.id,
        new.email,
        COALESCE(
            new.raw_user_meta_data >> 'full_name',
            new.raw_user_meta_data >> 'user_name',
            new.raw_user_meta_data >> 'name',
            '[redacted]'
        ),
        new.raw_user_meta_data >> 'avatar_url',
        CASE
            WHEN new.email = 'matthewloh256@gmail.com' THEN ARRAY['admin'::public.user_role, 'regular'::public.user_role] -- Assign both roles
            ELSE ARRAY['regular'::public.user_role] -- Assign regular role to others
        END
    )
    ON CONFLICT (id) DO UPDATE SET
        email = excluded.email,
        display_name = excluded.display_name,
        image_url = excluded.image_url,
        roles = excluded.roles; -- Update role in case of conflict as well

    -- Insert into user_roles table conditionally
    IF new.email = 'matthewloh256@gmail.com' THEN
        INSERT INTO public.user_roles (user_id, role)
        VALUES (
            new.id,
            'admin'::public.user_role -- Assign admin role
        )
        ON CONFLICT (user_id, role) DO NOTHING; -- Prevent duplicate entries
    ELSE
        INSERT INTO public.user_roles (user_id, role)
        VALUES (
            new.id,
            'regular'::public.user_role -- Assign regular role
        )
        ON CONFLICT (user_id, role) DO NOTHING; -- Prevent duplicate entries
    END IF;
    RETURN new;
END;

```

```

pragmatic-v1 - 20241026125752_storage.sql

CREATE TRIGGER create_user_on_signup
AFTER INSERT ON auth.users FOR EACH ROW
EXECUTE FUNCTION handle_new_user ();

```

Figure 4.8: Pragmatic uses triggers to manage new users across the various possible login and registration verification methods

In Supabase, Postgres functions can be used as triggers on the auth.users table's insert and update operations in order to automatically provide a minimized version of the users created to use in the system itself. This way, safer usage can be provided out of the box. Therefore, RBAC is implemented by using enums to automatically register users with roles using these triggers. In the system, there are four roles, consisting of "regular", "nomad", "owner" and "admin". These are enforced across the stack by using Postgres enums. The custom claims flow, which is to add on custom fields onto the JWT provided by Supabase Auth is integrated in order to help improve performance and provide a RBAC outlet. A possible implementation improvement is to have 2-factor authentication for admin roles. This could be achieved by using verified phone numbers with authenticator services integrated to further facilitate secure admin RBAC.

x Edit 'set_user_role' function

Definition

The language below should be written in `plpgsql`.

```

1 declare
2   claims jsonb;
3   user_email text;
4   user_roles public.user_role[];
5 begin
6   -- Log the entire event object
7   RAISE LOG 'Full event object: %', event;
8
9
10  -- Fetch the user roles in the user_roles table
11  select array_agg(role) into user_roles from public.user_roles where user_id =
12    (event->>'user_id')::uuid;
13
14  -- Get the user's email from the claims object
15  user_email := event->>'claims'->>'email';
16
17  -- Log the user's email
18  RAISE LOG 'User email: %', user_email;
19
20  claims := coalesce(event->>'claims', '{}'::jsonb);
21
22  -- Check if 'app_metadata' exists in claims
23  if jsonb_typeof(claims->>'app_metadata') is null then
24    claims := jsonb_set(claims, '{app_metadata}', '{}');
25  end if;
26
27  -- Set the user_roles in app_metadata
28  if user_roles is not null and array_length(user_roles, 1) > 0 then
29    claims := jsonb_set(claims, '{app_metadata, user_roles}', to_jsonb(user_roles));
30  else
31    claims := jsonb_set(claims, '{app_metadata, user_roles}', '[]'::jsonb);
32  end if;
33
34  -- Update the 'claims' object in the original event
35  event := jsonb_set(event, '{claims}', claims);
36
37  -- Log the final event object
38  RAISE LOG 'Final event object: %', event;
39
40  return event;
41 end;
```

Figure 4.9: The set_user_role function is used to provide custom claims for each JWT

```

pragmatic-v1 - use-user-role.tsx
import { useQuery } from "@tanstack/react-query"
import { AuthError, Session, User } from "@supabase/supabase-js"
import { jwtDecode } from "jwt-decode"
import type { JwtPayload } from "jwt-decode"

import useSupabaseBrowser from "@utils/supabase/client"
import { RoleType } from "@lib/auth/get-user-role"

type SupabaseJwtPayload = JwtPayload & {
    app_metadata: {
        user_roles: RoleType[]
    }
}

type UserRoleData = {
    session: Session | null
    user: User | null
    user_roles: RoleType[]
}

const areRolesEqual = (dbRoles: RoleType[], jwtRoles: RoleType[]): boolean => {
    if (dbRoles.length != jwtRoles.length) return false
    const sortedDbRoles = [...dbRoles].sort()
    const sortedJwtRoles = [...jwtRoles].sort()
    return sortedDbRoles.every((role, index) => role === sortedJwtRoles[index])
}

export function useUserRole() {
    const supabase = useSupabaseBrowser()

    const fetchUserRole = async (): Promise<UserRoleData> => {
        const {
            data: { session },
            error,
        } = await supabase.auth.getSession()
        if (error) throw error

        if (session) {
            const decodedJwt = jwtDecode<SupabaseJwtPayload>(
                session.access_token,
            )
            const { data: user } = await supabase.auth.getUser()

            // Get JWT roles first
            const jwtRoles = decodedJwt.app_metadata.user_roles

            // Only fetch DB roles if we have a user
            if (user.user?.id) {
                const { data: dbRoles } = await supabase
                    .from("user_roles")
                    .select("role")
                    .eq("user_id", user.user.id)

                // If we have DB roles, check if they're different from JWT roles
                if (dbRoles.length) {
                    const dbRoleTypes = dbRoles.map((r) => r.role as RoleType)
                    const userRoles = areRolesEqual(dbRoleTypes, jwtRoles)

                    // Only use DB roles if they're different from JWT roles
                    if (!userRoles) {
                        const dbRoleTypes = dbRoles.map((r) => r.role as RoleType)
                        const jwtRoles = jwtDecode<SupabaseJwtPayload>(
                            session.access_token,
                        ).app_metadata.user_roles
                        const userRoles = areRolesEqual(dbRoleTypes, jwtRoles)
                    }
                }
            }

            // Fallback to JWT roles if no DB roles or no difference
            return {
                session,
                user: user.user,
                user_roles: jwtRoles,
            }
        }
        return { session: null, user: null, user_roles: [] }
    }

    return useQuery<UserRoleData, AuthError>({
        queryKey: ["user-role"],
        queryFn: fetchUserRole,
        staleTime: 1000 * 60 * 5, // Consider data fresh for 5 minutes
    })
}

```

```

pragmatic-v1 - get-user-role.ts
import "server-only"

import { JWTPayload, jwtVerify } from "jose"

import { createClient } from "@utils/supabase/server"
import { ROLES } from "@utils/supabase/permissions"

export type RoleType = (typeof ROLES)[keyof typeof ROLES]

// Extend the JWTPayload type to include Supabase-specific metadata
type SupabaseJwtPayload = JWTPayload & {
    app_metadata: {
        user_roles: RoleType[]
    }
}

export async function getUserRole() {
    // Create a Supabase client for server-side operations
    const supabase = await createClient()

    // Retrieve the current session
    const {
        data: { session },
    } = await supabase.auth.getSession()

    let user_roles

    if (session) {
        // Extract the access token from the session
        const token = session.access_token

        try {
            // Encode the JWT secret for verification
            const secret = new TextEncoder().encode(
                process.env.SUPABASE_JWT_SECRET,
            )

            // Verify the JWT token and extract the payload
            const { payload } = await jwtVerify<SupabaseJwtPayload>(
                token,
                secret,
            )

            // Extract the role from the app_metadata in the payload
            user_roles = payload.app_metadata.user_roles
        } catch (error) {
            console.error("Failed to verify token:", error)
        }
    }

    return { session, user_roles: user_roles as RoleType[] }
}

// Helper function to check if the user has a specific role
export function hasRole(userRoles: RoleType[], requiredRole: RoleType) {
    return userRoles.includes(requiredRole)
}

```

Figure 4.10: Client-side and Server-side functions to decode the JWT through Auth Hooks.

These functions implemented allow viable RBAC checks across the server and client. Notably, server-side environments require an environment variable of SUPABASE_JWT_SECRET in order to decode the automatically minted JWT. Then, roles are returned to users for the system to consume in a predictable, up-to-date and secure fashion by having an array of roles.

4.2.3 Profile Management and Dashboards

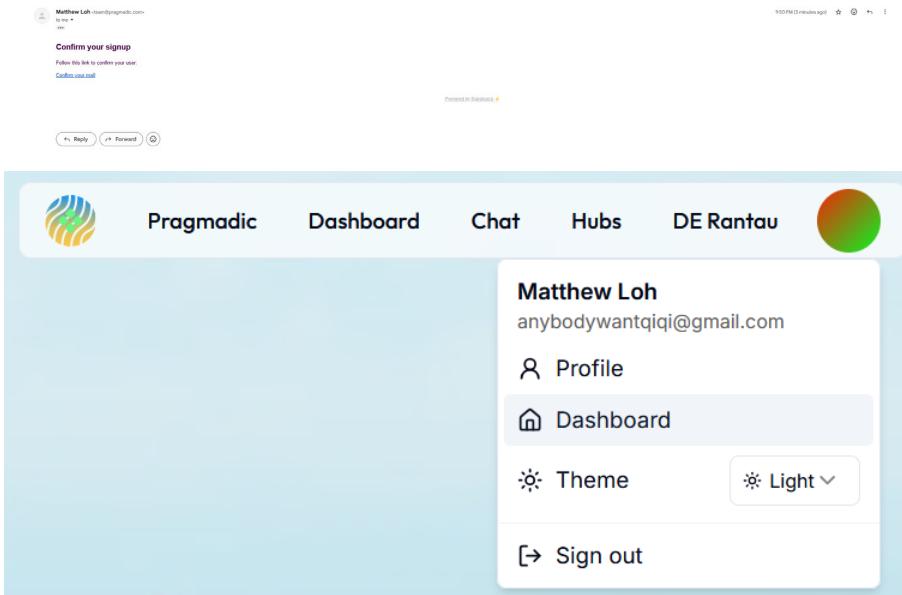


Figure 4.11: New user is successfully created using Email Password sign up.

After successful registration, users can login to the system and be greeted with various dashboards according to specific system roles.

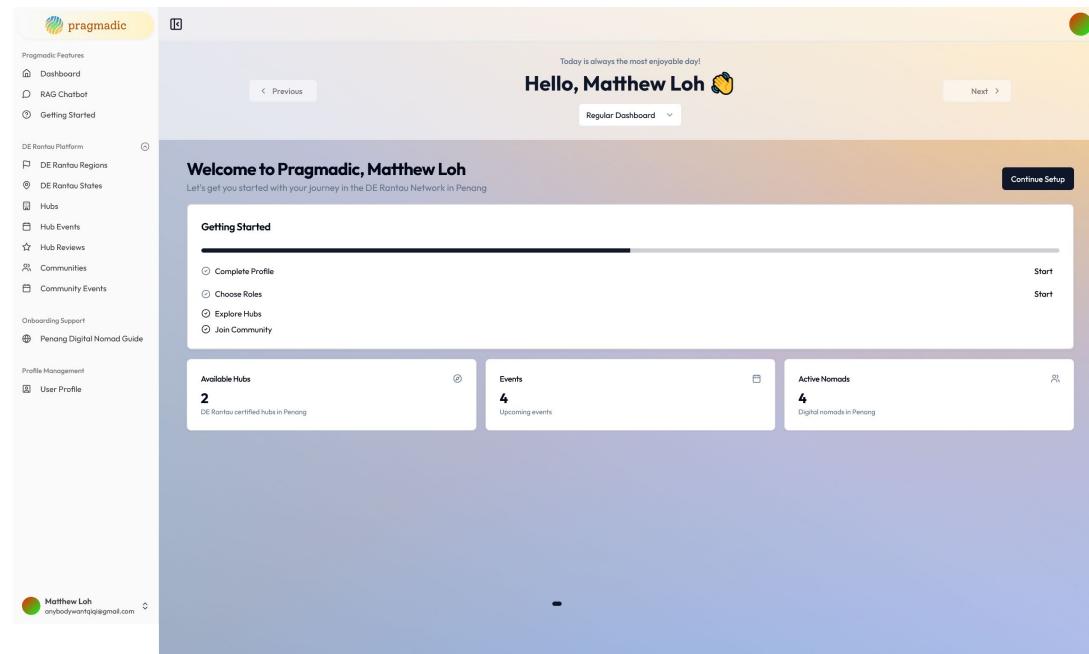


Figure 4.12: Dashboard for Newly Registered User

The screenshot shows the Pragmatic Getting Started page. On the left is a sidebar with navigation links: Pragmatic Features (Dashboard, RAG Chatbot, Getting Started), DE Rantau Platform (DE Rantau Regions, DE Rantau States, Hubs, Hub Events, Hub Reviews, Communities, Community Events), Onboarding Support (Penang Digital Nomad Guide), and Profile Management (User Profile). The main area has a header with three steps: 1. Basic Info (highlighted), 2. Choose Role, 3. Complete Profile. Below the header is a "Welcome to Pragmatic!" section with a placeholder profile picture and a "Change Picture" button. A sub-section titled "Complete Your Profile (Optional)" asks for a bio, occupation, location, website, contact number, and social links. The bio field contains the text "Tell us about yourself". The occupation field is "Occupation" and the location field is "Location". The website field is "Website" and the contact number field is "Contact Number". The social links field is empty. At the bottom are "Forward" and "Continue" buttons.

Figure 4.13: Getting started page for new users to create base profile information

This screenshot shows the same Pragmatic Getting Started page after profile creation. The sidebar and header are identical to Figure 4.13. The main area shows the "Welcome to Pragmatic!" section with a successfully uploaded profile picture of a city skyline at night. The "Change Picture" button is now grayed out. The "Complete Your Profile (Optional)" section shows the updated bio: "Full-stack developer passionate about creating accessible web applications. Always learning and sharing knowledge with the community. Coffee enthusiast and occasional rock climber. ☕️". The occupation is "Senior Full-Stack Developer" and the location is "San Francisco, CA". The website is "https://matthewloh.com" and the contact number is "+6016-334-5567". The social links field contains the URL "https://twitter.com/matthewloh1". At the bottom, there is a success message: "Profile saved successfully!" and the "Forward" and "Continue" buttons.

Figure 4.14: Successful Profile Creation and Profile Picture Uploaded

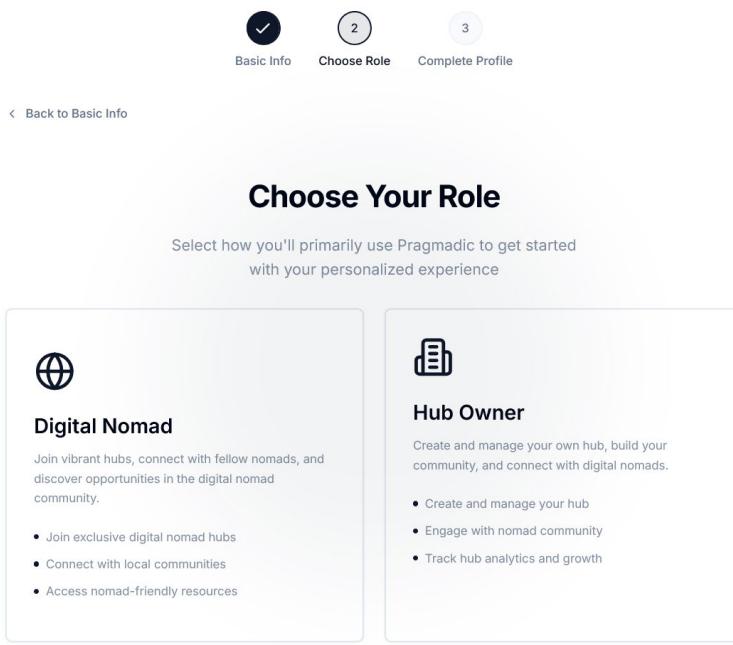


Figure 4.15: Role Selector

The screenshot displays the Pragmatic app's user interface for creating a nomad profile. It includes a sidebar with navigation links like Dashboard, RAG Chatbot, Getting Started, DE Rantau Platform, Hub Events, Hub Reviews, Communities, Community Events, Penang Digital Nomad Guide, User Profile, and Nomad Profile. The main area shows three steps: Basic Info (checkmark), Choose Role (checkmark), and Complete Profile (circle with '3').

Step 1: Basic Info

Back to Role Selection

Complete Your Nomad Profile
Tell us about yourself to connect with the right community

About You
Share your story, work experience, and what brings you to Penang...

Skills: Web Development, Design, Marketing...
Interests: Travel, Photography, Local Culture...

Current Location: City, Country
Contact Information: @username or https://linkedin.com/in/...

Complete Setup

Step 2: Choose Your Role

Back to Basic Info

We found your existing Digital Nomad profile!

Select how you'll primarily use Pragmatic to get started with your personalized experience

Digital Nomad (highlighted with a green border)
Join vibrant hubs, connect with fellow nomads, and discover opportunities in the digital nomad community.

- Join exclusive digital nomad hubs
- Connect with local communities
- Access nomad-friendly resources

Hub Owner
Create and manage your own hub, build your community, and connect with digital nomads.

- Create and manage your hub
- Engage with nomad community
- Track hub analytics and growth

Click to edit your Digital Nomad profile

You can always change your role or add additional roles later

Step 3: Complete Profile

Back to Role Selection

Update Your Nomad Profile
Review and update your profile information

About You
Full-stack developer with a background in AI. Penang's tech ecosystem has drawn me here, and I'm looking forward to collaborating with local tech enthusiasts.

Skills: JavaScript, Python, Machine Learning
Interests: Coding, AI, Entrepreneurship

Current Location: Georgetown, Penang
Contact Information: @al_dev

Update Profile

Figure 4.16: Create Nomad Profile and Obtain Nomad Role

Figure 4.17: Create Hub and Obtain Hub Owner Role

Users in Pragmatic are able to have access to creations of hubs, and hub event participation by creating profiles in respective modules. To do this, a fine-tuned and user-friendly flow is provided to the users that guides users throughout the profile creation process. Figure 4.15, Figure 4.14, Figure 4.16 and Figure 4.17 show the processes of the getting-started page. The sidebar component also renders the links for the respective role-specific functions. For example, only hub owners have access to the Analytics module.

```

pragmatic-v1 - page.tsx

// Get user profile data
const userProfile = await db.query.users.findFirst({
    where: eq(users.id, session.user.id),
})

// Get all profile types
const [baseProfile, nomadProfile, ownerProfile] = await Promise.all([
    db.query.profile.findFirst({
        where: eq(profileTable.userId, session.user.id),
    }),
    db.query.nomadProfile.findFirst({
        where: eq(nomadProfileTable.userId, session.user.id),
    }),
    db.query.hubOwnerProfiles.findFirst({
        where: eq(hubOwnerProfiles.userId, session.user.id),
    }),
])
const user_role_data = await db.query.userRoles.findMany({
    where: eq(userRoles.userId, session.user.id),
})

if (!userProfile) {
    notFound()
}

// Check roles and profile completion status
const hasNomadRole = user_role_data.some((role) => role.role === "nomad")
const hasOwnerRole = user_role_data.some((role) => role.role === "owner")

// Profile completion checks
const hasBaseProfile = baseProfile !== undefined
const hasNomadProfile = nomadProfile !== undefined
const hasOwnerProfile = ownerProfile !== undefined
// Combined status for each role
const nomadStatus = {
    hasRole: hasNomadRole,
    hasProfile: hasNomadProfile,
    needsProfile: hasNomadRole && !hasNomadProfile,
}

const ownerStatus = {
    hasRole: hasOwnerRole,
    hasProfile: hasOwnerProfile,
    needsProfile: hasOwnerRole && !hasOwnerProfile,
}

// Helper functions to determine step states
const isReset = searchParams.reset === "true"
const isBasicInfoIncomplete =
    userProfile.display_name === "[redacted]" || !userProfile.image_url
const hasBasicInfo =
    userProfile.display_name !== "[redacted]" && !userProfile.image_url
const hasSelectedRole = !searchParams.nomad || !searchParams.owner
const isSelectingRole = !hasSelectedRole && hasBasicInfo && !isReset

```

Figure 4.18: Code snippet for rendering profiles and roles of user.

In Figure 4.18, the code snippet is an extract from the getting-started page's server component, which fetches data and handles state. The page is able to render the RoleSelector, GettingStartedNomadProfileForm and GettingStartedHubOwnerProfile form. Then, the form components utilize server actions to have better developer experience when developing complex form schemas for managing specific profiles.

```

pragmatic-v1 - NomadProfileForm.tsx
const {
  data: existingProfile,
  isPending,
  refetch,
} = useQuery<NomadProfileRow>(
  supabase
    .from("nomad_profile")
    .select("*")
    .eq("user_id", userId)
    .single(),
  {
    // Disable automatic refetching
    refetchOnWindowFocus: false,
    refetchOnReconnect: false,
    refetchOnMount: false,
  },
)

const form = useForm({
  resolver: zodResolver(insertNomadProfileParams),
  defaultValues: {
    bio: "",
    skills: "",
    interests: "",
    currentlocation: "",
    contactinformation: "",
  },
})

// Update form when existing profile is loaded
useEffect(() => {
  if (existingProfile) {
    form.reset({
      bio: existingProfile.bio || "",
      skills: existingProfile.skills || "",
      interests: existingProfile.interests || "",
      currentlocation: existingProfile.current_location || "",
      contactinformation: existingProfile.contact_information || "",
    })
  }
}, [existingProfile, form])

async function onSubmit(data: z.infer<typeof insertNomadProfileParams>) {
  startTransition(async () => {
    try {
      const profileError = await createNomadProfileAction({
        ...data,
      })

      if (profileError) {
        toast.error(profileError)
        return
      }

      const roleError = await assignUserRoleAction({
        userId,
        role: "nomad",
      })

      if (roleError) {
        toast.error(roleError)
        return
      }

      toast.success(
        existingProfile
          ? "Profile updated successfully!"
          : "Profile created successfully!",
      )
      refetch()
      queryClient.invalidateQueries({ queryKey: ["user-role"] })
    } catch (error) {
      toast.error("Something went wrong")
      console.error(error)
    }
  })
}
}

pragmatic-v1 - HubOwnerProfileForm.tsx
export function GettingStartedOwnerProfileForm({
  userId,
}: GettingStartedOwnerProfileFormProps) {
  const hubOwner = useHubOwner(userId)
  const pending = startTransition() & useTransition()
  const queryClient = useQueryClient()
  // Fetch existing profile
  const {
    data: existingProfile,
    isloading,
    refetch,
  } = useQuery<HubOwnerProfileRow>(
    supabase
      .from("hub_owner_profiles")
      .select("*")
      .eq("user_id", userId)
      .single(),
  )

  const form = useForm({
    resolver: zodResolver(insertHubOwnerProfileParams),
    defaultValues: {
      companyName: "",
      businessRegistrationNumber: "",
      bio: "",
      businessContactNo: "",
      businessEmail: "",
      businessLocation: "",
      residingLocation: "",
      socialMediaHandles: "",
      websiteUrl: "",
    },
  })

  // Update form when existing profile is loaded
  useEffect(() => {
    if (existingProfile) {
      form.reset({
        companyName: existingProfile.company_name || "",
        business_registration_number: existingProfile.business_registration_number || "",
        bio: existingProfile.bio || "",
        business_contact_no: existingProfile.business_contact_no || "",
        business_email: existingProfile.business_email || "",
        business_location: existingProfile.business_location || "",
        residing_location: existingProfile.residing_location || "",
        social_media_handles: existingProfile.social_media_handles || "",
        website_url: existingProfile.website_url || "",
      })
    }
  }, [existingProfile, form])

  async function onSubmit(data: z.infer<typeof insertHubOwnerProfileParams>) {
    startTransition(async () => {
      try {
        const profileError = await createHubOwnerProfileAction({
          ...data,
        })

        if (profileError) {
          toast.error(profileError)
          return
        }

        // Only assign role if it's a new profile
        const roleError = await assignUserRoleAction({
          userId,
          role: "owner",
        })
        queryClient.invalidateQueries({ queryKey: ["user-role"] })

        if (roleError) {
          toast.error(roleError)
          return
        }

        toast.success(
          existingProfile
            ? "Profile updated successfully!"
            : "Profile created successfully!",
        )
        refetch()
      } catch (error) {
        toast.error("Something went wrong")
        console.error(error)
      }
    })
  }
}

```

Figure 4.19: Forms for nomad and hub owner utilize React Query for client-side asynchronous state management

In the React and web development world, React Query, now named Tanstack Query, remains ubiquitous to reduce the complexity of managing client-side data fetching for both rich user and developer experiences, handling caching, loading states, refetches and ensuring data is always fresh on the frontend. The configurability of the APIs provided means that the backend also does not have to be queried in such a way that the performance is degraded for the user. By integrating the Supabase Cache Helpers library together that provides helper query keys for the supabase-js client data fetching, the experience for developing is smooth and handled with best practices and full type safety.

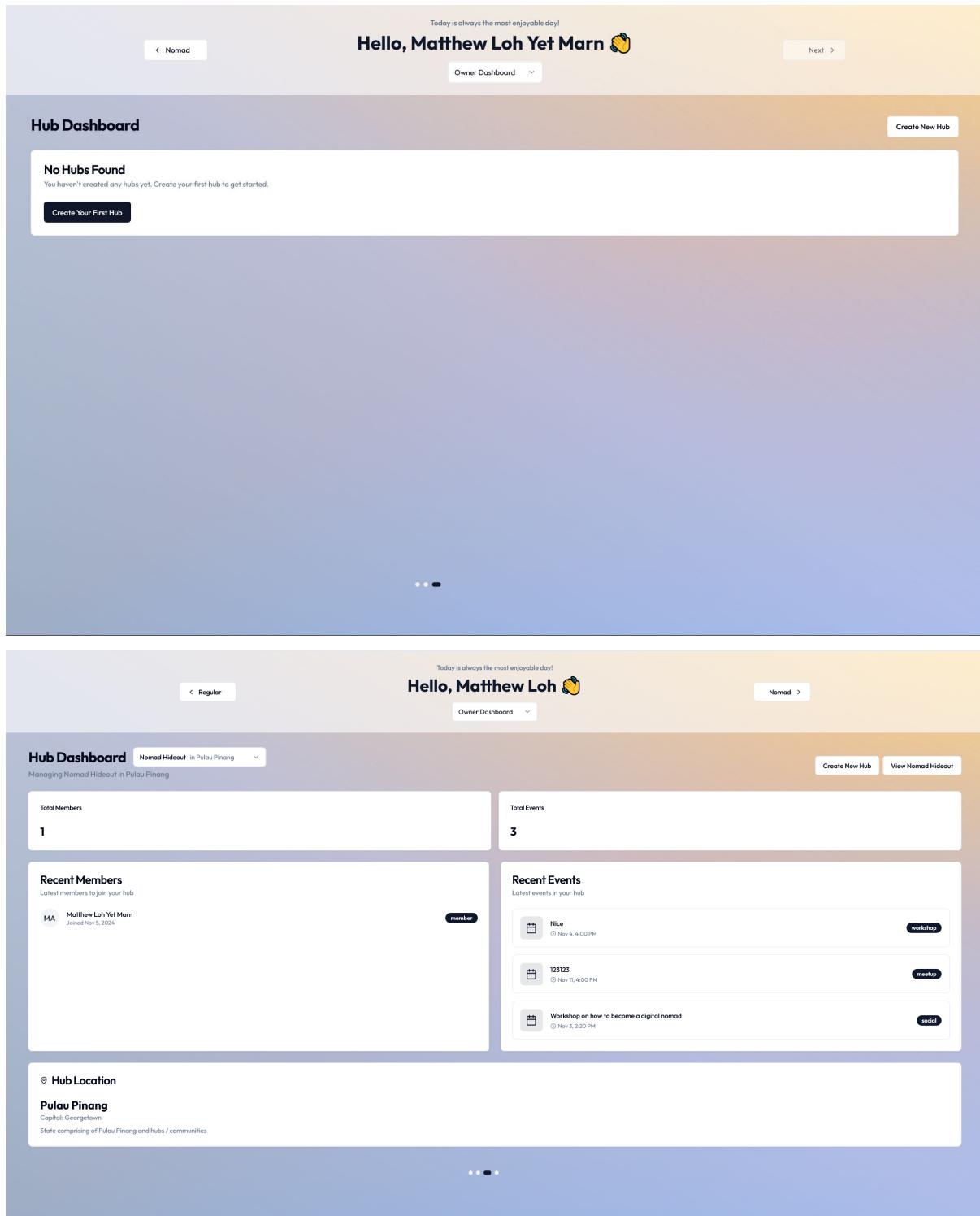


Figure 4.20: Hub Dashboard (Unpopulated and Populated) for Hub Owner

An important part of a good user experience in Pragmatic is the inclusion of rich client-side data fetching dashboards. Here, a quick glance for owners of hubs is provided such as in Figure 4.20 where owners are able to view details for hubs based on membership and event activity.

```


```

 "pragmas": {
 "version": "0.4"
 },
 "name": "DashboardSwitcher",
 "description": "A component that allows users to switch between different dashboard roles by dragging them horizontally. It includes state management for current role index, user roles, and container width, and handles touch events to update the role index and trigger role changes. It also includes a role indicator at the bottom showing the current role index and the total number of roles."}
}

export default function DashboardSwitcher({ user }) {
 const { data, userRoleData, sidebarKey } = user.userInfo

 // State
 const currentRoleIndex = useState(0)
 const containerWidth = useState(0)

 // Refs
 const containerRef = useRef<HTMLDivElement>()

 // Get roles from userRoleData with type safety
 const userRoles = userRoleData[user.roles] || []

 // Derived values from userRoleData
 const currentRole = userRoleData[currentRoleIndex]
 const lastRole = userRoleData[userRoles.length - 1]

 // Set current role index
 const setContainerWidth = (width) => {
 const containerWidthRef = useRef<HTMLDivElement>(null)
 const containerRef = useRef<HTMLDivElement>(null)

 if (width < 0) {
 containerRef.current.style.width = `calc(100% + ${width}px)`
 } else {
 containerRef.current.style.width = `calc(100% - ${width}px)`
 }

 containerWidthRef.current.style.width = width + 'px'
 }

 // Effects
 useEffect(() => {
 const updateContainerWidth = () => {
 if (containerRef.current) {
 setContainerWidth(containerRef.current.offsetWidth)
 }
 }

 window.addEventListener('resize', updateContainerWidth)
 return () => window.removeEventListener('resize', updateContainerWidth)
 }, [currentRoleIndex])

 // Set initial role when data loads
 const init = () => {
 if (userRoleData.user.roles.length) {
 // Default to first role in the list
 setCurrentRoleIndex(0)
 }
 }

 // Handlers
 const handleRoleChange = (value: string) => {
 const newIndex = userRoles.findIndex(role => role === value)
 if (newIndex === -1) {
 setCurrentRoleIndex(nextIndex)
 }
 }

 const handleDragEnd = (e: TouchEvent | PointerEvent, info: PointerInfo) => {
 if (info.offset.x > 300 || info.offset.x < -300) {
 setContainerWidth(currentRoleIndex + 1)
 } else if (info.offset.x > -100 & info.offset.x < 100) {
 setContainerWidth(currentRoleIndex - 1)
 }
 }

 // Render
 return (
 <div className="relative min-h-screen w-full" ref={containerRef}>
 <div className="absolute top-0 left-0 w-full h-full bg-white z-10">
 <div className="flex justify-between items-center px-4 py-4 w-full">
 <div style={{ flex: 1 }}>
 <button variant="outline" onClick={() => handleRoleChange('previous')}>Previous</button>
 <button variant="outline" onClick={() => handleRoleChange('next')}>Next</button>
 </div>
 <div style={{ flex: 1 }}>
 <div style={{ display: 'flex', gap: 10, align-items: 'center' }}>
 <div style={{ flex: 1 }}>
 <div style={{ display: 'flex', gap: 10, align-items: 'center' }}>
 <div style={{ flex: 1 }}>
 <div style={{ display: 'flex', gap: 10, align-items: 'center' }}>
 <div style={{ flex: 1 }}>
 <div style={{ display: 'flex', gap: 10, align-items: 'center' }}>
 <div style={{ flex: 1 }}>
 <div style={{ display: 'flex', gap: 10, align-items: 'center' }}>
 <div style={{ flex: 1 }}>
 <div style={{ display: 'flex', gap: 10, align-items: 'center' }}>
 <div style={{ flex: 1 }}>
 <div style={{ display: 'flex', gap: 10, align-items: 'center' }}>
 <div style={{ flex: 1 }}>
 <div style={{ display: 'flex', gap: 10, align-items: 'center' }}>
 <div style={{ flex: 1 }}>
 <div style={{ display: 'flex', gap: 10, align-items: 'center' }}>
 <div style={{ flex: 1 }}>
 <div style={{ display: 'flex', gap: 10, align-items: 'center' }}>
 <div style={{ flex: 1 }}>
 <div style={{ display: 'flex', gap: 10, align-items: 'center' }}>
 <div style={{ flex: 1 }}>
 <div style={{ display: 'flex', gap: 10, align-items: 'center' }}>
 <div style={{ flex: 1 }}>
 <div style={{ display: 'flex', gap: 10, align-items: 'center' }}>
 <div style={{ flex: 1 }}>
 <div style={{ display: 'flex', gap: 10, align-items: 'center' }}>
 <div style={{ flex: 1 }}>
 <div style={{ display: 'flex', gap: 10, align-items: 'center' }}>
 <div style={{ flex: 1 }}>
 <div style={{ display: 'flex', gap: 10, align-items: 'center' }}>
 <div style={{ flex: 1 }}>
 <div style={{ display: 'flex', gap: 10, align-items: 'center' }}>
 <div style={{ flex: 1 }}>
 <div style={{ display: 'flex', gap: 10, align-items: 'center' }}>
 <div style={{ flex: 1 }}>
 <div style={{ display: 'flex', gap: 10, align-items: 'center' }}>
 <div style={{ flex: 1 }}>
 <div style={{ display: 'flex', gap: 10, align-items: 'center' }}>
 <div style={{ flex: 1 }}>
 <div style={{ display: 'flex', gap: 10, align-items: 'center' }}>
 <div style={{ flex: 1 }}>
 <div style={{ display: 'flex', gap: 10, align-items: 'center' }}>
 <div style={{ flex: 1 }}>
 <div style={{ display: 'flex', gap: 10, align-items: 'center' }}>
 <div style={{ flex: 1 }}>
 <div style={{ display: 'flex', gap: 10, align-items: 'center' }}>
 <div style={{ flex: 1 }}>
 <div style={{ display: 'flex', gap: 10, align-items: 'center' }}>
 <div style={{ flex: 1 }}>
 <div style={{ display: 'flex', gap: 10, align-items: 'center' }}>
 <div style={{ flex: 1 }}>
 <div style={{ display: 'flex', gap: 10, align-items: 'center' }}>
 <div style={{ flex: 1 }}>
 <div style={{ display: 'flex', gap: 10, align-items: 'center' }}>
 <div style={{ flex: 1 }}>
 <div style={{ display: 'flex', gap: 10, align-items: 'center' }}>
 <div style={{ flex: 1 }}>
 <div style={{ display: 'flex', gap: 10, align-items: 'center' }}>
 <div style={{ flex: 1 }}>
 <div style={{ display: 'flex', gap: 10, align-items: 'center' }}>
 <div style={{ flex: 1 }}>
 <div style={{ display: 'flex', gap: 10, align-items: 'center' }}>
 <div style={{ flex: 1 }}>
 <div style={{ display: 'flex', gap: 10, align-items: 'center' }}>
 <div style={{ flex: 1 }}>
 <div style={{ display: 'flex', gap: 10, align-items: 'center' }}>
 </div>
 </div>
 </div>
 </div>
 </div>
 </div>
 </div>
 </div>
 </div>
)
}

```


```

Figure 4.21: Code Snippet for DashboardSwitcher

The DashboardSwitcher component dynamically switches between different user dashboards based on the user's role. It utilizes React hooks for state management and Framer Motion for smooth animations during transitions. The component fetches user roles using a custom hook, `useUserRole`, and displays the appropriate dashboard component from a predefined set (`DASHBOARD_COMPONENTS`). Users can navigate between roles using buttons or a dropdown selector. The component also includes responsive design features, such as updating layout on window resize, and provides a skeleton loading state while user data is being fetched.

4.3 Chatbot Module

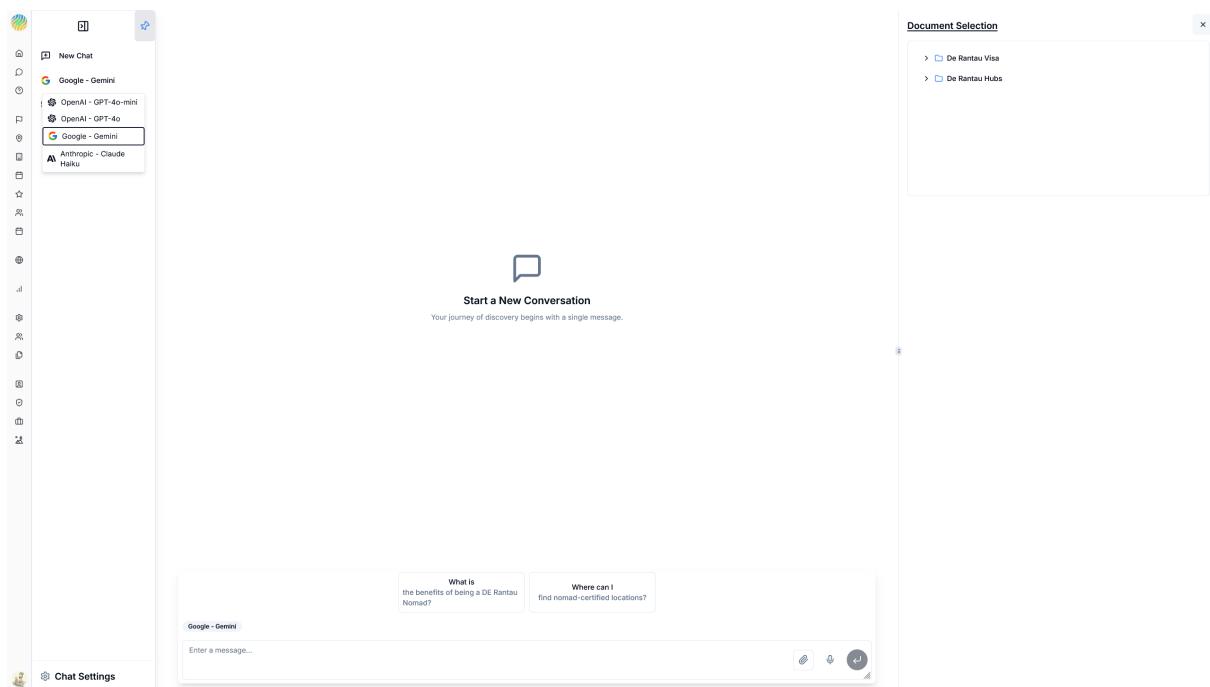


Figure 4.22: RAG-Integrated Answer Engine of Pragmatic

Figure 4.22 shows a chat interface Pragmatic designed to facilitate conversations with a large language model by using context using documents uploaded by admins of the system. The interface includes a chat area, a document selection panel, and a model selector for AI interactions. The sidebar allows the viewing of past chats, creation of new chats and selection of models. The prompt input is multimodal, allowing users to upload files to provide as context to further improve the accuracy of LLM output. The following section details the implementation details behind the entire Chatbot module, starting with the user-facing functionality which can be consumed to answer questions related to DE Rantau program.

4.3.1 Chat Interface

```

pragmatic-v1 ChatContentClient.tsx

export default function ChatContentClient({
  chatId,
  initialMessages,
  ...ChatContentProps
}):
  const [referenceDocs, setReferencedDocs] = useState<ReferencedDocument[]>([
    []
  ])
  const [isDocPanelOpen, setIsDocPanelOpen] = useState(false)
  const [isTransitioning, setIsTransitioning] = useState(false)
  const [isSidebarExpanded, setIsSidebarExpanded] = useState(false)
  const [selectedDocumentIds, setSelectedDocumentIds] = useState<string[]>([])
  const [selectedModel, setSelectedModel] = useState<ModelOption>(
    options.find(opt => opt.model === "gpt-3.5-turbo-0613" || opt.model === "gpt-4")
  )
  useEffect(() => {
    setIsTransitioning(true)
    const timer = setTimeout(() => setIsTransitioning(false), 300)
    return () => clearTimeout(timer)
  }, [isDocPanelOpen])

  const updateReferencedDocs = (newDocs: ReferencedDocument[]) => {
    setReferencedDocs(newDocs)
    if (newDocs.length > 0 && !isDocPanelOpen) {
      setIsDocPanelOpen(true)
    }
  }

  const handleDocumentsSelected = (documentIds: string[]) => {
    setSelectedDocumentIds(documentIds)
  }

  return (
    <>
      <div className="flex h-full flex-col">
        <ResizablePanelGroup
          direction="horizontal"
          className="h-full flex-1">
          <ChatSettingsSidebar
            chatId={chatId}
            isExpanded={isSidebarExpanded}
            selectedModel={selectedModel}
            onModelSelect={setSelectedModel}
            onExpandChange={setIsSidebarExpanded}
          />
          <ResizablePanel
            defaultSize={70}
            minSize={50}
            maxSize={100}
            className="!transition-all duration-300 ease-in-out",
            isTransitioning && "resize-none",
            >
            <div className="container mx-auto h-full bg-background p-0 animate-in">
              <ChatComponent
                chatId={chatId}
                initialMessages={initialMessages}
                onDocumentsReference={updateReferencedDocs}
                isDocPanelOpen={isDocPanelOpen}
                selectedModel={selectedModel}
                selectedDocumentIds={selectedDocumentIds}
              />
              </div>
              <ResizablePanel>
                <ReferenceDocumentsPanel
                  isOpen={isDocPanelOpen}
                  isTransitioning={isTransitioning}
                  documents={referenceDocs}
                  onClose={() => setIsDocPanelOpen(false)}
                  onDocumentsSelected={handleDocumentsSelected}
                  selectedDocuments={selectedDocumentIds}
                />
              </ResizablePanel>
            </div>
            <Button
              className="!fixed bottom-36 right-8 rounded-full p-2 transition-all duration-300 ease-in-out",
              isDocPanelOpen && "pointer-events-none opacity-0",
              onClick={() => setIsDocPanelOpen(true)}
            >
              <Maximize2 className="h-5 w-5" />
            </Button>
          </div>
        </ResizablePanelGroup>
      </div>
    </>
  )
}

```

Figure 4.23: ChatContentClient.tsx

The ChatContentClient component manages the chat's main functionality, including message display and document referencing. It uses ChatComponent to handle message input and display, supporting file uploads and AI model selection. It is a composition of multiple components with rich state being passed as props in a unidirectional flow. Vercel's AI SDK also provides extremely useful UI hooks and types to enable easy integration with tool calling and loading states for AI calls.

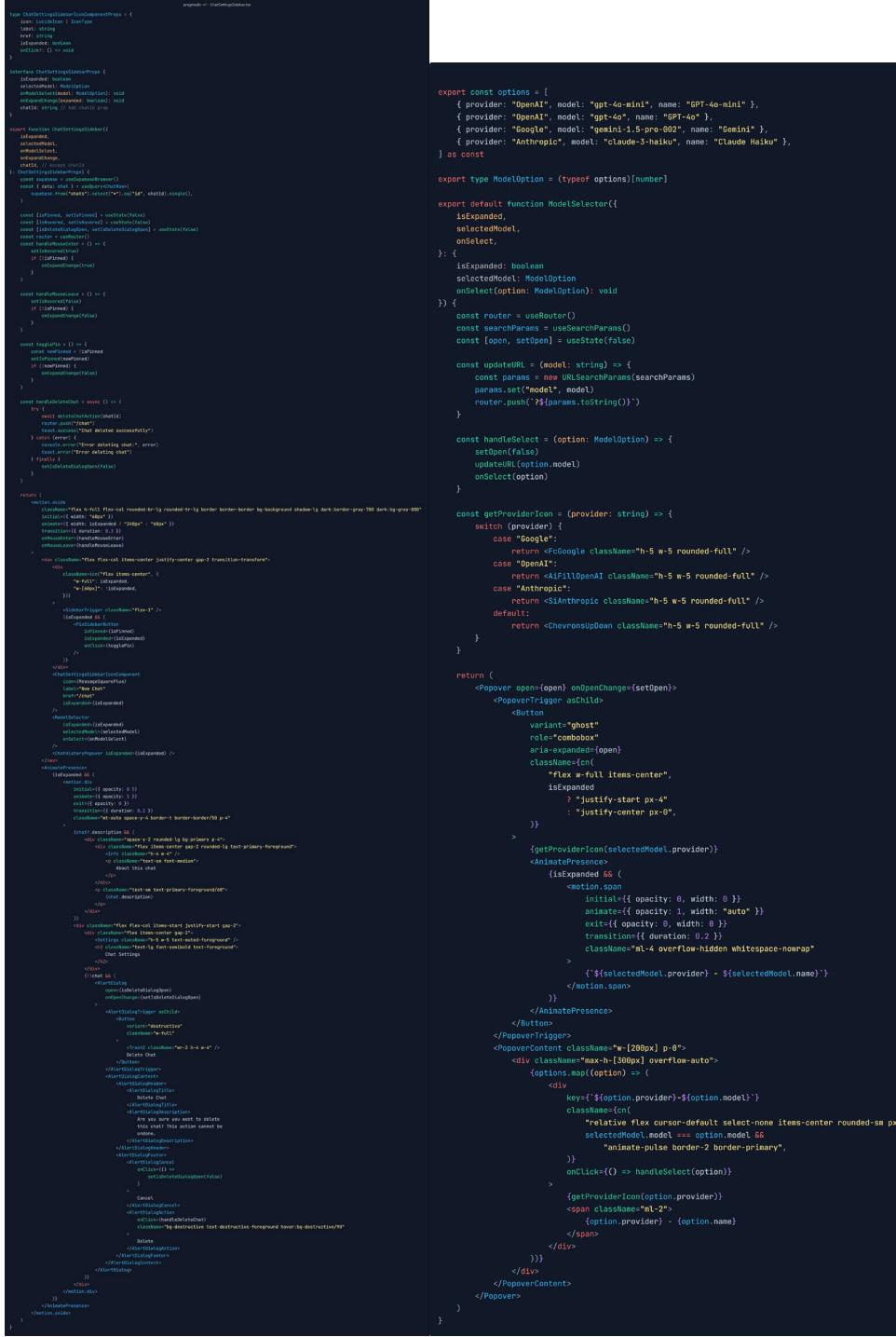


Figure 4.24: Chat Settings Sidebar Component and Model Selector

The ChatSettingsSidebar in Figure 4.24 provides options to select different AI models (e.g., OpenAI, Google, Anthropic) via the ModelSelector component. It also includes chat settings and history management. All the components used are based on shadcn's implementations of common Radix UI based building blocks, ensuring accessibility, composability and easy styling practices.

```

    type DocumentMetadata = {
      id: string;
      title: string | null;
      pageCount: number;
      originalName: string;
    }

    export type DocumentRow = {
      id: string;
      title: string;
      created_at: string;
      updated_at: string;
      metadata: DocumentMetadata;
      name: string;
    }

    interface DocumentSelectorProps {
      selectedDocuments: string[];
      selectionChange: (documents: string[]) => void;
    }

    export function DocumentSelector({ selectedDocuments, selectionChange }: DocumentSelectorProps) {
      const [expandedFolders, setExpandedFolders] = useState<string[]>([]);
      const [newSel, setNewSel] = useState<string>('');

      const documents: DocumentRow[] = [
        {
          id: '1',
          title: 'Document 1',
          created_at: '2023-01-01T12:00:00Z',
          updated_at: '2023-01-01T12:00:00Z',
          metadata: {
            id: '1',
            title: 'Document 1',
            pageCount: 1,
            originalName: 'document1.pdf'
          },
          name: 'document1.pdf'
        },
        {
          id: '2',
          title: 'Document 2',
          created_at: '2023-01-02T12:00:00Z',
          updated_at: '2023-01-02T12:00:00Z',
          metadata: {
            id: '2',
            title: 'Document 2',
            pageCount: 2,
            originalName: 'document2.pdf'
          },
          name: 'document2.pdf'
        },
        {
          id: '3',
          title: 'Document 3',
          created_at: '2023-01-03T12:00:00Z',
          updated_at: '2023-01-03T12:00:00Z',
          metadata: {
            id: '3',
            title: 'Document 3',
            pageCount: 3,
            originalName: 'document3.pdf'
          },
          name: 'document3.pdf'
        }
      ];

      const expandedFoldersState = expandedFolders || [];
      const newSelState = newSel || '';
      const selectedDocumentsState = selectedDocuments || [];

      const expandedFoldersRef = useRef(expandedFoldersState);
      const newSelRef = useRef(newSelState);
      const selectedDocumentsRef = useRef(selectedDocumentsState);

      const toggleFolder = (folder: string) => {
        const expanded = [...expandedFoldersRef.current];
        if (expanded.includes(folder)) {
          expanded.splice(expanded.indexOf(folder), 1);
        } else {
          expanded.push(folder);
        }
        setExpandedFolders(expanded);
      };

      const toggleDocument = (documentId: string) => {
        const selection = [...selectedDocumentsRef.current];
        if (selection.includes(documentId)) {
          selection.splice(selection.indexOf(documentId), 1);
        } else {
          selection.push(documentId);
        }
        setSelectedDocuments(selection);
        selectionChange(selection);
      };

      const renderFolder = (folderName: string, docs: DocumentRow[]) => {
        const expanded = expandedFoldersRef.current;
        const folderNameWithDash = folderName.replace(/\s/g, '-');

        return (
          <div key={folderName} className="space-y-1">
            <button
              variant="ghost"
              size="sm"
              onClick={() => toggleFolder(folderName)}
            >
              <span>{newSelRef.current === folderName ? 'x' : ''}</span>
              <span>{folderName}</span>
            </button>
            <div className="flex items-center">
              {expanded.map((f) => (
                <ChevronRight className="w-2 h-4 w-4" />
                <div className="ml-2 w-4 h-4 text-blue-500" />
                <span className="capitalize">{f}</span>
              ))}
            </div>
            <div className="ml-4 space-y-1 border-l pl-4">
              {docs.map((doc) => (
                <div key={doc.id}>
                  <div>
                    <span>{doc.title}</span>
                    <span>{doc.metadata.originalName || doc.name}</span>
                  </div>
                  <div>
                    {doc.metadata.pageCount > 0 ? (
                      <span>{doc.metadata.pageCount}</span>
                    ) : (
                      <span>No documents found</span>
                    )}
                  </div>
                </div>
              ))}
            </div>
          </div>
        );
      };

      if (!loading) {
        return (
          <div className="p-4 text-center text-muted-foreground">
            {loading ? 'Loading documents...' : ''}
          </div>
        );
      }

      return (
        <div>
          {expandedFolders.length > 0 ? (
            <div className="space-y-2">
              {expandedFolders.map((f) => (
                <span>{f}</span>
              ))}
            </div>
          ) : (
            <div>
              <span>No documents found</span>
            </div>
          )}
        </div>
      );
    }
  
```

Figure 4.25: Document Selector Code

For the document management that is, the selection of relevant documents to question, the DocumentSelector component allows users to browse and select documents, organized by folders. It uses Supabase to fetch document data and supports expanding/collapsing folders for better navigation.

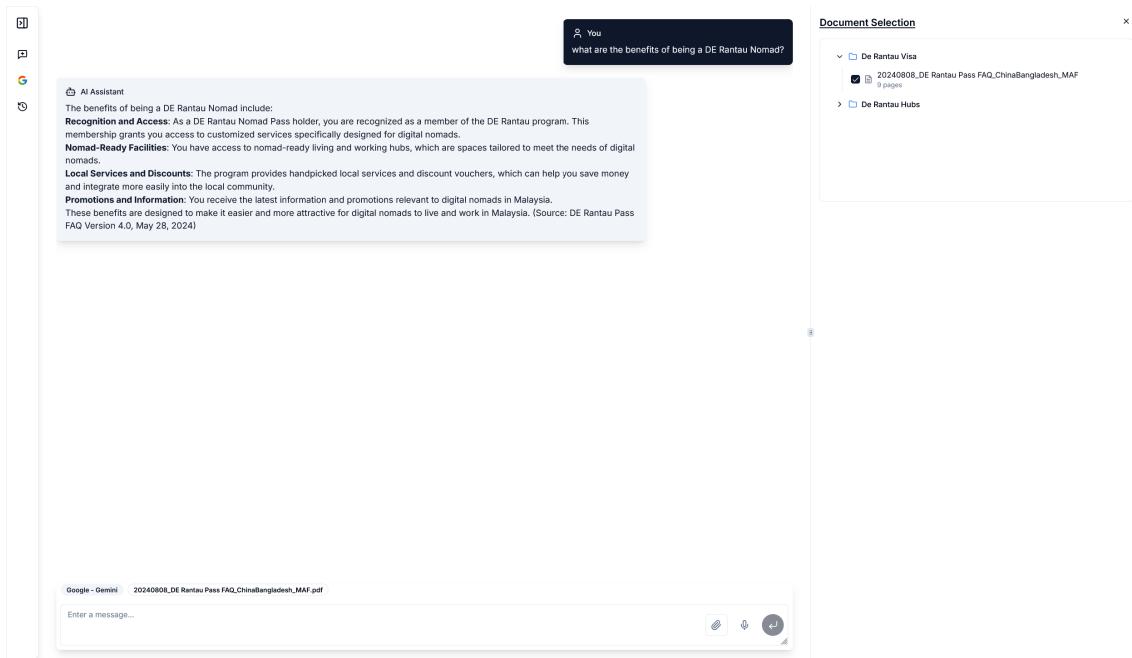


Figure 4.26: Example conversation by referencing a knowledge base document

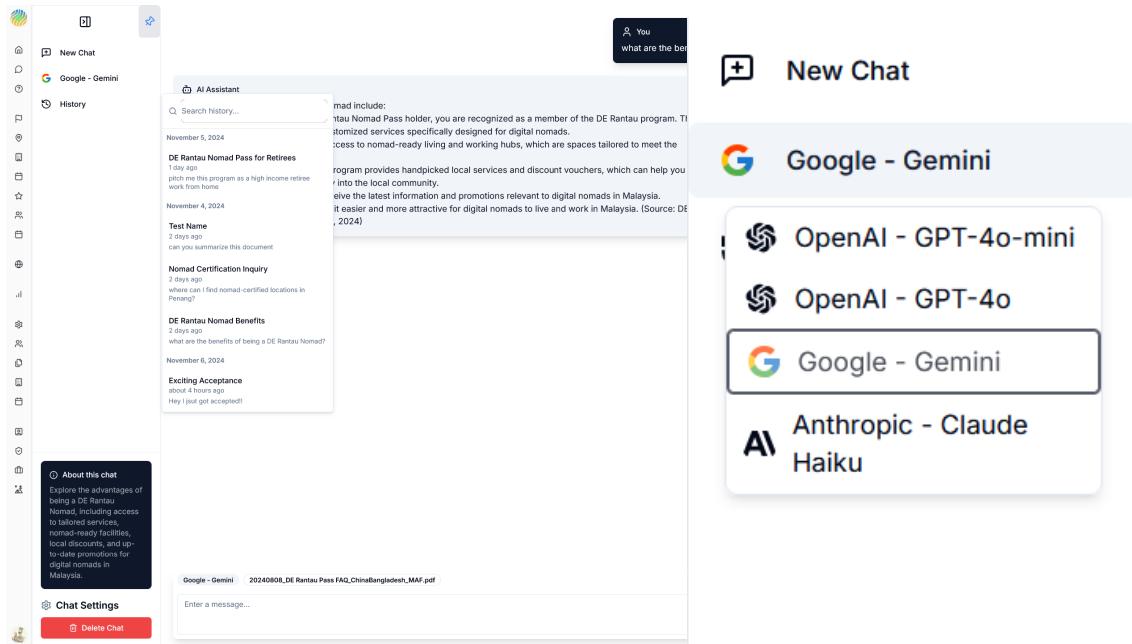


Figure 4.27: Ability to View Past Chats, Delete Chat and Select Different Models

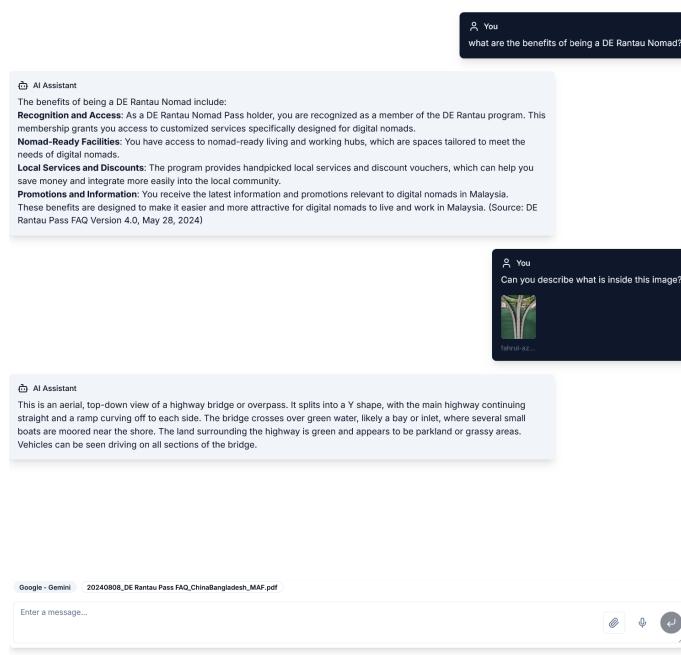


Figure 4.28: Ability to upload images for context

Figure 4.26, Figure 4.27 and Figure 4.28 show the ability for users to start new conversations that support personal files, select different LLM providers seamlessly, and manage chat settings including viewing history and deletion. The interface supports dynamic resizing and animations for smooth transitions between different states. The layout also adapts to different preferred screen sizes, ensuring usability across devices. The ResizablePanelGroup and ResizablePanel components enable flexible resizing of the chat and document panels.

4.3.2 Knowledge Base Management

```

pragmatic-v1 - app-sidebar.tsx
  {userRoleData.user_roles.includes("admin") && (
    <SidebarGroup>
      <SidebarGroupLabel>Admin</SidebarGroupLabel>
      <SidebarMenu>
        {adminOnlyItems.map((item) => (
          <SidebarMenuItem key={item.title}>
            <SidebarMenuItemButton asChild>
              <Link href={item.url}>
                {item.icon && (
                  <item.icon className="mr-2 h-4 w-4" />
                )}
                <span>{item.title}</span>
              </Link>
            </SidebarMenuItemButton>
          </SidebarMenuItem>
        ))}
      </SidebarMenu>
    </SidebarGroup>
  )}

pragmatic-v1 - sidebar-config.ts
export const adminOnlyItems = [
  { title: "Admin Dashboard", url: "/admin", icon: Settings },
  { title: "User Management", url: "/admin/users", icon: Users },
  {
    title: "Knowledge Base Management",
    url: "/admin/chat",
    icon: Files,
  },
]

pragmatic-v1 - middleware.ts
export async function updateSession(request: NextRequest) {
  let supabaseResponse = NextResponse.next({
    request,
  })

  const supabase = createServerClient(
    process.env.NEXT_PUBLIC_SUPABASE_URL!,
    process.env.NEXT_PUBLIC_SUPABASE_ANON_KEY!,
    {
      cookies: {
        getAll() {
          return request.cookies.getAll()
        },
        setAll(cookiesToSet) {
          cookiesToSet.forEach(({ name, value, options }) =>
            request.cookies.set(name, value, options)
          )
          supabaseResponse = NextResponse.next({
            request,
          })
          cookiesToSet.forEach(({ name, value, options }) =>
            supabaseResponse.cookies.set(name, value, options),
          )
        },
      },
    },
  )

  // IMPORTANT: Avoid writing any logic between createServerClient and
  // supabase.auth.getUser(). A simple mistake could make it very hard to debug
  // issues with users being randomly logged out.

  const {
    data: { session },
  } = await supabase.auth.getSession()

  // Get the user's role using the custom getUserRole function
  const { user_roles } = await getUserRole()
  console.log(user_roles)
  const url = new URL(request.url)

  await supabase.auth.getUser()

  if (session?.user.id) {
    if (url.pathname === "/login") {
      return NextResponse.redirect(new URL("/dashboard", request.url))
    }
    if (
      request.nextUrl.pathname.startsWith("/admin") &&
      !user_roles.includes("admin")
    ) {
      const url = request.nextUrl.clone()
      url.pathname = "/"
      return NextResponse.redirect(url)
    }
    return supabaseResponse
  } else {
    if (protectedPaths.includes(url.pathname)) {
      return NextResponse.redirect(
        new URL("/login?next=" + url.pathname, request.url),
      )
    }
    return supabaseResponse
  }

  return supabaseResponse
}

```

Figure 4.29: Admin-exclusive Sidebar Navigation Links and Middleware to Protect Admin Routes

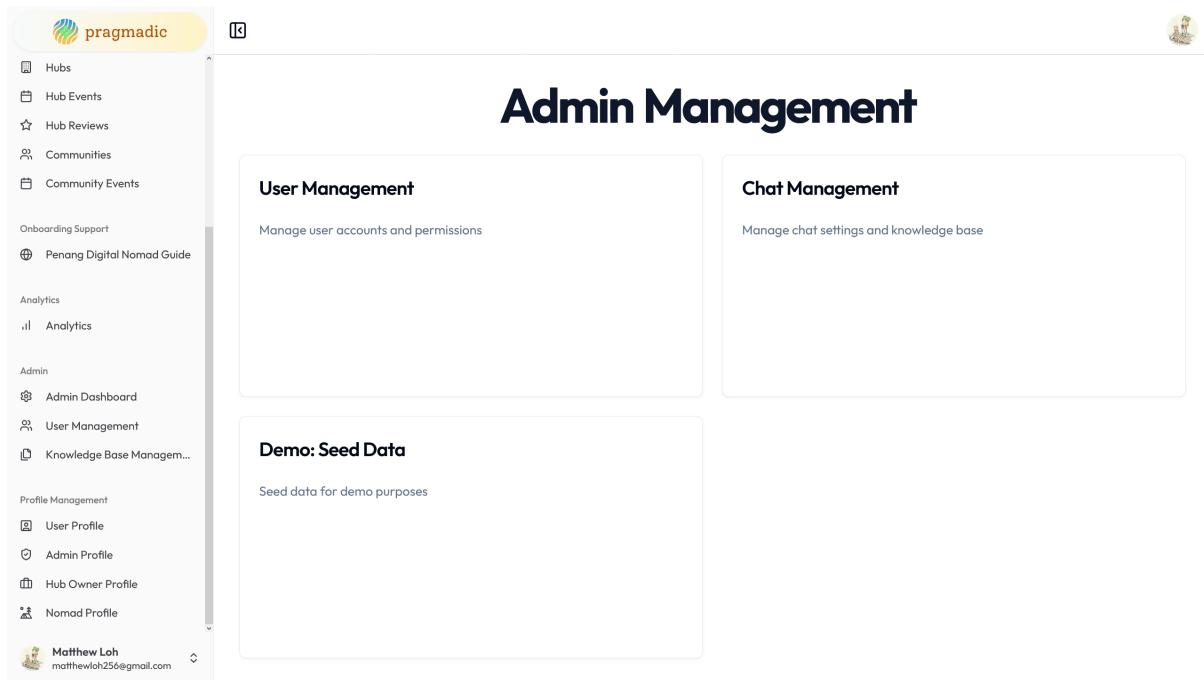


Figure 4.30: Admin Management Dashboard

Figure 4.29 and Figure 4.30 are screenshots of how the Admin Dashboard is able to be navigated to by only admin users. The middleware.ts is a Next.js built-in file convention that allows each request made to the server be checked and protected by proper session management practices. In the middleware.ts file of the system, roles are able to be checked by decoding the user_roles from the JWT with custom claims. Then, simple Array includes function calls are used to protect routes from unauthorized access by matching the request's path with the protected route.

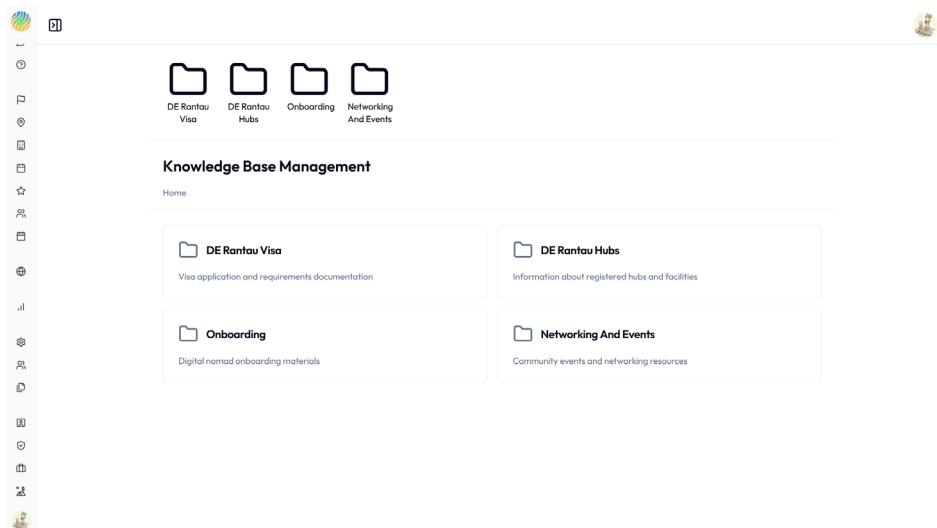


Figure 4.31: Knowledge Base Management Page of admin/chat

The folders available for admins to upload documents in the knowledge base is shown in Figure 4.31. Here, a Next.js catch all route is used with the convention `[...folders]` to enable optimized URL state management. The client components are able to consume the currently viewed folder to render state on the fly from the Supabase Storage.

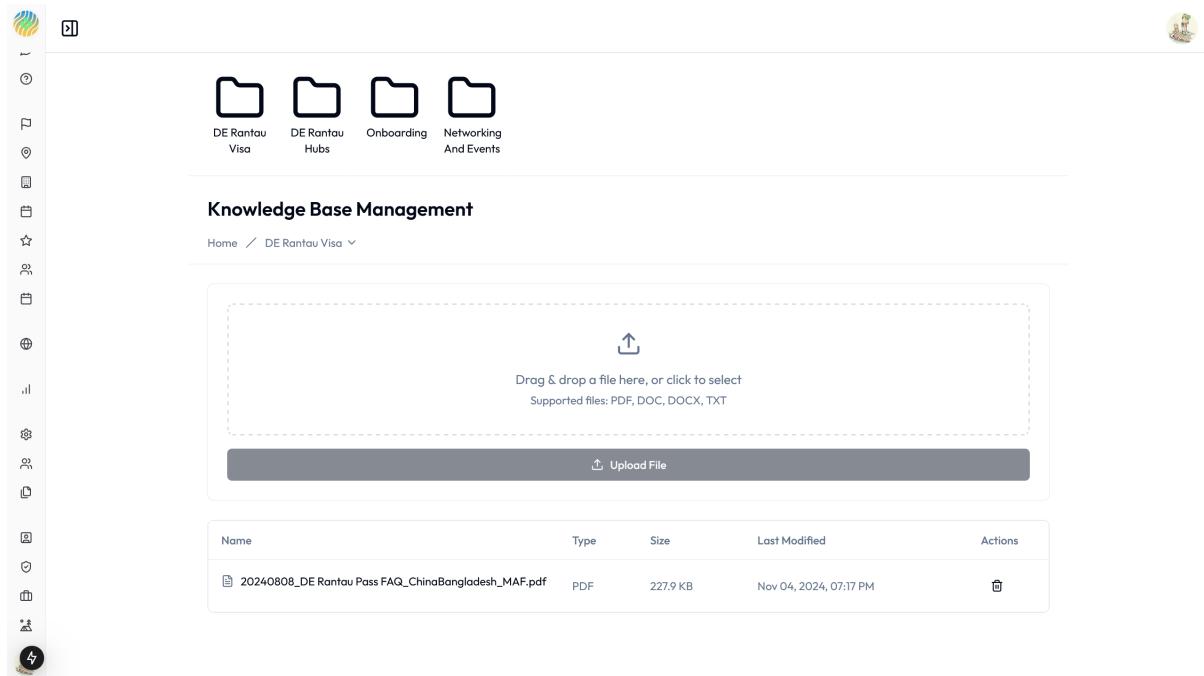


Figure 4.32: Folder View (DE Rantau Visa)

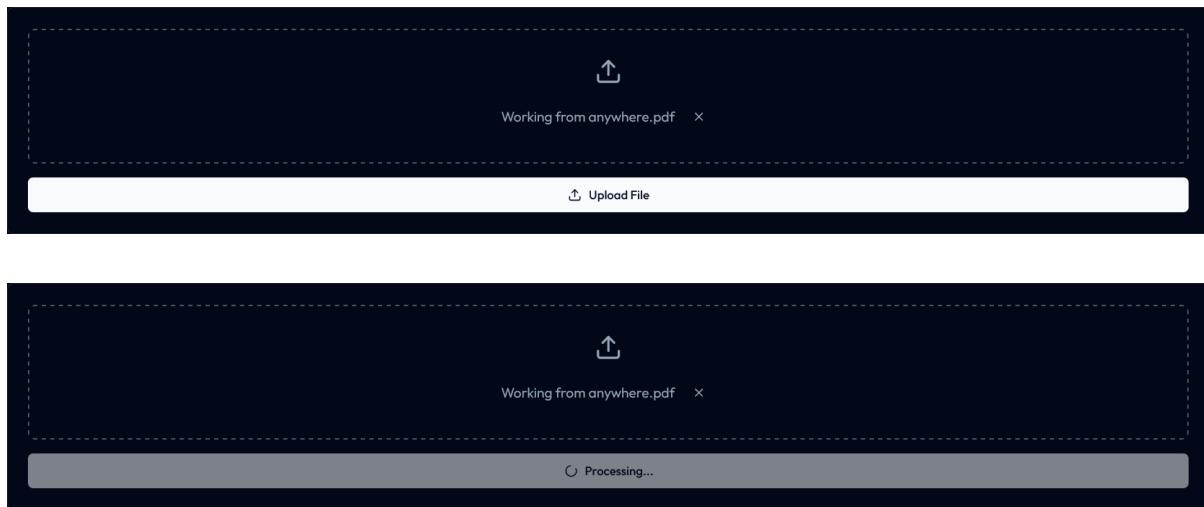


Figure 4.33: File Upload Component built with React Drop Zone with loading states

```

import {useState} from "react";
import {useDropzone} from "react-dropzone";
import {useFileUpload} from "use-file-upload";
import {useSupabase} from "@supabase/react";

const FileUpload = ({ folder }) => {
  const [file, setFile] = useState(null);
  const [isProcessing, setIsProcessing] = useState(false);
  const supabase = useSupabase();
  const [isPending, setIsPending] = useState(false);

  const handleFileChange = (e) => {
    const file = e.target.files[0];
    setFile(file);
  };

  const handleUpload = async () => {
    if (!file) return;
    const res = await useFileUpload({ file, folder });
    if (res.error) {
      console.error(res.error);
      return;
    }
    const fileUrl = res.data?.url;
    if (!fileUrl) return;
    const response = await fetch(fileUrl);
    const blob = await response.blob();
    const fileObj = {
      name: file.name,
      type: file.type,
      size: file.size,
      url: fileUrl,
      blob
    };
    const { data } = await supabase.storage
      .from("knowledge_base")
      .upload(fileObj.name, fileObj);
    if (data.error) {
      console.error(data.error);
      return;
    }
    setFile(null);
    setIsProcessing(false);
  };

  const handleReset = () => {
    setFile(null);
    setIsProcessing(false);
  };

  const handleDrop = (acceptedFiles) => {
    if (acceptedFiles.length) {
      setFile(acceptedFiles[0]);
    }
  };

  const getRootProps = () => {
    const dropzoneProps = useDropzone({
      onDrop: handleDrop,
      maxFiles: 1,
      accept: {
        "application/pdf": ["*.pdf"],
        "application/msword": ["*.doc"],
        "application/vnd.openxmlformats-officedocument.wordprocessingml.document": ["*.docx"],
        "text/plain": ["*.txt"]
      }
    });
    return dropzoneProps;
  };

  const handleUploadAsync = async () => {
    if (!file || !folder) return;
    try {
      setIsProcessing(true);
      const uploadResult = await upload({ files: [file] });
      const result = uploadResult[0];
      if (result.error) {
        throw new Error(`Upload failed: ${result.error.message}`);
      }
      if (!result.data) {
        throw new Error(`Upload failed: No data returned`);
      }
      // Pass the object ID from the upload result
      await generateEmbeddingForPdf(result);
      result.data.filePath;
      result.folder;
      result.data.id;
    } catch (error) {
      console.error(`Upload error: ${error}`);
      toast.error(`Failed to process file`, { error });
    }
    resetUpload();
  };
};

return (
  <div className="space-y-4">
    <div>
      {getRootProps()}
      <div>
        <div>
          <div>
            <div><span>${file}</span></div>
            <div><span>${file.name}</span></div>
            <button
              variant="ghost"
              size="icon"
              onClick={() => {
                e.stopPropagation();
                resetUpload();
              }}
            ><span>Delete</span></button>
            <div><span>${file.size}</span></div>
            <div><span>${file.type}</span></div>
            <div><span>${file.url}</span></div>
          </div>
        </div>
        <div><span>${file}</span></div>
        <div><span>${file.name}</span></div>
        <button
          onClick={handleUpload}
          disabled={!file || isPending || isProcessing}
          className="w-full"
        ><span>${isPending || isProcessing ? "Uploading..." : "Upload File"}</span></button>
      </div>
    </div>
  </div>
);
}

export default FileUpload;

```

Figure 4.34: File Upload and File List Components Code Snippets

The `FileList` and `FileUpload` components taken in Figure 4.34 in the admin section of the application manage file storage and display using Supabase. The `FileList` component displays files in a table format, showing details such as name, type, size, and last modified date. It uses the `useDirectory` hook to fetch files and the `useRemoveFiles` hook for deletion, providing a delete confirmation dialog and handling errors with toast notifications. The `FileUpload` component allows users to upload files via drag-and-drop or file selection, supporting PDF, DOC, DOCX, and TXT formats. It uses the `useUpload` hook for file uploads and triggers the `generateEmbeddingsForPdfAction` for processing, displaying upload progress and handling errors with toast notifications.

4.3.3 Document Processing and Retrieval

```

    pragmatics:v1 uploadFileTo
export async function generateEmbeddingsForPdfAction(
  fullPath: string,
  folder: string,
  objectId: string,
) {
  const supabase = await createClient()
  const { user } = await supabase.auth.getUser()
  if (!user) {
    redirect("/login")
  }

  let document: any = null

  try {
    const fileName = fullPath.split("/").pop() || ""
    const storagePath = fullPath.replace("knowledge_base/", "")

    // 1. Create document record
    const [createdDocument] = await db
      .insert("documents")
      .values([
        {
          name: fileName,
          ownerId: user.id,
          contentType: "pdf",
          status: "processing",
          path: fullPath,
          pathTokens: getPathTokens(fullPath),
          parentId: folder,
          objectId: objectId,
          metadata: {
            originalName: fileName,
            folder: folder,
            fullPath: fullPath,
          },
        }
      ])
      .returning()

    document = createdDocument

    // 2. Download file from Supabase
    const [data: FileData, error: downloadError] = await supabase.storage
      .from("knowledge_base")
      .download(storagePath)

    if (error || !fileData) {
      throw new Error(`Failed to download file: ${error?.message}`);
    }
  }

  // 3. Extract and process PDF content using Pdf.js
  const arrayBuffer = await fileData.arrayBuffer()
  const pdf = await getDocumentProxy(new Uint8Array(arrayBuffer))

  const { text, totalPages } = await extractText(pdf, {
    mergePages: true,
  })

  if (!text) {
    throw new Error("Failed to extract text from PDF");
  }

  // 4. Split into chunks
  const textSplitter = new RecursiveCharacterTextSplitter({
    chunkSize: 1000,
    chunkOverlap: 200,
  })
  const chunkedContent = await textSplitter.createDocuments([text])

  // 5. Generate embeddings
  const embeddings = await embedMany({
    model: opnai.embedding("text-embedding-llm"),
    values: chunkedContent.map((chunk) => chunk.pageContent),
  })

  // 6. Get public URL for reference
  const [urlData] = await supabase.storage
    .from("knowledge_base")
    .getPublicUrl(storagePath)

  // 7. Insert chunks with embeddings
  await Promise.all(
    chunkedContent.map((chunk, i) =>
      db.insert("document_chunks").values([
        {
          documentId: document.id,
          content: chunk.pageContent,
          embedding: embeddings[i],
          filePath: storagePath,
          metadata: {
            chunkIndex: i,
            length: chunk.pageContent.length,
            folder: folder,
            pageCount: totalPages,
          },
        },
      ]),
    )
  )

  // 8. Update document status
  await db
    .update("documents")
    .set({
      status: "completed",
      chunkCount: chunkedContent.length,
      processedAt: new Date(),
      folder: folder,
      title: fileName.replace(".pdf", ""),
      sourceUrl: urlData.publicURL,
      metadata: {
        ...document.metadata,
        pageCount: totalPages,
      },
    })
    .where(eq("documents.id", document.id))

  return { success: true, documentId: document.id }
} catch (error) {
  console.error(`Error processing document: ${error.message}`)
  if (document.id && error instanceof Error) {
    await db
      .update("documents")
      .set({
        status: "failed",
        metadata: {
          ...document.metadata,
          error: error.message,
        },
      })
      .where(eq("documents.id", document.id))
  }
  throw error
}
}

```

Figure 4.35: generateEmbeddingsForPdfAction server action

The generateEmbeddingsForPdfAction server action is responsible for processing uploaded PDF files. It begins by creating a Supabase client and checking user authentication. If the user is not authenticated, it redirects to the login page. The action then creates a document record in the database, downloads the file from Supabase storage, and extracts text from the PDF using the unpdf library. The extracted text is split into chunks using a RecursiveCharacterTextSplitter, and embeddings are generated for each chunk using OpenAI's embedding model. The action retrieves a public URL for the file and inserts the chunks with their embeddings into the database. Finally, it updates the document status to "completed" and records metadata such as the number of pages and the source URL. If an error occurs during processing, the document status is updated to "failed," and the error message is recorded in the metadata. This comprehensive process ensures that uploaded documents are efficiently processed and stored with relevant metadata and embeddings for future retrieval and analysis. Helper functions provided by the Vercel AI SDK also significantly simplify the process of embedding many chunks of text using the "text-embedding-3-small" model provided by OpenAI. The Drizzle ORM library also allows for database transactions to ensure that operations are atomic.

content	text	embedding	vector	document_id	uuid	file_path	text	metadata	jsonb	created_at	timestamp
67c912e4-8927-45ca-b955-8f64c2f55ab5	This shows a stark contrast between the t	[+0.010463827, 0.039042678, 0.0789649, dce550b2-661d-4856-9f95-4aa91...	→ de_rantau_hubs/Working from anywhere {"folder": "de_rantau_hubs", "length": "997", "created_at": "2024-11-06 15:18:31.66164+000}	de_rantau_hubs							
2871cfae-d1de-419e-979a-444d76d0ed4d	its migration laws and created a new "sing	[+0.26694797, 0.0736308, 0.06429379, -dce550b2-661d-4856-9f95-4aa91...	→ de_rantau_hubs/Working from anywhere {"folder": "de_rantau_hubs", "length": "994", "created_at": "2024-11-06 15:18:31.69597+000}	de_rantau_hubs							
ca903fc8-f5d3-45d0-b99c-1ead8fd5e471	bigger, industrialised economies. Second,	[+0.027322097, 0.04119426, 0.039728505, dce550b2-661d-4856-9f95-4aa91...	→ de_rantau_hubs/Working from anywhere {"folder": "de_rantau_hubs", "length": "996", "created_at": "2024-11-06 15:18:31.69034+000}	de_rantau_hubs							
babe67b6-bbbd-4ae6-94e6-746303399b6	view that digital nomad visas can be impl	[+0.014437318, 0.051458546, 0.047468055, dce550b2-661d-4856-9f95-4aa91...	→ de_rantau_hubs/Working from anywhere {"folder": "de_rantau_hubs", "length": "999", "created_at": "2024-11-06 15:18:31.704228+000}	de_rantau_hubs							
c642dc34-76be-4626-aee6-bfe62a9919a	Grenada Tourism Authority announced th	[+0.019355343, 0.021442494, 0.062327053, dce550b2-661d-4856-9f95-4aa91...	→ de_rantau_hubs/Working from anywhere {"folder": "de_rantau_hubs", "length": "998", "created_at": "2024-11-06 15:18:31.70814+000}	de_rantau_hubs							
701c8ab7-ed73-494b-9248-a56a6ba4512	Identified approaches also shed light on t	[+0.0015789102, 0.016476423, 0.06273663, dce550b2-661d-4856-9f95-4aa91...	→ de_rantau_hubs/Working from anywhere {"folder": "de_rantau_hubs", "length": "997", "created_at": "2024-11-06 15:18:31.731321+000}	de_rantau_hubs							
7b3c05d4-8328-4698-ac77-700865cc01f	aims to boost the country's tourism indu	[+0.009398270, 0.0379851, 0.071315383, -dce550b2-661d-4856-9f95-4aa91...	→ de_rantau_hubs/Working from anywhere {"folder": "de_rantau_hubs", "length": "994", "created_at": "2024-11-06 15:18:31.735581+000}	de_rantau_hubs							
9efcd32-756b-4da3-9ebc-292a7b5c47d6	nomad. Digital nomad visas have been i	[+0.007160932, 0.037249086, 0.06550336, dce550b2-661d-4856-9f95-4aa91...	→ de_rantau_hubs/Working from anywhere {"folder": "de_rantau_hubs", "length": "990", "created_at": "2024-11-06 15:18:31.735811+000}	de_rantau_hubs							
dff037d8-7533-4ff0-8fb2-0327c7573c	taxation of high-earning nomads is anoth	[+0.03138821, 0.034427926, 0.05086902, -dce550b2-661d-4856-9f95-4aa91...	→ de_rantau_hubs/Working from anywhere {"folder": "de_rantau_hubs", "length": "998", "created_at": "2024-11-06 15:18:31.741329+000}	de_rantau_hubs							
d9891bdc-e45b-415b-9922-75fe69fc0333	generates income outside the host count	[+0.09716792, 0.029044908, 0.04705965, dce550b2-661d-4856-9f95-4aa91...	→ de_rantau_hubs/Working from anywhere {"folder": "de_rantau_hubs", "length": "993", "created_at": "2024-11-06 15:18:31.747611+000}	de_rantau_hubs							
a14387fe-b960-4633-be0c-3e6f633a610f	in the sample) have been particularly "fir	[+0.01046187, 0.024852097, 0.062327053, dce550b2-661d-4856-9f95-4aa91...	→ de_rantau_hubs/Working from anywhere {"folder": "de_rantau_hubs", "length": "999", "created_at": "2024-11-06 15:18:31.742228+000}	de_rantau_hubs							
047b54f6-0a85-42d9-90bd-3977fe4d67	by their income from work or entrepreneur	[+0.02346296, 0.05715934, 0.043971065, dce550b2-661d-4856-9f95-4aa91...	→ de_rantau_hubs/Working from anywhere {"folder": "de_rantau_hubs", "length": "995", "created_at": "2024-11-06 15:18:31.74045+000}	de_rantau_hubs							
f441d777-9408-4602-972c-7bf0ceccas9	some cases possibly leading even to perm	[+0.002533673, 0.0049470535, 0.047682215, dce550b2-661d-4856-9f95-4aa91...	→ de_rantau_hubs/Working from anywhere {"folder": "de_rantau_hubs", "length": "997", "created_at": "2024-11-06 15:18:31.766608+000}	de_rantau_hubs							
1627674c-aesa-48ab-9c17-1e943d149478	of digital nomads from the institutional pe	[+0.009817370, 0.03931653, 0.04532659, -dce550b2-661d-4856-9f95-4aa91...	→ de_rantau_hubs/Working from anywhere {"folder": "de_rantau_hubs", "length": "999", "created_at": "2024-11-06 15:18:31.768079+000}	de_rantau_hubs							
b3fc94e-417b-4a40-bd22-ac650abc1e	I leverage the new work patterns to diversi	[+0.018861583, 0.019059888, 0.048058636, dce550b2-661d-4856-9f95-4aa91...	→ de_rantau_hubs/Working from anywhere {"folder": "de_rantau_hubs", "length": "996", "created_at": "2024-11-06 15:18:31.772217+000}	de_rantau_hubs							
925a75db-ff22-4e85-9978-d4a42665850	Future research could, first and foremost,	[+0.008554096, 0.022907637, 0.05230723, dce550b2-661d-4856-9f95-4aa91...	→ de_rantau_hubs/Working from anywhere {"folder": "de_rantau_hubs", "length": "990", "created_at": "2024-11-06 15:18:31.789828+000}	de_rantau_hubs							
fc606324-705c-4cf2-8e76-56321c00004	broaden tourism development strategies	[+0.008363192, 0.028703634, 0.0653026, dce550b2-661d-4856-9f95-4aa91...	→ de_rantau_hubs/Working from anywhere {"folder": "de_rantau_hubs", "length": "998", "created_at": "2024-11-06 15:18:31.808466+000}	de_rantau_hubs							
4ddc9ab-adb2-4915-4076-70bd46507	et al., 2004; Mancinelli, 2020). Finally, thi	[+0.003392994, 0.028703634, 0.0675054, dce550b2-661d-4856-9f95-4aa91...	→ de_rantau_hubs/Working from anywhere {"folder": "de_rantau_hubs", "length": "995", "created_at": "2024-11-06 15:18:31.810304+000}	de_rantau_hubs							
aed04701-8f70-4a85-a756-d1c1e6466	Research 105 (2024) 103715 11 Credit author	[+0.0219701, 0.046387203, 0.05038061, dce550b2-661d-4856-9f95-4aa91...	→ de_rantau_hubs/Working from anywhere {"folder": "de_rantau_hubs", "length": "999", "created_at": "2024-11-06 15:18:31.82235+000}	de_rantau_hubs							
b7c17a00-1100-4d62-834e-4b2538d6f6d	Conference Imaging Experiences: Creat	[+0.019470375, 0.0476339, 0.0855651, -0.00 dce550b2-661d-4856-9f95-4aa91...	→ de_rantau_hubs/Working from anywhere {"folder": "de_rantau_hubs", "length": "996", "created_at": "2024-11-06 15:18:31.835649+000}	de_rantau_hubs							
9cd2112-6252-43f2-8972-623d76749c5	(9'21), 259–266. Benson, M., & O'Reilly, K.	[+0.02446959, 0.059382267, 0.016087910, dce550b2-661d-4856-9f95-4aa91...	→ de_rantau_hubs/Working from anywhere {"folder": "de_rantau_hubs", "length": "984", "created_at": "2024-11-06 15:18:31.83814+000}	de_rantau_hubs							
82c693fb-d4f3-a42c-a477-575fb0fb39	Cohorts and policy outputs in advance	[+0.009046213, 0.021334467, 0.026682584, dce550b2-661d-4856-9f95-4aa91...	→ de_rantau_hubs/Working from anywhere {"folder": "de_rantau_hubs", "length": "998", "created_at": "2024-11-06 15:18:31.844244+000}	de_rantau_hubs							
36652749-995c-45d4-9125-d45459556	00182-3/METRICS. Brynjolfsson, E., Hori	[+0.0129871, 0.05673492, 0.06053395, -dce550b2-661d-4856-9f95-4aa91...	→ de_rantau_hubs/Working from anywhere {"folder": "de_rantau_hubs", "length": "995", "created_at": "2024-11-06 15:18:31.84439+000}	de_rantau_hubs							
e63d2f15-169a-49b6-a4b8-1864cca78655	of international tourism. Social Research,	[+0.020538326, 0.022490649, 0.03116122, dce550b2-661d-4856-9f95-4aa91...	→ de_rantau_hubs/Working from anywhere {"folder": "de_rantau_hubs", "length": "993", "created_at": "2024-11-06 15:18:31.85266+000}	de_rantau_hubs							
c5382dbd-4856-4ef1-a339-12305cb159b	Proceedings of the 2021 IEEE 24th Intern	[+0.007865691, 0.02301534, 0.05070219, -dce550b2-661d-4856-9f95-4aa91...	→ de_rantau_hubs/Working from anywhere {"folder": "de_rantau_hubs", "length": "996", "created_at": "2024-11-06 15:18:31.85398+000}	de_rantau_hubs							
8e80883b-329c-402d-ba22-d798db4b881	and the use of disciplining practices to m	[+0.0037469903, 0.037876334, 0.04851479, dce550b2-661d-4856-9f95-4aa91...	→ de_rantau_hubs/Working from anywhere {"folder": "de_rantau_hubs", "length": "999", "created_at": "2024-11-06 15:18:31.854194+000}	de_rantau_hubs							
7af46ef8-7bdc-4fca-a872-99d652d7ec3	T. B. (2018). Decent digital work: Technol	[+0.03953013, 0.02193215, 0.035626296, -dce550b2-661d-4856-9f95-4aa91...	→ de_rantau_hubs/Working from anywhere {"folder": "de_rantau_hubs", "length": "997", "created_at": "2024-11-06 15:18:31.85555+000}	de_rantau_hubs							
4f1467dd-200c-42b5-8b52-1d0ccac7ed3	Sciences 2021 (HICSS-54). Godin, K., Sta	[+0.03979565, 0.0252016, 0.0418594, -dce550b2-661d-4856-9f95-4aa91...	→ de_rantau_hubs/Working from anywhere {"folder": "de_rantau_hubs", "length": "990", "created_at": "2024-11-06 15:18:31.87859+000}	de_rantau_hubs							
72987-00-8794-402b-95b2-592e29868ff	nomad: Defining the phenomenon. Inform	[+0.040000368, 0.028453577, 0.04370675, dce550b2-661d-4856-9f95-4aa91...	→ de_rantau_hubs/Working from anywhere {"folder": "de_rantau_hubs", "length": "996", "created_at": "2024-11-06 15:18:31.89159+000}	de_rantau_hubs							

Figure 4.36: The chunked and embedded document is stored in the document_chunks table with related metadata.

```

pragmatic-v1 - ChatComponent.tsx
const [uploadedFiles, setUploadedFiles] = useState<FileInfo[]>([])

const {
  messages,
  handleSubmit,
  input,
  setInput,
  append,
  handleInputChange,
  isLoading,
  stop,
} = useChat({
  body: {
    chatId,
    selectedDocumentIds,
    model: selectedModel.model,
  },
  initialMessages,
  onFinish: () => {
    router.push(`/chat/${chatId}`)
    window.history.replaceState(null, "", `/chat/${chatId}`)
  },
  async onToolCall({ toolCall }) {
    console.log("toolCall", toolCall.toolName)
    console.log("toolCall", toolCall.toolCallId)
  },
})
}

pragmatic-v1 - ragMiddleware.ts
export const ragMiddleware: Experimental_LanguageModelV1Middleware = {
  transformParams: async ({ params }) => {
    const { session } = await getUserAuth()
    if (!session) return params
    console.log("params", params)
    const { prompt: messages, providerMetadata } = params

    // validate the provider metadata with Zod:
    const { success, data } = selectionSchema.safeParse(providerMetadata)
    if (!success) return params // no documents selected

    const selectedDocumentIds = data.documents.selection
    if (selectedDocumentIds.length === 0) return params

    const recentMessage = messages.pop()
    if (!recentMessage || recentMessage.role !== "user") {
      if (recentMessage) messages.push(recentMessage)
      return params
    }

    const lastUserMessageContent = recentMessage.content
      .filter((content) => content.type === "text")
      .map((content) => content.text)
      .join("\n")

    // Classify the user prompt
    const { object: classification } = await generateObject({
      model: openai("gpt-4o-mini", { structuredOutputs: true }),
      output: "enum",
      enum: ["question", "statement", "other"],
      system: "classify the user message as a question, statement, or other",
      prompt: lastUserMessageContent,
    })
    console.log("classification", classification)

    // only use RAG for questions
    if (classification === "question") {
      messages.push(recentMessage)
      return params
    }

    // Generate hypothetical answer
    const { text: hypotheticalAnswer } = await generateText({
      model: openai("gpt-4o-mini", { structuredOutputs: true }),
      system: "Answer the users question:",
      prompt: lastUserMessageContent,
    })
    console.log("hypotheticalAnswer", hypotheticalAnswer)
    // Embed the hypothetical answer
    const { embedding: hypotheticalAnswerEmbedding } = await embed({
      model: openai.embedding("text-embedding-3-small"),
      value: hypotheticalAnswer,
    })

    // Get document chunks from Supabase
    const chunksBySelection = await getChunksByDocumentIds({
      documentIds: selectedDocumentIds,
    })
    if (!chunksBySelection) {
      messages.push(recentMessage)
      return params
    }

    // Calculate similarities and rank chunks
    const chunksWithSimilarity = chunksBySelection.map((chunk) => ({
      ...chunk,
      similarity: cosineSimilarity(
        hypotheticalAnswerEmbedding,
        chunk.embedding,
      ),
    }))
    // Sort by similarity and take top K chunks
    chunksWithSimilarity.sort((a, b) => b.similarity - a.similarity)
    const k = 10
    const topKChunks = chunksWithSimilarity.slice(0, k)

    // Add context to the message
    messages.push({
      role: "user",
      content: [
        ...recentMessage.content,
        {
          type: "text",
          text: "\nRelevant context from selected documents:",
        },
        ...topKChunks.map((chunk) => ({
          type: "text" as const,
          text: chunk.content,
        })),
      ],
    })
    return { ...params, prompt: messages }
  },
}
}

return result.toDataStreamResponse()
}
}

```

Figure 4.37: chat/route.ts is the endpoint which is called by useChat hook

The useChat hook provided by the AI SDK UI package provides the ability to extend the interface of information that is able to be passed through to the LLM call. In this case, the DocumentSelector is able to pass through the document ID to carry out the RAG workflow. The RAG (Retrieval-Augmented Generation) workflow in this system is designed to enhance AI responses by incorporating relevant document context. The ragMiddleware in src/lib/ai/ragMiddleware.ts plays a crucial role in this process. It begins by validating the provider metadata using Zod to ensure that document selections are valid. If documents are selected, it processes the user's most recent message, classifying it as a question, statement, or other using OpenAI's gpt-4o-mini model. If the message is a question, a hypothetical answer is generated and embedded using OpenAI's embedding model. The middleware retrieves document chunks from Supabase using the getChunksByDocumentIds function, calculates cosine similarity between the hypothetical answer's embedding and each chunk's embedding, and selects the top K chunks with the highest similarity. These chunks are then appended to the user's message as additional context, enhancing the AI's ability to generate informed responses.

```

pragmatic-v1 - custom-model.ts

import { openai } from "@ai-sdk/openai"
import { experimental_wrapLanguageModel as wrapLanguageModel } from "ai"
import { customMiddleware } from "@lib/ai/custom-middleware"
import { google } from "@ai-sdk/google"
import { anthropic } from "@ai-sdk/anthropic"
import { ragMiddleware } from "./ragMiddleware"

export const gpt4oModel = wrapLanguageModel({
  model: openai("gpt-4o"),
  middleware: ragMiddleware,
})

export const gpt4ominiModel = wrapLanguageModel({
  model: openai("gpt-4o-mini"),
  middleware: ragMiddleware,
})

export const geminiProModel = wrapLanguageModel({
  model: google("gemini-1.5-pro-latest"),
  middleware: ragMiddleware,
})

export const geminiFlashModel = wrapLanguageModel({
  model: google("gemini-1.5-flash"),
  middleware: ragMiddleware,
})

export const anthropicModel = wrapLanguageModel({
  model: anthropic("claude-3-haiku-20240307"),
  middleware: customMiddleware,
})

```

Figure 4.38: Custom Model used to Wrap Different Providers with ragMiddleware

Custom models are defined using the AI SDK by wrapping language models with middleware. Figure 4.38 demonstrates how to integrate the ragMiddleware with OpenAI's gpt-4o model.

This setup allows the model to leverage the RAG workflow for improved response generation. The ChatComponent in src/features/chat/components/ChatComponent.tsx utilizes the useChat hook to manage chat interactions. This hook handles message input, submission, and response processing. It passes the selectedDocumentIds to the language model as part of the request body, enabling the RAG workflow to retrieve relevant document chunks. The useChat hook also manages the chat's state, including messages, input, and loading status, ensuring a seamless user experience. By integrating the RAG workflow and custom models, the system effectively combines AI capabilities with document retrieval to provide contextually enriched responses, enhancing the overall chat experience.

4.4 Onboarding Support Module



Figure 4.39: Onboarding Component available to all users

The onboarding map module is designed to provide users with an interactive map interface that displays nearby events and allows for easy navigation and exploration. The primary function is to enhance user engagement by visually representing event locations on a map, making it easier for users to discover and participate in events.

4.4.1 Rich Interactive Mapbox Onboarding Support Map Component

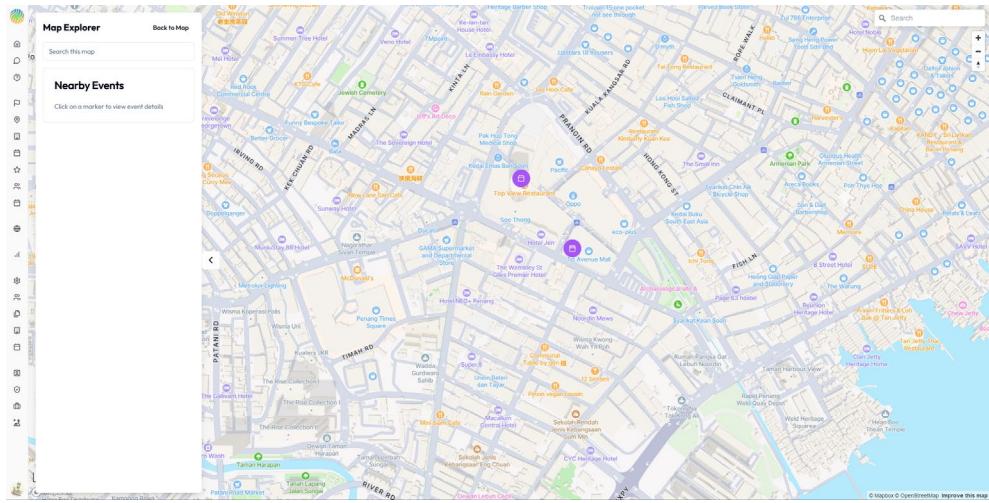


Figure 4.40: Interactive Map Component using MapboxGL under the hood

The module integrates with Supabase to fetch event data and uses Mapbox for map rendering using React-map-gl library to work in Next.js compatibly, offering a seamless and dynamic user experience. By clicking on map markers, users can view detailed event information, including venue, address, and event timings, all within an intuitive interface. This functionality aims to facilitate user onboarding by providing a clear and engaging way to explore events and locations.

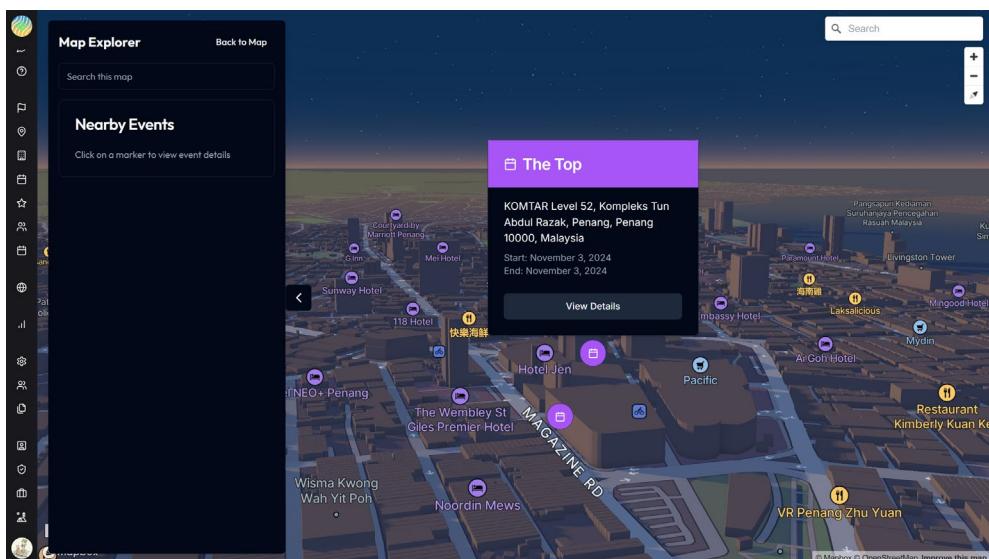


Figure 4.41 Dark Mode Aware Styling and Ability to Control Map using Mapbox APIs

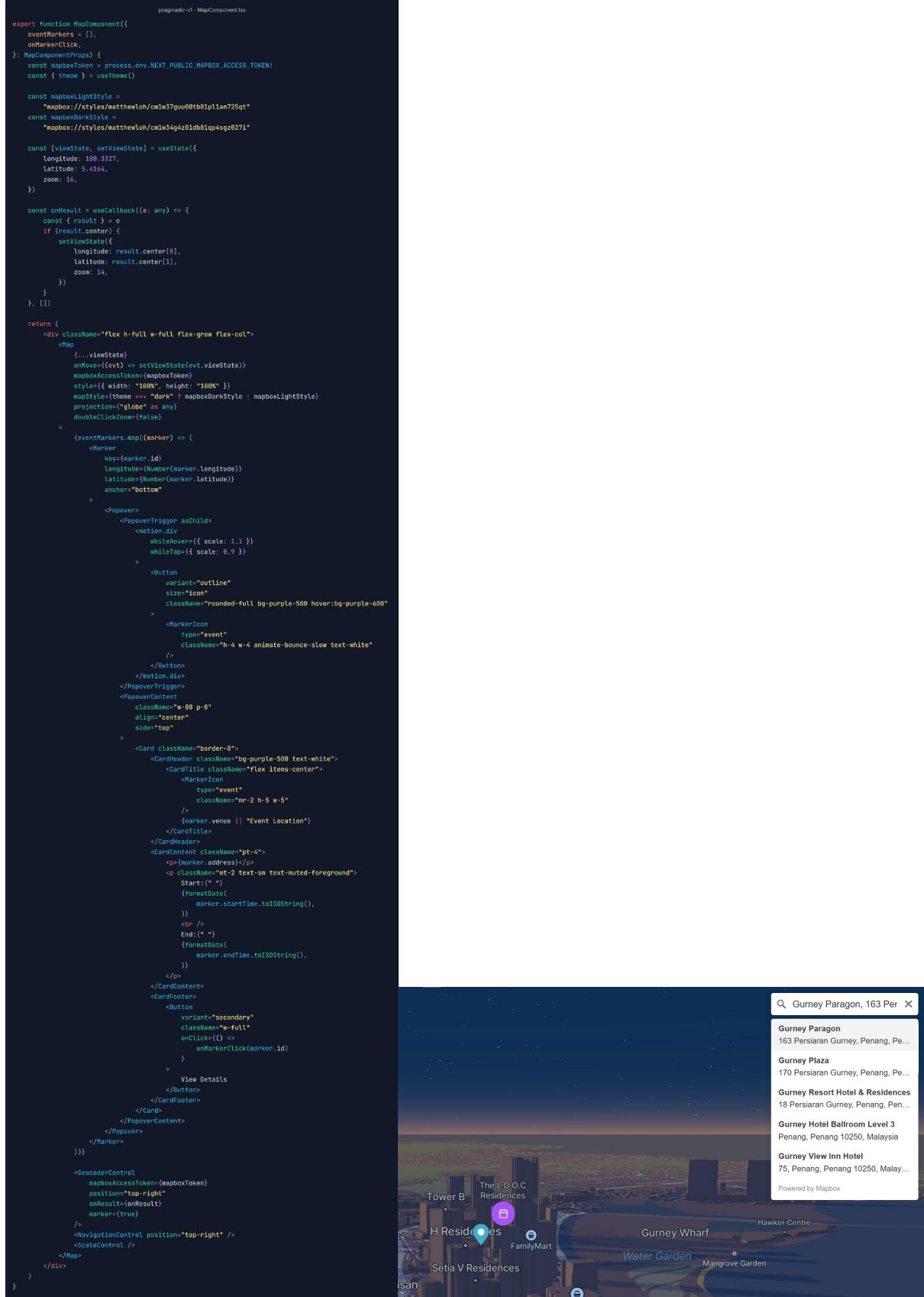


Figure 4.42: Dark mode Integrated Styling and Ability to Control Map using Mapbox APIs (Geocoding)

The module leverages the Mapbox API to provide advanced map functionalities, including geocoding and navigation controls. Users can search for locations using the Mapbox Geocoder, which updates the map view based on search results. The map styles are dynamically adjusted based on the user's theme preference, switching between light and dark styles to enhance visual appeal and usability. The light style uses the mapboxLightStyle URL, while the dark style uses the mapboxDarkStyle URL. This adaptability ensures that the map remains visually consistent with the rest of the application, providing a cohesive user experience. The module uses react-map-gl to implement the map interface, providing a robust and flexible solution for rendering interactive maps in React applications. The MapComponent manages the map's state, including view settings like latitude, longitude, and zoom level. It also integrates Mapbox controls such as NavigationControl and ScaleControl to enhance user interaction. The MapProvider component wraps the map, ensuring that all map-related components have access to the necessary context. This implementation allows for smooth map interactions and efficient rendering of map elements, contributing to a responsive and user-friendly interface built on top of Mapbox standard components.

4.4.2 Browsing Location-based Events

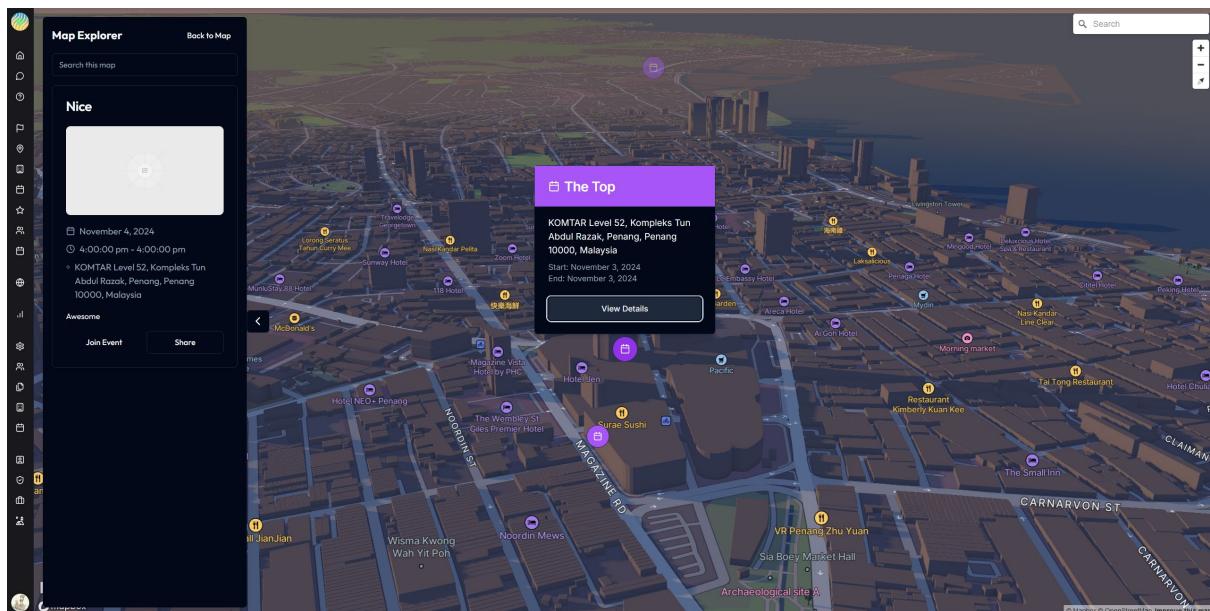


Figure 4.43: Browsing Events on the MapSidebar and MapComponent

```

pragmatic-v1 - page.tsx
export default async function OnboardingMapPage() {
  const supabase = await createClient()
  const { data: EventMarkerData } = await supabase
    .from("event_markers")
    .select("*")
    .order("created_at", { ascending: false })
  const eventMarkers = EventMarkerData?.map((marker) => ({ ...marker,
    createdAt: new Date(marker.created_at),
    updatedAt: new Date(marker.updated_at),
    userId: marker.user_id,
    eventId: marker.event_id,
    eventType: marker.event_type,
    startTime: new Date(marker.start_time),
    endTime: new Date(marker.end_time),
    latitude: marker.latitude.toString(),
    longitude: marker.longitude.toString(),
  }))
  return <MapClientComponent eventMarkers={eventMarkers || []} />
}

pragmatic-v1 - MapClientComponent.tsx
export default function MapClientComponent({
  eventMarkers,
}: {
  eventMarkers: EventMarker[]
}) {
  const router = useRouter()
  const searchParams = useSearchParams()
  const { open } = useSidebar()

  const handleMarkerClick = (markerId: string) => {
    const params = new URLSearchParams(searchParams)
    params.set("marker", markerId)
    params.set("type", "event")
    router.replace(`/onboarding/map?${params.toString()}`)
  }

  return (
    <MapProvider>
      <MapComponent
        eventMarkers={eventMarkers}
        onMarkerClick={handleMarkerClick}
      />
      <MapSidebar onBackToMap={() => router.replace("/onboarding/map")}></MapSidebar>
    </MapProvider>
  )
}

pragmatic-v1 - MapComponent.tsx
{eventMarkers.map((marker) => (
  <Marker
    key={marker.id}
    longitudes={Number(marker.longitude)}
    latitudes={Number(marker.latitude)}
    anchor="bottom"
  >
    <Popover>
      <PopoverTrigger asChild>
        <div
          whileHover={{ scale: 1.1 }}
          whileTap={{ scale: 0.9 }}
        >
          <Button
            variant="outline"
            size="icon"
            className="rounded-full bg-purple-500 hover:bg-purple-600"
          >
            <MarkerIcon
              type="event"
              className="h-4 w-4 animate-bounce-slow text-white"
            />
          </Button>
        </div>
      </PopoverTrigger>
      <PopoverContent
        className="w-80 p-8"
        align="center"
        side="top"
      >
        <Card className="border-0">
          <CardHeader className="bg-purple-500 text-white">
            <CardTitle className="flex items-center">
              <MarkerIcon
                type="event"
                className="mr-2 h-5 w-5"
              />
              {marker.venue || "Event location"}
            </CardTitle>
          </CardHeader>
          <CardContent className="pt-4">
            <p>{marker.address}</p>
            <p className="mt-2 text-sm text-muted-foreground">
              Start: {` ${marker.startTime.toISOString()}`}
            </p>
            <hr />
            <p>End: {` ${marker.endTime.toISOString()}`}</p>
            <button
              variant="secondary"
              className="w-full"
              onClick={() => onMarkerClick(marker.id)}
            >
              View Details
            </button>
          </CardContent>
          <CardFooter>
            <button
              variant="secondary"
              className="w-full"
              onClick={() => onMarkerClick(marker.id)}
            >
              View Details
            </button>
          </CardFooter>
        </Card>
      </PopoverContent>
    </Popover>
  </Marker>
))
)}

```

Figure 4.44: Onboarding Map Page fetches data on the server to be consumed by MapComponent as props

Event markers are fetched from a Supabase database in a server component, where each marker represents a hub event location. The OnboardingMapPage server component retrieves marker data, including coordinates, event details, and timestamps, and passes it to the

MapClientComponent. This data is then used to render markers on the map, with each marker displaying a popover containing event information when clicked. The fetching process ensures that the map is populated with up-to-date event data, allowing users to explore current and upcoming events in their vicinity.

Additionally, the map module includes a sidebar that integrates with search parameters to display detailed event information. This way, with the URL being a source of truth, it enables users to be able to share the current item viewed. To achieve this, the MapSidebar component uses the useSearchParams hook to retrieve query parameters, such as the selected marker ID and event type. This integration allows the sidebar to dynamically update its content based on user interactions with the map. When a marker is clicked, the sidebar displays relevant event details, including the event name, description, and timings. This seamless integration between the map and sidebar enhances the user experience by providing quick access to detailed event information.

4.4.3 Hub and Communities Events

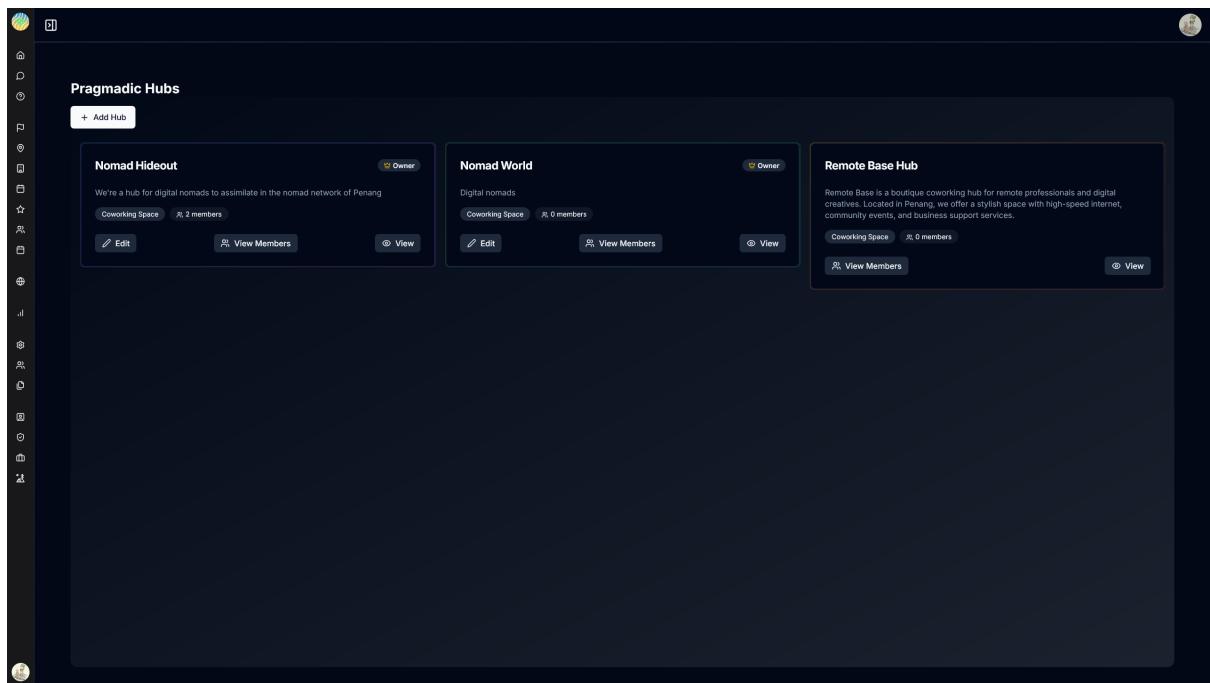


Figure 4.45: Hubs Browsing Page (Owner View)

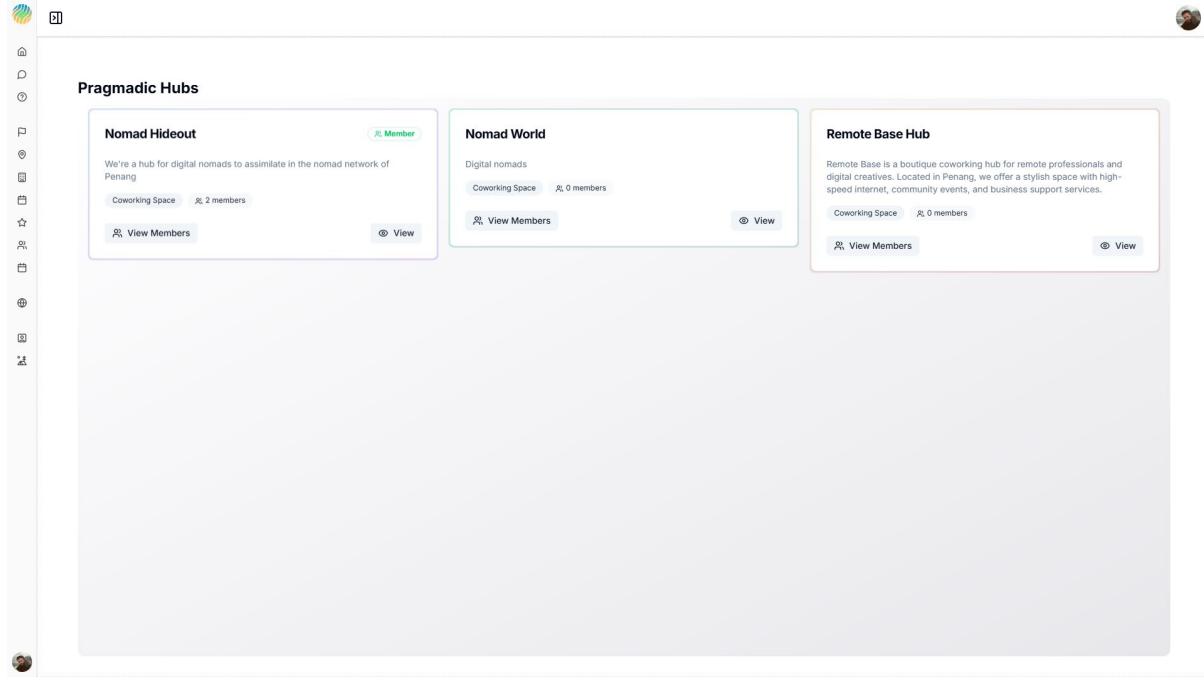


Figure 4.46: Hub Member and Nomad View

The HubList and its constituent components pictured in Figure 4.45 and Figure 4.46 is a dynamic interface for displaying and managing hubs within the Pragmatic platform. It implements role-based access control, differentiating between administrators, hub owners, and regular members. The component utilizes optimistic updates using the `useOptimistic` React 18 hook for a responsive user experience, immediately reflecting changes while server confirmation occurs in the background.

The component manages the display of hubs using a card-based layout, with each hub card showing essential information including the hub's name, description, type, and member count. The implementation leverages Framer Motion for smooth animations and transitions, creating an engaging user interface. The component integrates with Supabase for real-time data management and uses React Query for efficient data fetching and caching. It handles loading states with skeleton loaders and provides an empty state when no hubs are available.

Administrators have comprehensive control over the hub ecosystem. They can view all hubs, create new ones, and edit existing hubs regardless of ownership. The interface provides administrators with a dedicated "Add Hub" button that's always visible, and they have access to edit functionality for any hub in the system. The component implements special UI indicators for admin actions and includes additional management options that are exclusively available

to administrators, ensuring they can effectively oversee and manage the entire hub infrastructure.

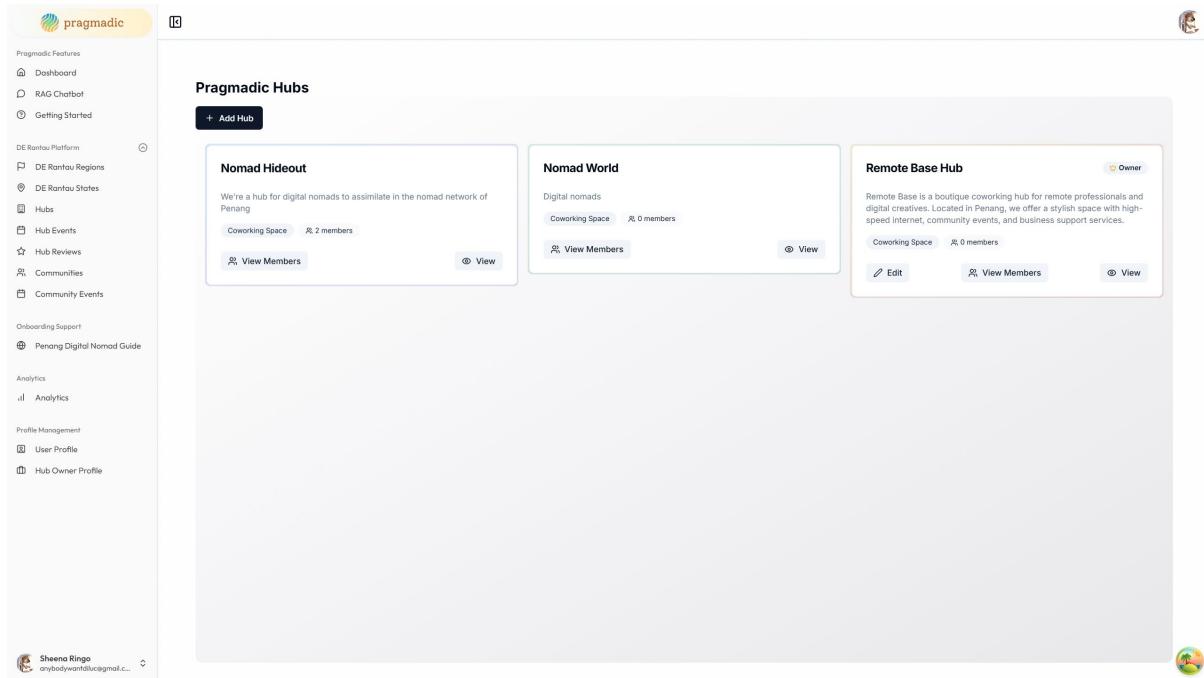


Figure 4.47: Owner View (Alternate)

Hub owners have specific privileges for managing their own hubs as seen in Figure 4.47. The component displays owner-specific controls and badges, including an owner crown icon, for hubs they've created. Owners can edit their hub's details, manage member access, and view member lists. The interface implements owner-specific validation checks before displaying edit controls and ensures that owners can only modify their own hubs. The component also provides owners with insights into their hub's membership and activity through member count displays and access to member management features.

Regular members see a streamlined version of the interface focused on hub discovery and participation. The component shows membership status through badges and provides appropriate navigation options for viewing hub details and participating in hub activities. Members can view hub information, see their membership status indicated by badges, and access hub content through the "View" button. The interface adapts to show relevant information for members while restricting access to management features, creating a clear distinction between member and owner/admin capabilities.

The figure displays two side-by-side dark-themed forms for managing hubs, likely created using the Shadcn UI library.

Create Hub Form (Left):

- Name:** Input field (empty).
- Description:** Input field (empty).
- Type Of Hub:** Input field (empty).
- Public:** Checkbox (unchecked).
- Info:** Input field (empty).
- State:** Input field containing "Pulau Pinang".
- Buttons:** "Create" button.

Edit Hub Form (Right):

- Name:** Input field containing "Nomad Hideout".
- Description:** Input field containing "We're a hub for digital nomads to assimilate in the nomad network of Penang".
- Type Of Hub:** Input field containing "Coworking Space".
- Public:** Checkbox (checked).
- Info:** Input field containing "Contact us at nomadhideout@gmail.com".
- State:** Input field containing "Pulau Pinang".
- Buttons:** "Save" button (white background) and "Delete" button (red background).

Figure 4.48: Hub Creation and Edit Form

The HubForm component seen in Figure 4.48 provides a robust form interface for creating and editing hub details, supporting comprehensive validation and feedback for each field. It integrates the `useValidatedForm` hook for form validation with Zod schemas and uses the `TAddOptimistic` type to handle optimistic updates, ensuring smooth user interactions. The form restricts access to administrators and hub owners, enforcing role-based permissions for creating, updating, and deleting hubs. Key fields include hub name, description, type, visibility status, and additional details, each with built-in error handling. The form is styled consistently using the Shadcn UI library, with toast notifications offering immediate feedback on actions. The hub schema, defined with Drizzle ORM and Zod, outlines relationships with users and states and sets up a flexible structure for hub management. Form state is efficiently managed with React's `useState` and `useTransition` hooks, allowing responsive and seamless submission interactions.

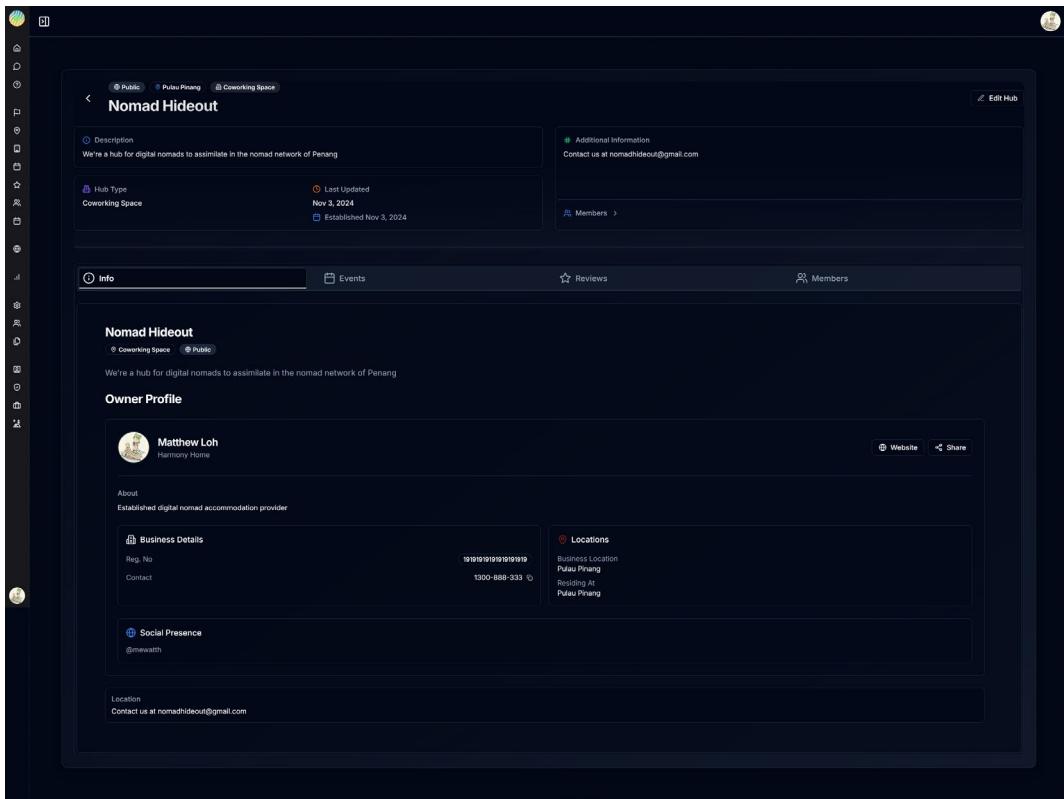


Figure 4.49: Hub View (Owner Profile Info Tab)

The Hub View seen in Figure 4.49 provides a well-organized interface for hub management, specifically designed for hub owners. Built in [hubId]/page.tsx, this main container component fetches and displays hub-related data, including associated events, reviews, and user information, with all interactions wrapped in an OptimisticHub component to offer instant UI feedback. At the top of the page, key hub details are displayed, such as the hub name (stylized with gradient text), visibility status badges (public or private), linked location information, hub type (e.g., "Coworking Space"), and an edit button accessible to hub owners. The layout is divided into four primary sections through the HubTabs component, creating a tabbed interface that includes an Info tab (currently displayed), an Events tab to showcase hub-hosted events, a Reviews tab for user reviews, and a Members tab for managing members. The Info tab reveals detailed information about the hub, including its description, the owner's profile, business contact details, and location information, along with metadata such as the hub's establishment date and last update timestamp. Additionally, the system supports real-time updates by using React Query's refetch intervals and implements optimistic UI updates, making interactions feel instantaneous and responsive for mutations specifically.

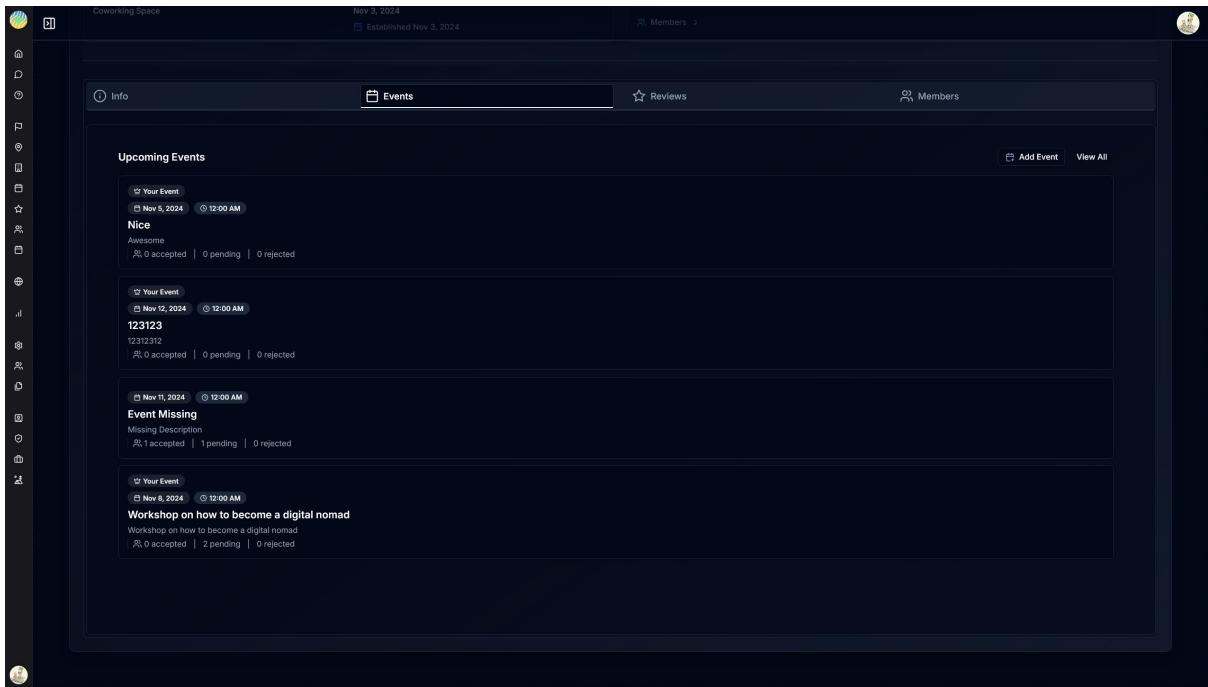


Figure 4.50: Hub View Events Tab (Hub Owner and Event Creator)

The EventList component implement in a Tab shadcn component seen in Figure 4.50 offers a comprehensive event management interface within the hub system, displaying upcoming events with their associated details and participant status. It utilizes optimistic updates for real-time owner mutations and includes role-based access control to determine user permissions for event management. Each event is displayed in a card format showing event name, date, time, and participant statistics (accepted, pending, rejected). The component supports event creation for authorized users (admins/owners) and provides detailed event information with participant management capabilities. The interface includes animations through Framer Motion and integrates with Supabase + React Query for real-time data fetching across client-side navigation.

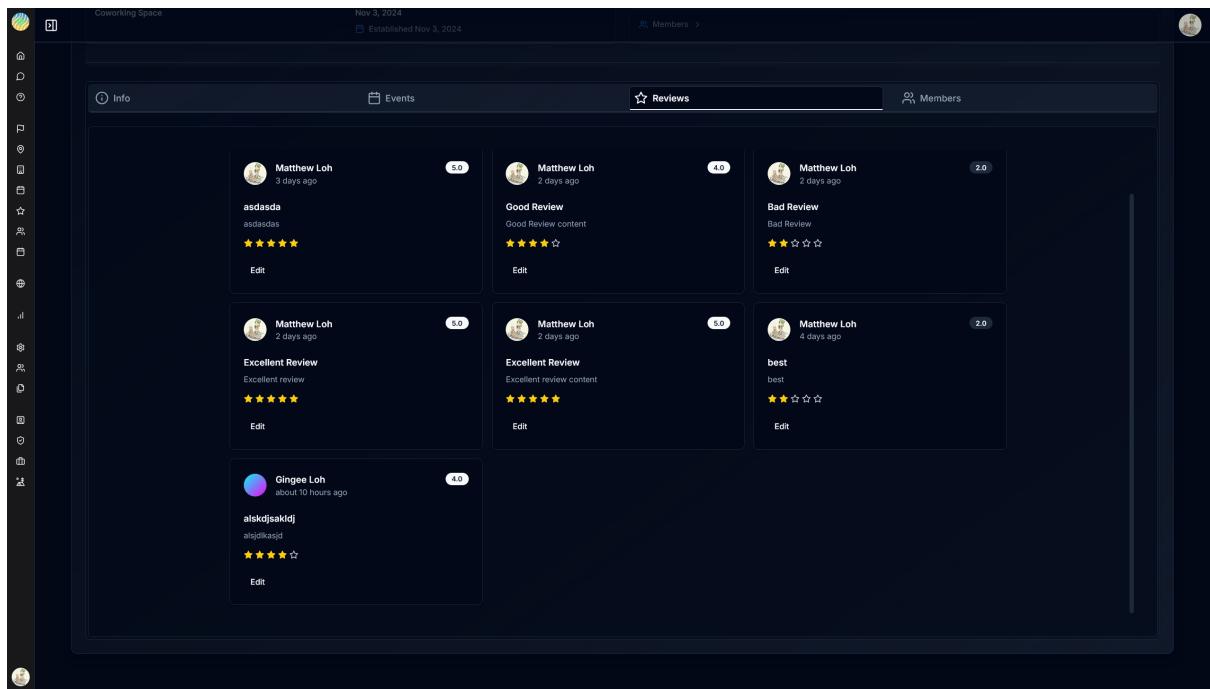


Figure 4.51: Hub Reviews Tab View

The ReviewList component implements a comprehensive review management system for hubs, displaying user reviews in a responsive grid layout. Each review is presented in a card format showing the reviewer's avatar, name, rating (displayed with star icons), review title, content, and timestamp. The component supports CRUD operations through a modal interface using ReviewForm and implements an empty state with a call-to-action for first reviews. The review cards feature interactive elements including rating badges that change color based on the score, formatted timestamps using date-fns, and edit capabilities for authorized users. The implementation includes smooth animations and hover effects, with responsive design considerations for different screen sizes through a grid-based layout system.

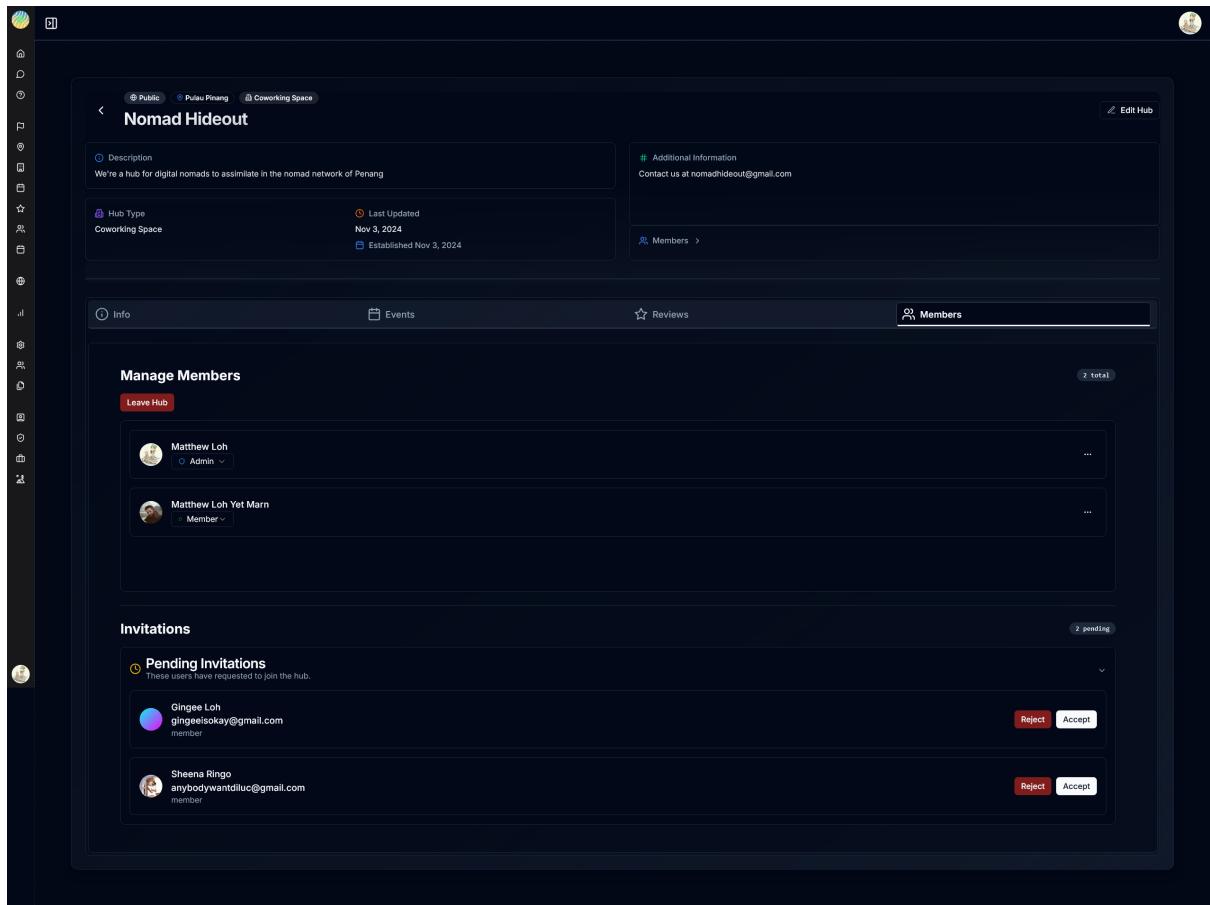


Figure 4.52: Hub Page Members View (Owner of Hub)

The UserInviteHubList component pictured in Figure 4.52 implements a sophisticated member management system for hub owners, utilizing role-based access control and complex state management through Supabase and React Query again. The component renders differently based on user roles, with hub owners seeing comprehensive management controls while regular members see a simplified view. The interface is structured into two main sections: active members management and invitation handling. For owners, it provides granular control over member roles (admin/member) and invitation statuses (pending/accepted/rejected) through a dropdown interface, while implementing optimistic updates for a responsive user experience.

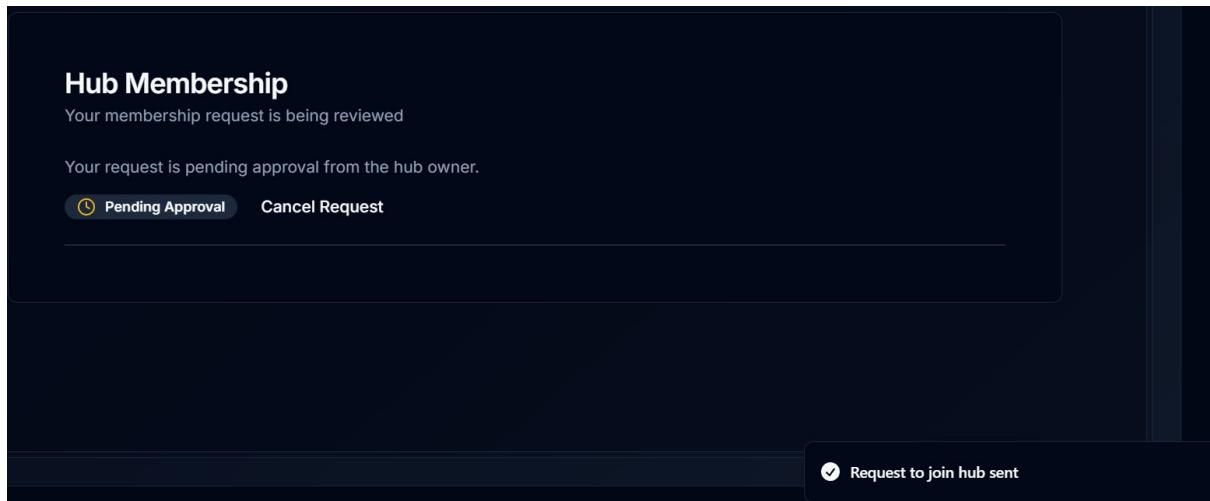


Figure 4.53: Hub Member Request Creation

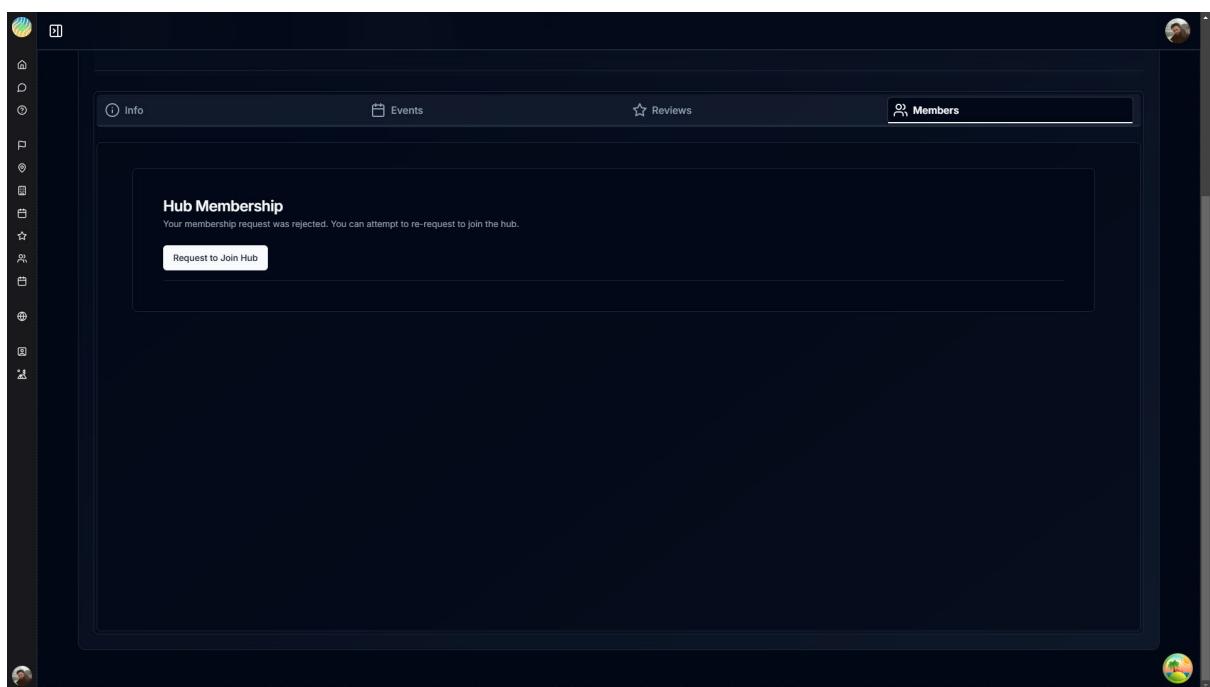


Figure 4.54: Hub Member Join Request Rejected

The components use Supabase client fetches to manage member states, handling complex scenarios like role changes, invitation responses, and membership status updates. It includes collapsible sections for pending and rejected invitations, member statistics, and role management interfaces. The implementation features error handling with toast notifications, loading states, and empty states for different scenarios. The component integrates with the broader hub management system through the HubTabs interface, maintaining consistent state

across the application while providing intuitive controls for member management through a clean, organized interface with clear visual hierarchies and status indicators.

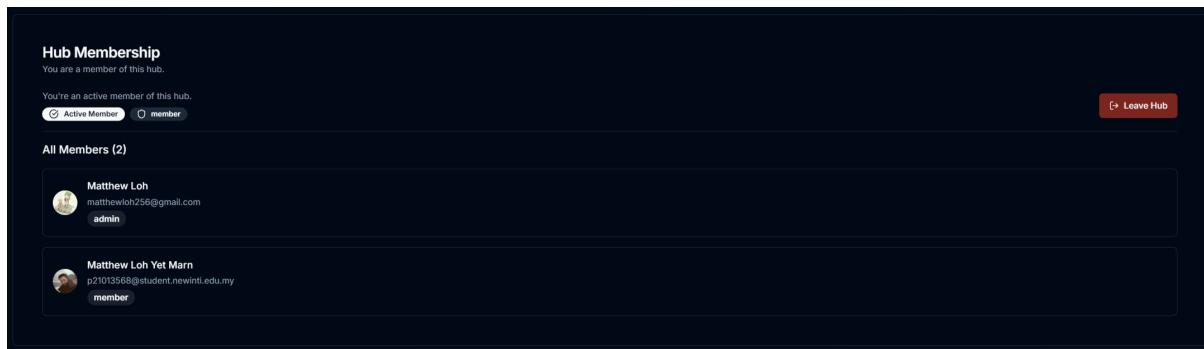


Figure 4.55: Hub Member Join Request Accepted and Leave Button

A screenshot of a "Create Event" form. The form has several input fields: "Event Name" (with placeholder "Enter event name"), "Start Date" and "End Date" (each with a "Select date" button and a calendar icon), "Description" (with placeholder "Describe your event"), "Additional Information" (with placeholder "Enter any additional information"), "Event Type" (a dropdown menu with placeholder "Select event type"), and a "Mark as Complete" checkbox. A large blue "Create" button is at the bottom right.

Figure 4.56: Create Event Form (Owner / Admin Member)

Events within hubs follow their own permission structure, where event creators (hub owners or admins) can manage participant lists, update event details, and control event visibility. The system tracks event participation through a separate join request system, maintaining counts of accepted, pending, and rejected participants. The review system allows members to provide feedback on hubs, implementing a rating system that contributes to the hub's overall reputation. All these interactions are managed through optimistic updates for better user experience, with proper error handling and state management ensuring data consistency across the platform.

The screenshot displays the 'Event View (Creator)' interface for a digital nomad workshop. The left sidebar contains navigation links for Pragmatic Features, DE Rantau Platform, DE Rantau Regions, DE Rantau States, Hubs, Hub Events, Hub Reviews, Communities, and Profile Management. A user profile for Matthew Loh is shown at the bottom left.

Event Details: Workshop on how to become a digital nomad, starting on November 8, 2024, and ending on November 14, 2024. The event type is social and organized by Matthew Loh (matthewloh25@gmail.com). Additional information: Contact us at nomadhideout@gmail.com.

Event Location: G Hotel, Gurney Plaza, 170 Persiaran Gurney, Penang, Penang 10250, Malaysia. The map shows the location in George Town, Penang, with markers for various landmarks like Tanjung Bungah, Permai Village, Tanjung Tokong, Pulau Tikus, and Butterworth.

Event Participants: Matthew Loh Yet Marn (p2101356@student.newinti.edu.my) is listed as a participant.

Invite Participants: A form to invite users, selecting Matthew Loh Yet Marn as the user and Member as the role, with a 'Invite User' button.

Figure 4.57: Event View (Creator)

Hub event markers are a key feature of the map module, representing specific event locations on the map. Each marker is associated with an event and includes details such as the venue, address, and event timings. The MapComponent renders these markers created using the Marker component from react-map-gl, and each marker is interactive, allowing users to click and view additional event details in a popover. On the creation side, a CoordinatesSelector is provided to allow 1-1 imagining of locations for each event as well as provide valid coordinates information like latitude and longitude. The built-in Geocoder search also allows easy querying of locations by addresses according to Mapbox provided data. The markers are styled with a distinct icon and color to differentiate them from other map elements, making it easy for users

to identify and explore events. This feature supports the module's goal of facilitating user engagement and exploration.

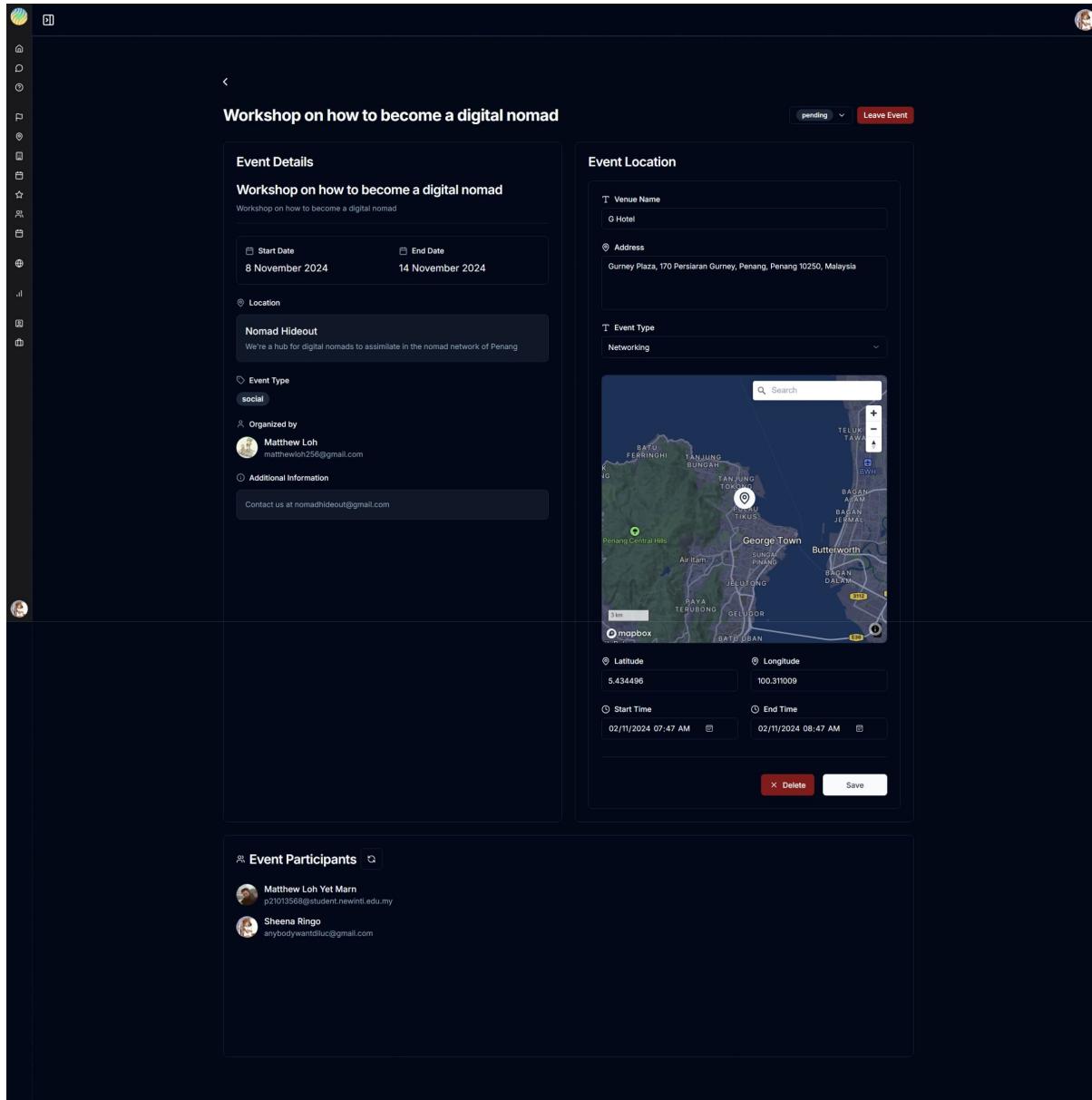


Figure 4.58: Participant View (“member” Role)

Event participants are able to join, be invited and also generate a response according to statuses like pending, accepted or rejected. Participants are only able to view other participants upon sending a join request.

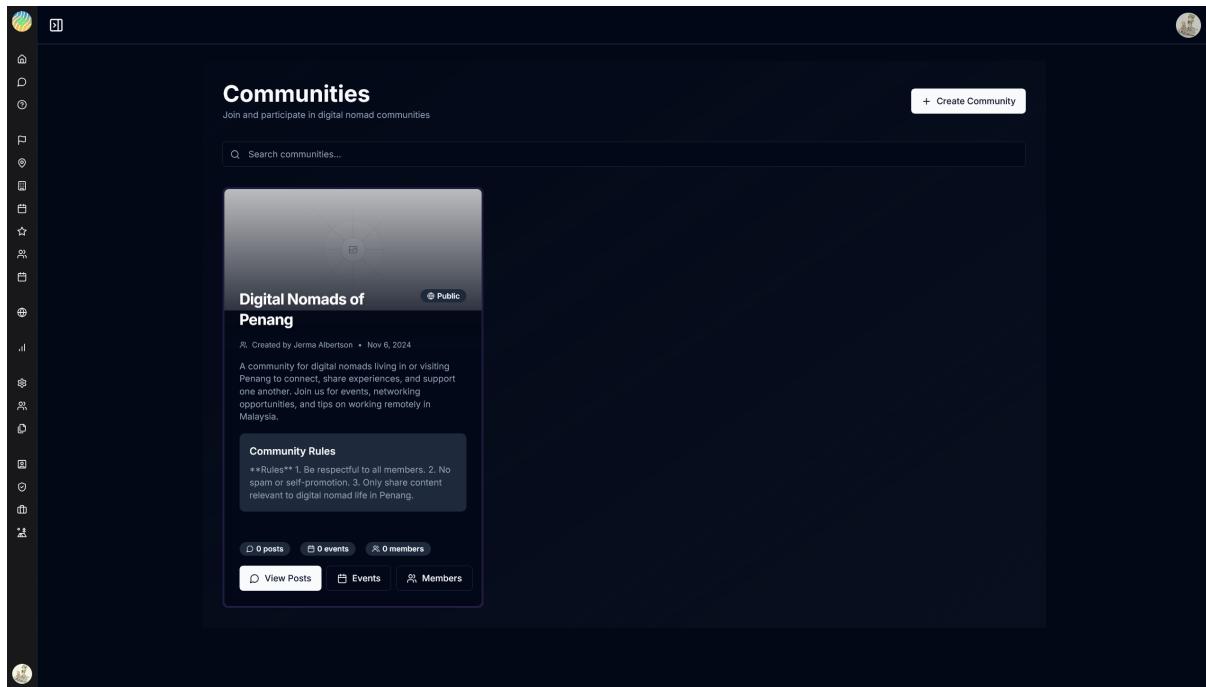


Figure 4.59: Communities List Page

The Communities module introduces a social networking feature on the Pragmatic platform, designed for digital nomads to create and join themed communities. The schema “communities.ts” structures each community with essential details like name, description, rules, and privacy settings (e.g., public or private communities), and supports a many-to-many relationship for managing members via roles (admin or member). An invitation system tracks membership requests with statuses like pending, accepted, and rejected, enhancing control over community access similar to hubs. The module offers two primary views. The Communities Listing Page (communities/page.tsx) displays a collection of available communities, allowing users to filter and create new communities. It is optimized for server-side rendering, implementing loading states and optimistic updates for a smooth and responsive experience.

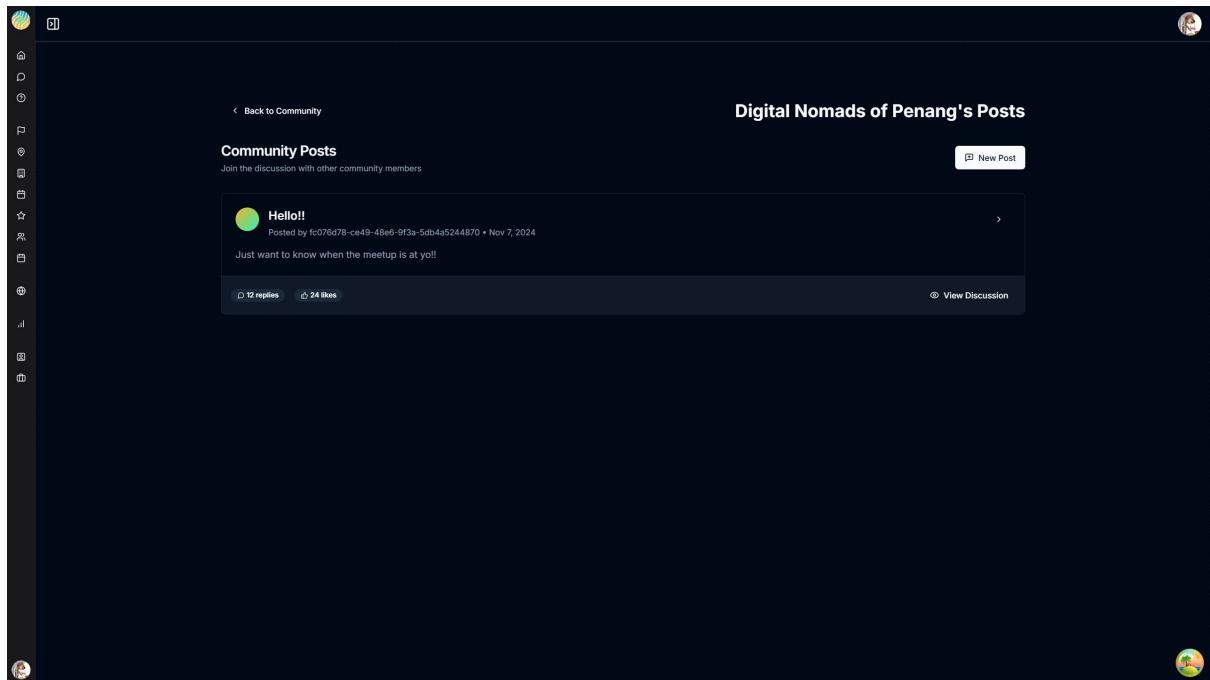


Figure 4.60: Community Posts of a Community

The Individual Community Page (`communities/[communityId]/page.tsx`) provides an in-depth view of each community, presenting details, rules, member lists with role indicators, community posts, events, and an admin-managed member interface. Business logic within this module ensures that only authenticated users can create communities. Each community can be public or private, and while members can create posts and events, admins have permissions to manage member roles and adjust community settings. The Communities module complements the hub system by offering an informal, discussion-oriented space for digital nomads to engage, fostering a community-driven environment for sharing experiences and networking.

4.5 Analytics Module

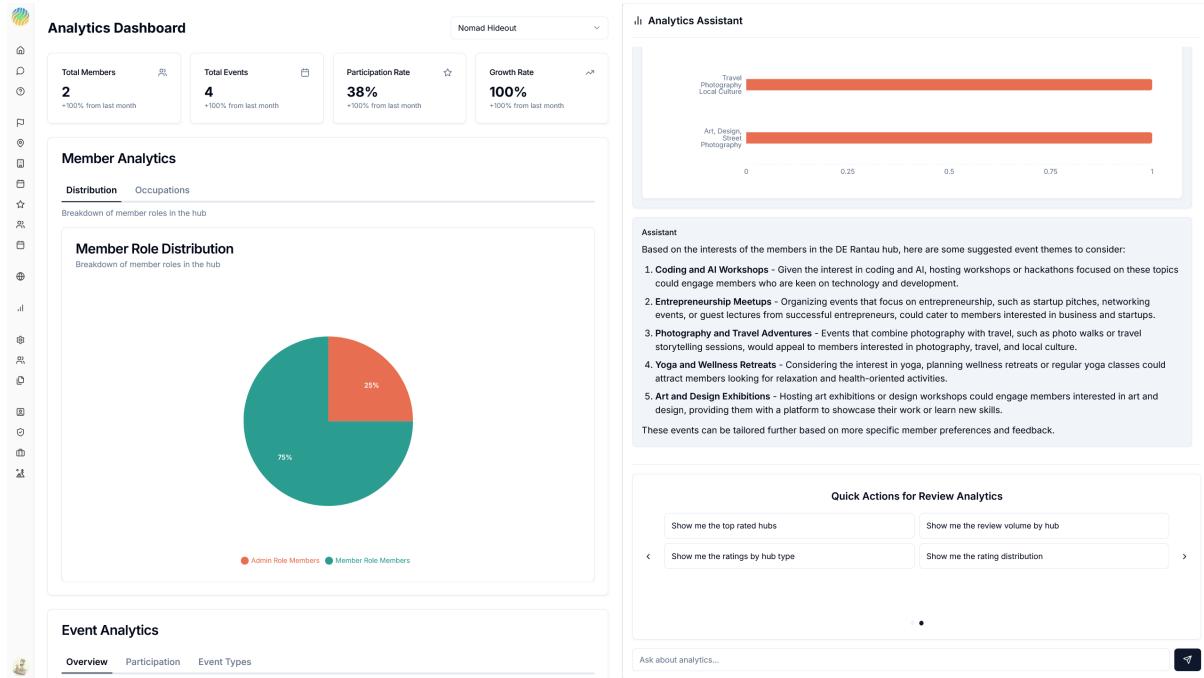


Figure 4.61: Analytics Module View for Hub Owner

The following section covers the AI-integrated analytics chat component along with the underlying processes and components that are used to generate charts and visualizations of various metrics expected from a hub.

4.5.1 Data Retrieval and Visualization



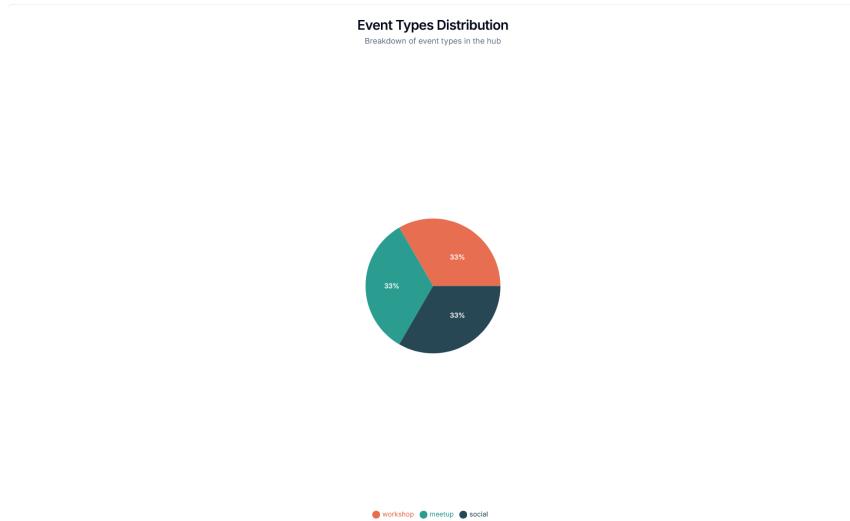


Figure 4.62: Pie Chart visualization of member role breakdown and growth details

The Analytics Dashboard implements a comprehensive data visualization and monitoring system for hub management, utilizing a combination of real-time data fetching through Supabase and dynamic chart rendering. The system's core functionality is built around the `AnalyticsDashboard` component, which manages multiple data streams including member statistics, event participation, and engagement metrics. It implements a responsive layout using `ResizablePanelGroup` for flexible viewing, with the main dashboard occupying 70% of the screen and an AI assistant chat interface taking up the remaining 30%. The dashboard fetches data through custom hooks using `useQuery` from Supabase, handling complex data relationships between hubs, members, events, and participant information. The visualization layer utilizes various chart components (`MemberDistributionChart`, `OccupationsChart`, `EventsOverviewChart`) to present data in meaningful ways, with each chart being responsive to real-time data updates. The implementation includes loading states, error handling, and optimistic updates to ensure a smooth user experience while managing large datasets and complex data relationships.

4.5.2 LLM-based Unstructured Data Processing and Suggestions Generation

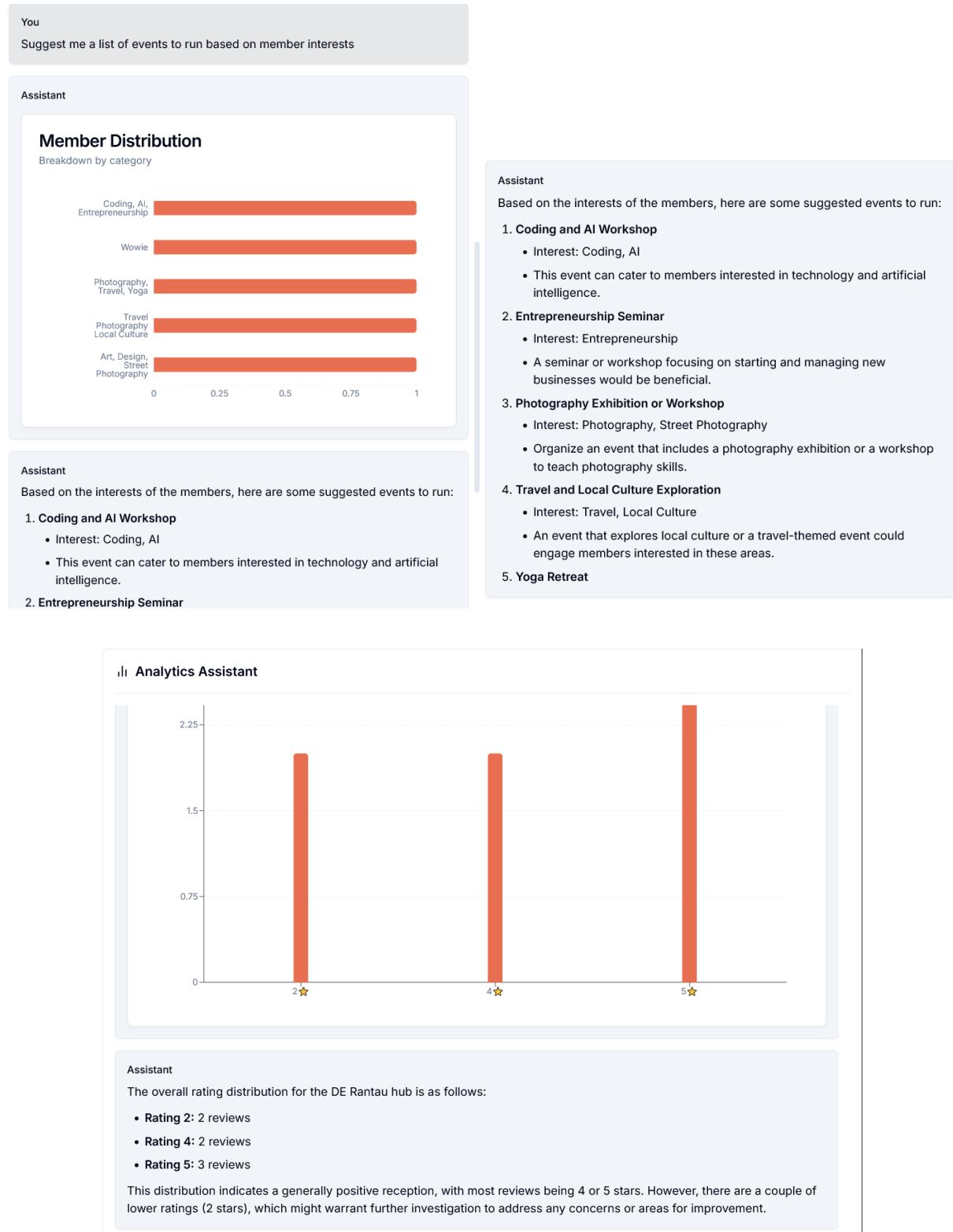


Figure 4.63: Analytics Chat supports Member and Review analytics

The Analytics Chat component (`AnalyticsChat.tsx`) represents an innovative approach to data analysis through natural language processing and AI-assisted insights generation. The system leverages the Vercel AI SDK's `useChat` hook to implement a conversational interface that can interpret user queries and generate appropriate visualizations and insights. The component includes specialized tools for processing different types of analytics requests, including member statistics (`getMemberStats`), hub metrics analysis (`hubMetricsAnalysis`), and review analysis (`getReviewAnalysis`). Each tool can generate specific visualizations based on the query context, with results rendered through specialized chart components.

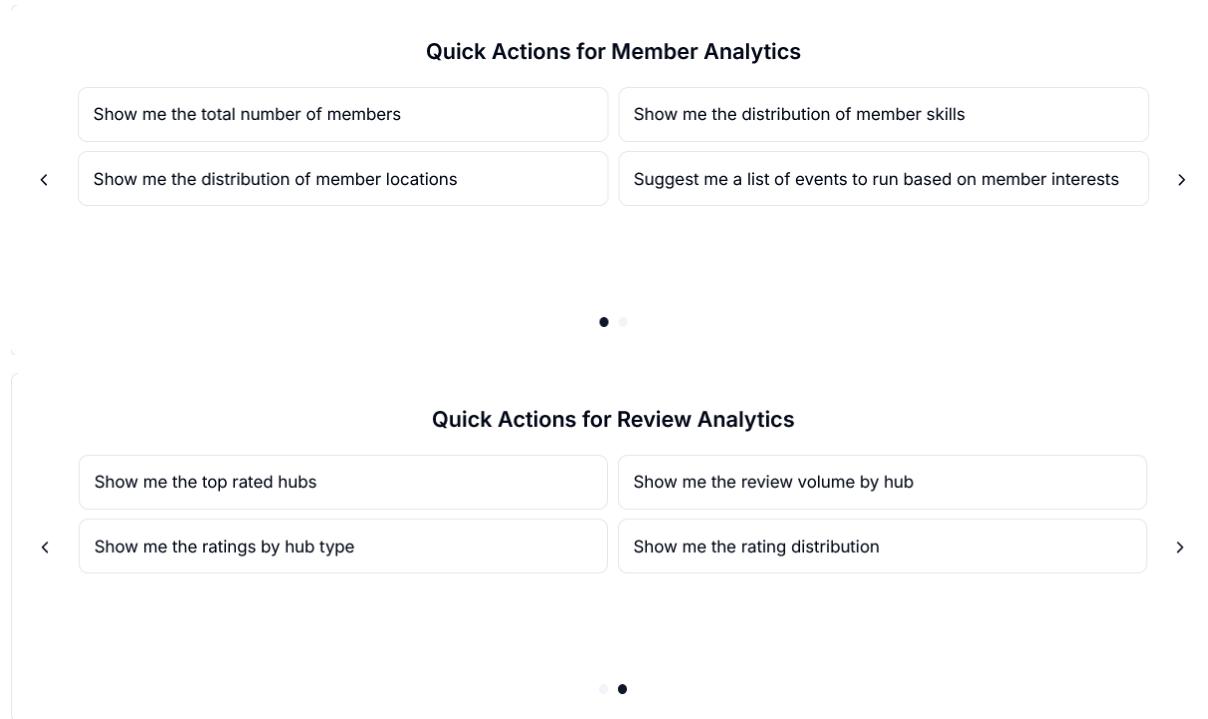


Figure 4.64: Animated Prompt Suggestions

The system implements a sophisticated suggestion system through `AnimatedPromptSuggestions`, offering contextual prompts for different types of analytics queries. The chat interface maintains conversation history and can process both structured and unstructured requests, providing a flexible way for users to explore their data through natural language interactions.

4.5.3 LLM Interactivity and Other Metrics

```
pragmatic-v1 - route.ts

export async function POST(req: Request) {
  const { messages } = await req.json()
  const {
    data: { session },
  } = await getSession()

  if (!session) {
    return new Response("Unauthorized", { status: 401 })
  }

  const result = await streamText({
    model: openai("gpt-4-turbo"),
    system: "You are an analytics assistant for DE Rantau hub owners. You help analyze member data and event participation to provide insights.",
    messages: convertToCoreMessages(messages),
    tools,
    maxSteps: 5,
    async onFinish({ text, toolCalls, toolResults, usage, finishReason }) {
      console.log("onFinish")
      // console.log(text, toolCalls, toolResults, usage, finishReason)
      // console.log(messages)
    },
  })

  return result.toDataStreamResponse()
}
```

Figure 4.65: Analytics Chatbot is built on Vercel AI SDK toolcalling

The Analytics module implements an advanced tool-calling system through a combination of OpenAI's function calling capabilities and custom-built analytics tools. The implementation in analytics/chat/route.ts establishes a streaming API endpoint that processes natural language queries through Vercel AI SDK helper UI library, while tools.ts defines a comprehensive suite of specialized analytics functions using the AI SDK's tool creation utilities. The system includes six primary tools: memberStatsTool for demographic analysis, eventStatsTool for event metrics, participationTool for engagement tracking, sentimentAnalysisToolOfHubReviews for review sentiment analysis, hubMetricsAnalysisTool for multi-dimensional hub performance analysis, and reviewAnalysisTool for detailed review statistics.

```

pragmatic-v1 - tools.ts

export const memberStatsTool = createTool({
  description: "Get member statistics and demographics",
  parameters: z.object({
    metric: z.enum(["skills", "interests", "locations", "total"]),
  }),
  execute: async function ({ metric }): Promise<string> {
    if (metric === "total") {
      const totalMembers = await db
        .select({ count: sql<number>'count(*)' })
        .from(nomadProfile)

      const response: ChartResponse = { total: totalMembers[0].count }
      return JSON.stringify(response)
    }

    const memberStats = await db
      .select({
        value:
          metric === "locations"
            ? nomadProfile.currentLocation
            : metric === "skills"
            ? nomadProfile.skills
            : nomadProfile.interests,
        count: sql<number>'count(*)',
      })
      .from(nomadProfile)
      .groupBy(
        metric === "locations"
          ? nomadProfile.currentLocation
          : metric === "skills"
          ? nomadProfile.skills
          : nomadProfile.interests,
      )

    const response: ChartResponse = [
      distribution: memberStats.map((stat) => ({
        value: stat.value || "Unknown",
        count: Number(stat.count),
      })),
    ]
    return JSON.stringify(response)
  },
})

```

Figure 4.66: memberStatsTool for the LLM to call to retrieve data

Each tool is structured with specific parameters defined through Zod schemas and executes complex SQL queries using Drizzle ORM to fetch and process data from the Supabase database. The route handler manages authentication, streams responses, and coordinates tool execution through the AI SDK's streamText function, which can chain multiple tool calls in a single conversation turn.

```

pragmatic-v1-tools.ts
export const eventStatsTool = createTool({
  description: "Get event statistics and participation data",
  parameters: z.object({
    metric: z.enum(["attendance", "types", "upcoming"]),
  }),
  execute: async function ({ metric }) {
    const query = db
      .select({
        id: hubEvents.id,
        name: hubEvents.name,
        type: hubEvents.typeOfEvent,
        startDate: hubEvents.startDate,
        participants: sql<number>`count(${usersToEvents.userId})`,
        acceptedParticipants: sql<number>`count(case when ${usersToEvents.invite_status} = 'accepted' then 1 end)`,
      })
      .from(hubEvents)
      .leftJoin(usersToEvents, eq(hubEvents.id, usersToEvents.eventId))
      .groupBy(hubEvents.id)

    const events = await query
    switch (metric) {
      case "types":
        const typeDistribution = events.reduce(
          (acc, event) => {
            acc[event.type] = (acc[event.type] || 0) + 1
            return acc
          },
          {} as Record<string, number>,
        )
        return JSON.stringify({ types: typeDistribution })
      case "upcoming":
        const upcoming = events
          .filter(e => new Date(e.startDate) > new Date())
          .map(e => ({
            name: e.name,
            date: e.startDate,
            registeredParticipants: e.participants,
            confirmedParticipants: e.acceptedParticipants,
          }))
        return JSON.stringify({ upcoming })
      default: // attendance
        return JSON.stringify({
          events: events.map(e => ({
            name: e.name,
            date: e.startDate,
            totalParticipants: e.participants,
            confirmedParticipants: e.acceptedParticipants,
          })),
        })
    }
  },
})

```

Figure 4.67: eventStatsTool to retrieve event statistics and participation date

```

pragmatic-v1-tools.ts
export const participationTool = createTool({
  description: "Get insights about member participation",
  parameters: z.object({
    type: z.enum(["active_rate", "event_frequency"]),
  }),
  execute: async function ({ type }) {
    const participation = await db
      .select({
        userId: usersToEvents.userId,
        eventCount: sql<number>`count(${usersToEvents.eventId})`,
      })
      .from(usersToEvents)
      .groupBy(usersToEvents.userId)

    if (type === "active_rate") {
      const totalUsers = participation.length
      const activeUsers = participation.filter(
        (p) => p.eventCount > 2,
      ).length
      return JSON.stringify({
        activeRate: (activeUsers / totalUsers) * 100,
        totalUsers,
        activeUsers,
      })
    }

    const avgEvents =
      participation.reduce((acc, p) => acc + p.eventCount, 0) /
      participation.length
    return JSON.stringify({ averageEventsPerUser: avgEvents })
  },
})

```

Figure 4.68: participationTool

```

pragmatic-v1-tools
export const hubMetricsAnalysisTool = createTool({
  description:
    "Analyze hub metrics across different aspects using radar chart visualization",
  parameters: z.object({
    aspects: z.array(z.string()).optional(),
    timeframe: z.enum(["week", "month", "quarter", "year"]).optional(),
  }),
  execute: async function ({ aspects, timeframe = "month" }) {
    // Default aspects if none provided
    const defaultAspects = [
      "eventParticipation",
      "memberGrowth",
      "communityEngagement",
      "skillsDiversity",
      "locationSpread",
      "interestOverlap",
    ]

    const metricsToAnalyze = aspects || defaultAspects

    // Fetch and calculate metrics
    const metrics = await Promise.all(
      metricsToAnalyze.map(async (aspect) => {
        let value = 0
        switch (aspect) {
          case "eventParticipation":
            const events = await db
              .select({
                participants: sql<number>'count(${usersToEvents.userId})',
              })
              .from(usersToEvents)
            value = events[0]?.participants || 0
            break
          case "memberGrowth":
            const members = await db
              .select({ count: sql<number>'count(*)' })
              .from(nomadProfile)
            value = members[0].count || 0
            break
          // Add other metric calculations
        }
        return {
          aspect,
          value: normalizeValue(value, 100), // Normalize to 0-100 scale
        }
      }),
    )

    return JSON.stringify({
      metrics,
      timeframe,
    })
  },
})

```

Figure 4.69: hubMetricsAnalysisTool

The tools implement sophisticated data processing, including normalization functions, statistical calculations, and data aggregation, with results formatted for direct visualization through the dashboard's charting components. This architecture enables natural language processing of analytics requests, transforming them into structured database queries and returning formatted data that can be immediately visualized, while maintaining type safety through comprehensive TypeScript definitions and proper error handling throughout the tool execution pipeline.

```

prisma/v1/tools
export const reviewAnalysisTool = createTool({
  name: "Review Analysis Tool",
  description: "Analyze member reviews and ratings with visualizations",
  parameters: z.object({
    metric: z.enum([
      "rating_distribution",
      "top_rated_hubs",
      "review_volume",
      "category_analysis"
    ]),
    hubId: z.string().optional(),
    category: z.string().optional(),
  }),
  execute: async function ({ metric, hubId, category }) {
    switch (metric) {
      case "rating_distribution": {
        const distribution = await db
          .select()
          .from(reviews)
          .groupBy(reviews.rating)
          .count(sqlCounters).count("*");
        const reviews = await db
          .leftJoin(hubs, eq(reviews.hubId, hubs.id))
          .groupBy(hubs.id, hubs.name)
          .having(sqlCounters.count() >= 3); // Only hubs with at least 3 reviews
        const hubStats = reviews.map((r) => {
          const rating = Number(r.rating);
          const count = Number(r.count);
          const avgRating = Number(r.avgRating);
          return { hubId: r.id, hubName: r.name, rating, count, avgRating };
        });
        return JSON.stringify({
          type: "rating_distribution",
          data: hubStats.map((h) => {
            const rating = h.rating;
            const count = h.count;
            const avgRating = h.avgRating;
            return { rating, count, avgRating };
          })
        });
      }
      case "hub_comparison": {
        const hubStats = await db
          .leftJoin(hubs, eq(hubs.id, reviews.hubId))
          .groupBy(hubs.id, hubs.name)
          .having(sqlCounters.count() >= 3);
        const reviews = await db
          .leftJoin(hubs, eq(reviews.hubId, hubs.id))
          .groupBy(hubs.id, hubs.name)
          .having(sqlCounters.count() >= 3); // Only hubs with at least 3 reviews
        const stats = reviews.map((r) => {
          const hub = hubStats.find((h) => h.id === r.hubId);
          const rating = Number(r.rating);
          const count = Number(r.count);
          const avgRating = Number(r.avgRating);
          return { hubId: r.hubId, hubName: hub.name, rating, count, avgRating };
        });
        return JSON.stringify({
          type: "hub_comparison",
          data: stats.map((h) => {
            const rating = h.rating;
            const count = h.count;
            const avgRating = h.avgRating;
            return { rating, count, avgRating };
          })
        });
      }
      case "top_rated_hubs": {
        const topHubs = await db
          .select()
          .from(hubs)
          .groupBy(hubs.id, hubs.name)
          .having(sqlCounters.count() >= 3)
          .orderBy(sqlCounters.count().desc());
        const reviews = await db
          .leftJoin(hubs, eq(reviews.hubId, hubs.id))
          .groupBy(hubs.id, hubs.name)
          .having(sqlCounters.count() >= 3)
          .orderBy(sqlCounters.avg(reviews.rating).desc());
        const topHubsData = reviews.map((r) => {
          const hub = topHubs.find((h) => h.id === r.hubId);
          const rating = Number(r.rating);
          const count = Number(r.count);
          const avgRating = Number(r.avgRating);
          return { hubId: r.hubId, hubName: hub.name, rating, count, avgRating };
        });
        console.log("Top hubs", topHubsData);
        const formatted = JSON.stringify({
          type: "top_rated_hubs",
          data: topHubsData.map((h) => {
            const rating = h.rating;
            const count = h.count;
            const avgRating = h.avgRating;
            return { rating, count, avgRating };
          })
        });
        return formatted;
      }
      case "review_volume": {
        const volumeData = await db
          .select()
          .from(hubs)
          .groupBy(hubs.id, hubs.name)
          .having(sqlCounters.count() >= 3)
          .orderBy(sqlCounters.count().desc());
        const reviews = await db
          .leftJoin(hubs, eq(reviews.hubId, hubs.id))
          .groupBy(hubs.id, hubs.name)
          .having(sqlCounters.count() >= 3)
          .orderBy(sqlCounters.sum(reviews.rating).desc());
        const volumeData = reviews.map((r) => {
          const hub = volumeData.find((h) => h.id === r.hubId);
          const rating = Number(r.rating);
          const count = Number(r.count);
          const sumRating = Number(r.sumRating);
          return { hubId: r.hubId, hubName: hub.name, rating, count, sumRating };
        });
        return JSON.stringify({
          type: "review_volume",
          data: volumeData.map((h) => {
            const rating = h.rating;
            const count = h.count;
            const sumRating = h.sumRating;
            return { rating, count, sumRating };
          })
        });
      }
      case "category_analysis": {
        const categoryStats = await db
          .select()
          .from(categories)
          .groupBy(categories.id, categories.name)
          .having(sqlCounters.count() >= 3)
          .orderBy(sqlCounters.sum(reviews.rating).desc());
        const reviews = await db
          .leftJoin(hubs, eq(reviews.hubId, hubs.id))
          .groupBy(hubs.id, hubs.name)
          .having(sqlCounters.count() >= 3)
          .orderBy(sqlCounters.sum(reviews.rating).desc());
        const stats = reviews.map((r) => {
          const category = categoryStats.find((c) => c.id === r.categoryId);
          const rating = Number(r.rating);
          const count = Number(r.count);
          const sumRating = Number(r.sumRating);
          return { categoryId: r.categoryId, categoryName: category.name, rating, count, sumRating };
        });
        return JSON.stringify({
          type: "category_analysis",
          data: stats.map((c) => {
            const rating = c.rating;
            const count = c.count;
            const sumRating = c.sumRating;
            return { rating, count, sumRating };
          })
        });
      }
    }
  }
});

```

Figure 4.70: Review Analysis Tool, the most complex tool

The implementation includes sophisticated data processing pipelines that aggregate information from multiple sources to generate actionable insights such as suggesting events based on the demographic information like occupation and interests. This component integrates with the broader analytics system to provide a comprehensive view of hub performance and member engagement patterns.

```
pragmatic-v1 - engagement-metrics.tsx

export function EngagementMetrics({ analytics }: EngagementMetricsProps) {
  if (!analytics) return null

  const metrics = {
    totalMembers: analytics.users_to_hubs.filter(
      (u) => u.invite_status === "accepted",
    ).length,
    activeEvents: analytics.hub_events.filter((e) => !e.is_complete).length,
    participationRate: calculateParticipationRate(analytics.hub_events),
    memberGrowth: calculateGrowthRate(analytics.users_to_hubs),
  }

  return (
    <div className="grid gap-4 md:grid-cols-2 lg:grid-cols-4">
      <MetricCard
        title="Total Members"
        value={metrics.totalMembers}
        description="Active members in the hub"
      />
      <MetricCard
        title="Active Events"
        value={metrics.activeEvents}
        description="Currently running events"
      />
      <MetricCard
        title="Participation Rate"
        value={`${metrics.participationRate}%`}
        description="Average event participation"
      />
      <MetricCard
        title="Member Growth"
        value={`${metrics.memberGrowth}%`}
        description="Last 30 days"
      />
    </div>
  )
}
```

Figure 4.71 Engagement Metrics Card

```

pragmatic-v1 - calculations.ts

import { type HubAnalytics } from "./types"
import { subDays, parseISO, isAfter } from "date-fns"

export function calculateParticipationRate(
  events: HubAnalytics["hub_events"],
): number {
  if (!events || events.length === 0) return 0

  const totalParticipants = events.reduce((acc, event) => {
    const acceptedParticipants =
      event.users_to_events?.filter((user) => user.pending === false)
      .length ?? 0
    const totalInvites = event.users_to_events?.length ?? 0
    return (
      acc + (totalInvites > 0 ? acceptedParticipants / totalInvites : 0)
    )
  }, 0)

  return Math.round((totalParticipants / events.length) * 100)
}

export function calculateMonthlyChange(
  data: HubAnalytics["hub_events"] | HubAnalytics["users_to_hubs"],
): number {
  if (!data || data.length === 0) return 0

  const now = new Date()
  const thirtyDaysAgo = subDays(now, 30)

  const currentPeriod = data.filter((item) => {
    const date = parseISO(item.created_at)
    return isAfter(date, thirtyDaysAgo)
  }).length

  const previousPeriod = data.filter((item) => {
    const date = parseISO(item.created_at)
    return !isAfter(date, thirtyDaysAgo)
  }).length

  if (previousPeriod === 0) return currentPeriod > 0 ? 100 : 0

  return Math.round(((currentPeriod - previousPeriod) / previousPeriod) * 100)
}

export function calculateGrowthRate(
  members: HubAnalytics["users_to_hubs"],
): number {
  if (!members || members.length === 0) return 0

  const now = new Date()
  const thirtyDaysAgo = subDays(now, 30)

  const newMembers = members.filter((member) => {
    const joinDate = parseISO(member.created_at)
    return (
      isAfter(joinDate, thirtyDaysAgo) &&
      member.invite_status === "accepted"
    )
  }).length

  const previousMembers = members.filter((member) => {
    const joinDate = parseISO(member.created_at)
    return (
      !isAfter(joinDate, thirtyDaysAgo) &&
      member.invite_status === "accepted"
    )
  }).length

  if (previousMembers === 0) return newMembers > 0 ? 100 : 0

  return Math.round((newMembers / previousMembers) * 100)
}

```

Figure 4.72: Calculations and formulas for metrics

Finally, the remainder of the analytics module implements an engagement tracking and analysis system through the `EngagementMetrics` component and associated calculation and visualization tools. This system processes live data to calculate key performance indicators including member participation rates, event success metrics, and community engagement levels.

4.6 Deployment Readiness through Vercel and Supabase

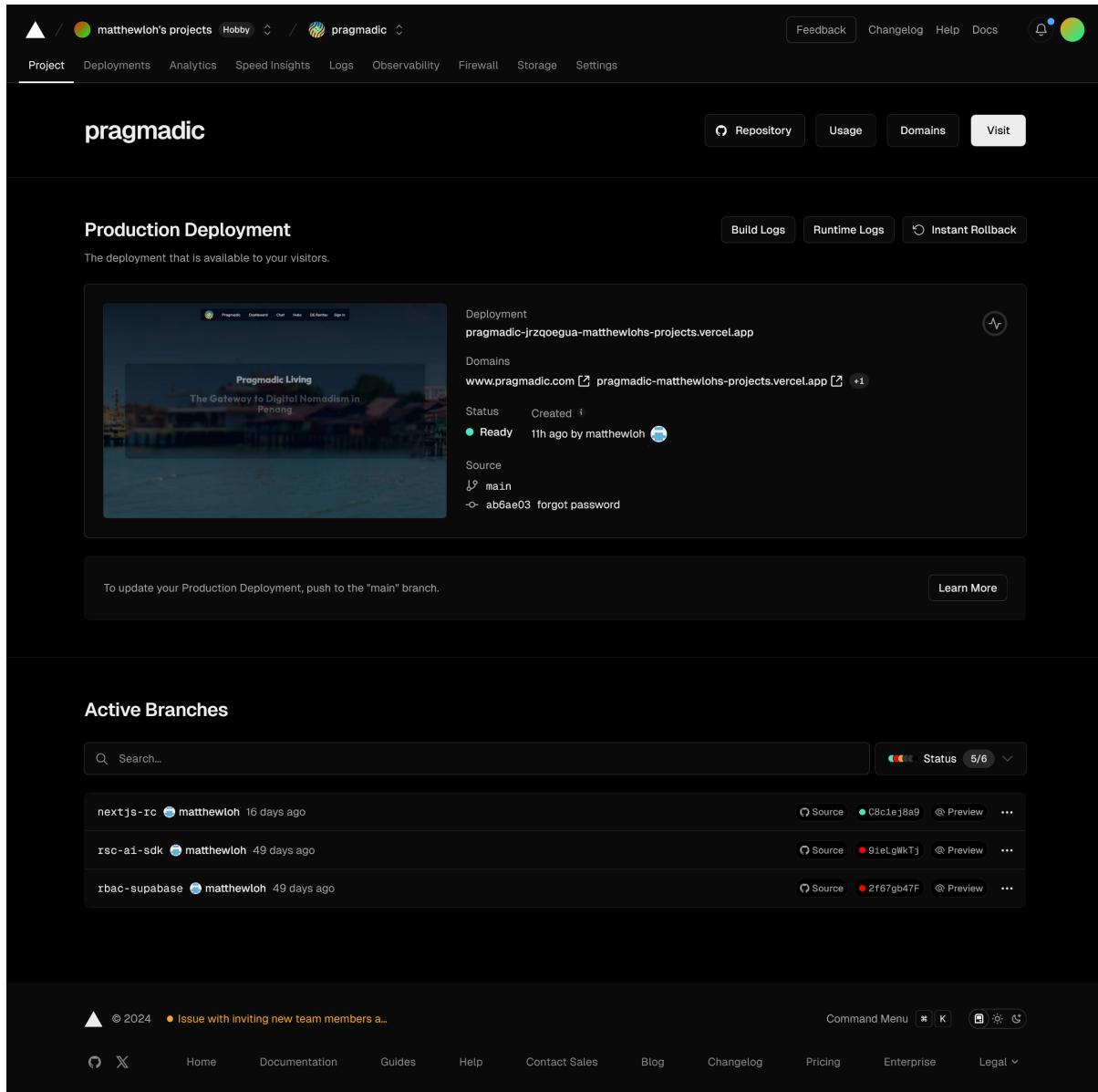


Figure 4.73: Pragmatic Project on Vercel Hobby Tier

Vercel provides a comprehensive deployment platform perfectly suited for the Pragmatic application's Next.js architecture, offering seamless integration with GitHub for continuous deployment. When code is pushed to the repository, Vercel automatically triggers builds, runs tests if any, and deploys to production or preview environments, with automatic branch deployments for collaborative development. The platform handles critical infrastructure concerns including automatic SSL/TLS certificate management through Let's Encrypt, HTTPS enforcement, and edge network distribution through their global CDN. For www.pragmatic.com which was purchased for a year, Vercel manages domain configuration,

DNS settings, and automatic certificate renewal, while providing real-time deployment logs, rollback capabilities, and environment variable management. The platform's serverless architecture aligns perfectly with Next.js's server components and API routes, automatically scaling based on demand without manual configuration.

However, this deep integration comes with vendor lock-in concerns, as migrating away from Vercel would require significant architectural changes, especially regarding API routes and edge functions. The platform's optimization for Next.js makes alternatives less attractive, potentially limiting future flexibility in framework choices. Therefore, portability could be a concern depending on deployment standards, but for a solo project, it should suffice.

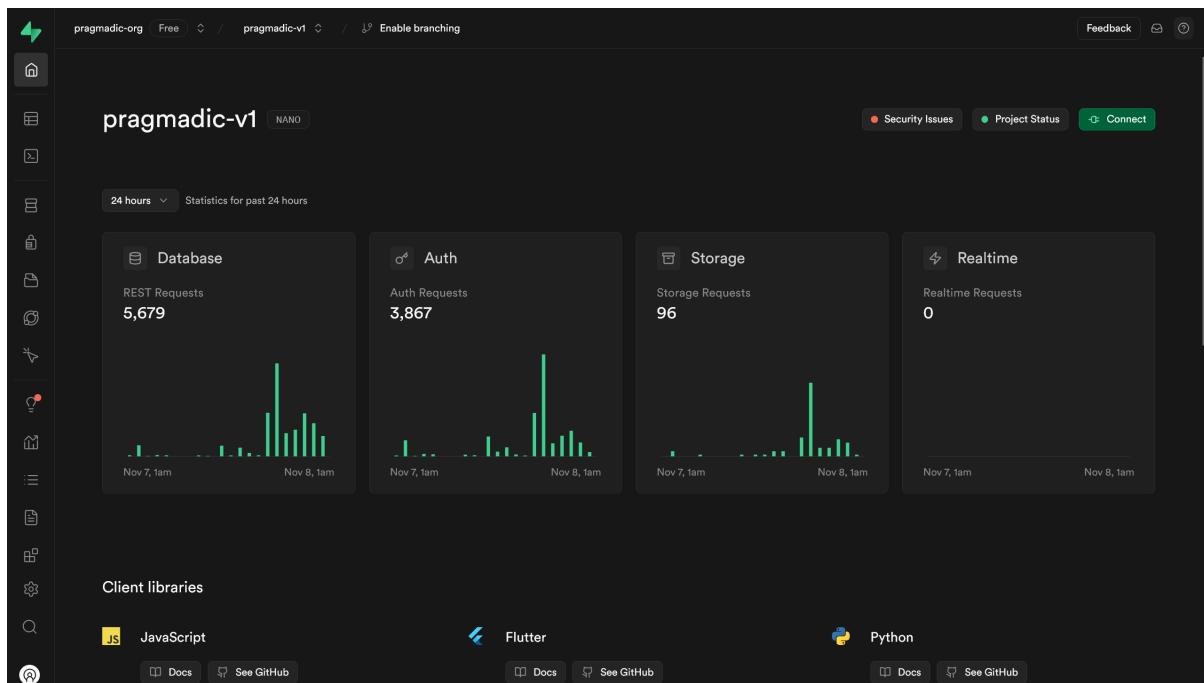


Figure 4.74: Pragmatic Supabase dashboard on Hobby Tier

Supabase provided significant advantages as a managed PostgreSQL platform throughout Pragmatic's implementation, offering a comprehensive suite of features that would otherwise require extensive self-hosted infrastructure. The platform handles critical security concerns including automatic TLS encryption for all connections, database encryption at rest using AES-256, and automated backup management. For production deployments and on the paid tier, Supabase manages database replication, point-in-time recovery, and automatic failover. Nevertheless, crucial features like providing row-level security (RLS) policies and AI assistants that integrate seamlessly with the application's authentication system along with a generous hobby tier of usage was crucial to ensure the development could finish on time. The

platform's real-time capabilities are particularly valuable for Pragmatic's collaborative features, offering WebSocket connections that are automatically secured and scaled.

Compared to self-hosting, Supabase massively eliminates the operational overhead of managing PostgreSQL clusters, handling security patches, and maintaining high availability configurations. The platform's integration with Vercel simplifies deployment through environment variable synchronization and Kong's API gateway communication. While self-hosting would provide complete control over the infrastructure, the engineering effort required to match Supabase's feature set being limited to the authentication system, storage solution, and real-time subscriptions would be substantial. The main consideration for production use is data residency, which Supabase addresses through region selection and compliance certifications. The service's pricing model scales with usage that exceeds the free tier, making it cost-effective for growing applications while maintaining enterprise-grade security features that would be complex to implement in a self-hosted environment.

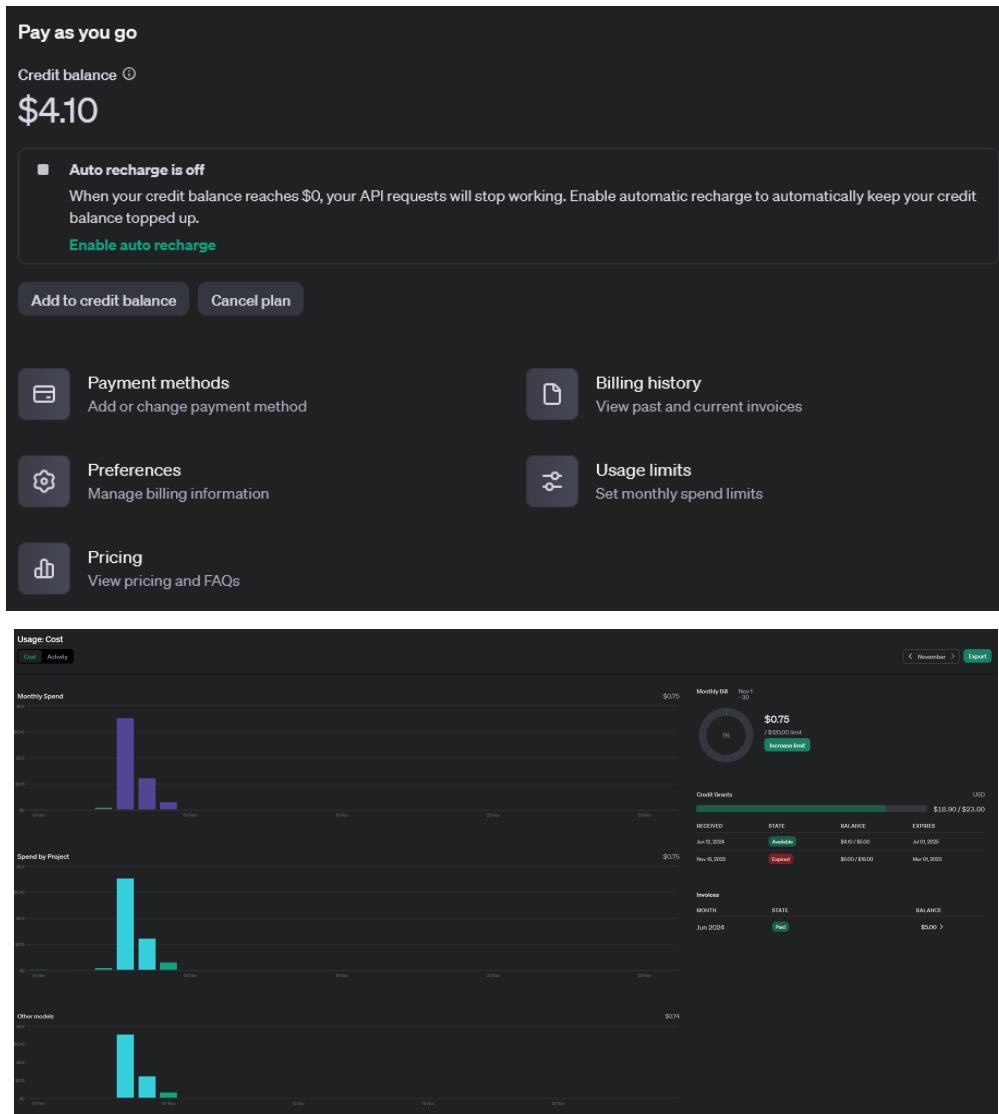


Figure 4.75: OpenAI Platform Billing and Usage Dashboards

The current implementation of Pragmatic utilizes a multi-model approach with OpenAI's GPT-4, Anthropic's Claude, and Google's Gemini, but the emergence of Llama 3.2 and other open-source models presented interesting alternatives. Local LLMs could potentially offer reduced latency, enhanced privacy, and cost benefits, but require significant infrastructure investment and optimization. Llama 3.2 at higher billions of parameters continues to demonstrate competitive performance against commercial models while allowing full control over inference and fine-tuning, particularly valuable for Pragmatic's domain-specific analytics and community management features. However, the trade-off lies in deployment complexity and resource requirements - local models need substantial GPU resources and optimization for production performance. While commercial APIs provide immediate scalability and reliability, they incur significant costs at scale and pose data privacy considerations. The optimal approach

might involve a hybrid system: using local models for sensitive operations and basic queries while leveraging commercial APIs for complex analytics and natural language understanding. This strategy would balance performance, cost, and privacy while maintaining the flexibility to adapt as open-source models continue to improve. Due to hardware limitations, Ollama was not able to be implemented given constraints to the resources provided.

4.7 Conclusion

This chapter wraps up the extensive amounts of development to produce a production-ready web application that attempts to fulfil the primary objectives set out for this project in the previous chapters. All ideas came to fruition as the Chatbot, Onboarding Support and Analytics modules were developed to a minimum viable product. These modules are flanked by a steady and secure role-based access control implementation and authentication flow to allow careful planning of database schema and Supabase integration. Throughout the development process, Vercel and Supabase as managed services proved extremely effective in allowing a single developer to magnify the capabilities of productivity and feature completion. In the next chapter, Black Box Testing, Integration Testing and Security Testing for the chatbot feature will be carried out and documented with proof of test case results.

CHAPTER 5: TESTING AND EVALUATION

5.1 Introduction

The testing and evaluation phase of Pragmatic marks a comprehensive assessment of the platform's core functionalities, integration points, and evaluation of security posture. This chapter details the approach to testing the application's core components, with particular emphasis on the unique challenges presented by the integration of Large Language Models (LLMs) with traditional web functionalities. The testing was carried out to strategically address three main areas: integration testing of core modules, validation of LLM-augmented features (analytics, chat, and security evaluation of the combined web and AI components. Special attention is paid to the RAG (Retrieval-Augmented Generation) implementation black box testing, which serves as a crucial bridge between the original objective's document management system and its AI capabilities and effectiveness. The testing methodology employs a combination of black-box testing, integration testing, and security evaluation techniques to ensure the platform's reliability, performance, and security. Through this multi-faceted testing approach, the system kills two birds with one stone in not only the technical functionality but also ensuring the user experience and data handling aspects of the platform meets expectations of production-readiness.

5.2 Black Box Testing

5.2.1 Login and Registration Module Black Box Testing

Table 5.1: Login and Registration Black Box Testing Table

MODULE: LOGIN AND REGISTRATION				
Test Scenario	Test Case	Test Steps	Expected Outcome	Security Validation / Results
- Email and Password Credentials Sign Up Flow	Valid Email and Password Credentials Registration	1. Enter valid first/last name 2. Enter valid email 3. Enter matching passwords 4. Submit form	✓ Success message ✓ Email verification sent ✓ Redirect to verification page	Password meets complexity (ZPasswordSchema) Appendix I

MODULE: LOGIN AND REGISTRATION				
Test Scenario	Test Case	Test Steps	Expected Outcome	Security Validation / Results
	Password Validation	<ol style="list-style-type: none"> 1. Test passwords against schema: - Uppercase 2. Lowercase 3. Number 4. Special char 5. Length 8-72 	<p>✓ Form validation errors for each rule violation</p>	Password strength enforcement Appendix I
	Duplicate Email	Attempt registration with existing email	<p>✓ Resend login email for PKCE flow</p>	No account enumeration Appendix II
OAuth Flow	Google Login	Click “Continue with Google”	<p>✓ OAuth popup</p> <p>✓ Successful redirect</p> <p>✓ Profile data synced</p>	- Valid OAuth state- Token security Appendix III
	GitHub Login	Click “Continue with GitHub”	<p>✓ OAuth popup</p> <p>✓ Successful redirect</p> <p>✓ Profile data synced</p>	- Valid OAuth state- Token security Appendix III
Login Flow	Valid Credentials	Enter correct email/password	<p>✓ Successful login</p> <p>✓ Correct redirect to next URL</p>	- Session token created CSRF protection
	Invalid Credentials	Enter incorrect password	<p>✓ Generic “Invalid credentials” message</p>	No specific error details Appendix IV
	Rate Limiting	Multiple failed attempts	<p>✓ Temporary account lockout</p>	Brute force protection
Session Management	Session Persistence	Check session after login	<p>✓ Session maintained across tabs</p>	Secure session storage

MODULE: LOGIN AND REGISTRATION				
Test Scenario	Test Case	Test Steps	Expected Outcome	Security Validation / Results
	Session Expiry	Wait for session timeout	✓ Redirect to login	Token expiration handled (done by Supabase)
Authorization	Role Access	Access role-specific routes	✓ Correct access for role ✓ Blocked unauthorized access	Role-based access control (Unable to access admin page)
Magic Link	Email Link	Request magic link login	✓ Email received ✓ Link works once ✓ Secure redirect	Link expiration One-time use Appendix V

5.2.2 Getting Started Module Black Box Testing

Table 5.2: Getting Started Module Black Box Testing Table

MODULE: GETTING STARTED AND PROFILE CREATION				
Test Scenario	Test Case	Test Steps	Expected Outcome	Security Validation / Results
Basic Profile Setup	Required Fields	Submit empty display name Submit without profile image	✓ - Validation errors ✓ Form not submitted ✓ Error messages displayed	Field validation Required field handling carried out Appendix VI
	Image Upload	Upload valid image Attempt to Upload invalid format3. Upload large file	✓ - Success for valid image ✓ Error for invalid formats ✓ Size limit enforcement	File validation Storage integration Appendix VI
Role Selection	Single Role Selection	Select Digital Nomad role	✓ - Role saved- Redirect to nomad profile form	Role state update Navigation flow Appendix VII
	Multiple Role States	Test existing roles Test incomplete profiles Test role switching	✓ Correct role badges ✓ Proper status indicators ✓ Profile completion prompts	Role persistence State management Appendix VII
Nomad Profile	Profile Creation	Complete nomad profile form	✓ Profile saved ✓ Skills/interests stored ✓ Location data saved	Data validation Form submission Appendix VII

MODULE: GETTING STARTED AND PROFILE CREATION				
Test Scenario	Test Case	Test Steps	Expected Outcome	Security Validation / Results
Hub Owner Profile	Business Validation	Enter business details Submit registration info Add contact details	✓ Business profile created ✓ Registration validated ✓ Contact info stored	Business data validation Contact format validation Appendix VIII
Navigation Flow	Step Progression	Basic info → Role → Profile Test back navigation Test skip optional fields	✓ Correct step highlighting ✓ State preservation ✓ Optional field handling	Navigation state Data persistence Appendix VIII

5.2.3 RAG Chatbot Module Black Box Testing

Table 5.3: RAG Chatbot Module Black Box Testing Table

MODULE: RAG CHATBOT			
Test Category	Test Query/Action	Expected Behavior	Validation Points
Basic Chat Functionality	“Hello” “What is DE Rantau?” Multi-turn conversation	✓ Appropriate greetings ✓ Contextual responses Conversation history maintained	Response coherence Context retention UI updates Appendix IX
Document Retrieval	- “What are the requirements for DE Rantau visa?”- “Tell me about coworking spaces in Penang”	✓ Relevant document chunks retrieved ✓ Citations included ✓ Accurate information from docs	Document selection Citation accuracy Response relevance Appendix IX

MODULE: RAG CHATBOT			
Test Category	Test Query/Action	Expected Behavior	Validation Points
Multi-Modal Input	Upload PDF while asking question Attach image with query- Multiple file types	✓ File upload success ✓ Proper file handling ✓ Correct file preview	Upload functionality File type handling UI feedback Appendix IX
Model Selection	Switch between models Test each model response Compare response quality	✓ Model switch successful ✓ Different response styles ✓ Performance differences	Model switching UI Response Variations Appendix IX
Error Handling	Empty queries Invalid file types Network disconnection Rate limiting	✓ Appropriate error messages ✓ Graceful degradation ✓ Rate limit notifications	Error messaging Recovery behavior User feedback Appendix IX
Context Window	Long conversations Large document selection Complex queries	✓ Context window management ✓ Relevant information retention ✓ Performance maintenance	Memory management Response quality System stability Appendix IX
UI/UX Testing	Responsive design Loading states Chat scroll behavior	✓ Smooth animations ✓ Clear loading indicators- Proper scroll management	Visual feedback User experience (Vercel AI SDK)

5.3 Integration Testing

Integration testing for the Pragmadic platform focuses on validating the complex interactions between multiple system components, particularly the challenges of integrating traditional web functionalities with LLM-powered features. The testing strategy addresses three primary domains: hub management, analytics processing, and LLM-augmented interactions. The hub

module testing validates the complete lifecycle of hub creation, event management, and member participation, ensuring data consistency and real-time updates across the system. Analytics integration testing focuses on data flow, visualization accuracy, and tool-calling mechanisms that power the AI-assisted analytics features. A comprehensive approach ensures that all system components work together seamlessly while maintaining performance, security, and reliability standards. The integration testing strategy used in Pragmatic particularly emphasizes the validation of asynchronous operations, state management, and real-time data synchronization across the platform's distributed architecture.

5.3.1 Hub Module Integration Testing

5.3.1(a) Hub Creation Flow

The hub creation flow integration testing focuses on verifying the complete lifecycle of hub creation, role assignment, and member management. The testing process begins with validating the hub owner profile creation, which is a prerequisite for hub creation. The process implemented test cases that verify the data consistency between the hubOwnerProfiles table and the users table, ensuring proper foreign key relationships. The testing carried out includes validation of the hub creation form (HubForm.tsx), checking required fields, and location data processing. Critical test paths include verifying the role-based access control during hub creation, ensuring only authenticated users with 'owner' roles can create hubs. The tests ran through validated optimistic updates through the useOptimisticHubs hook, ensuring the UI reflects changes immediately while maintaining data consistency with the backend. Integration points with Supabase are thoroughly tested, including asynchronous state management and proper error handling during database operations. Additionally, the process tested the integration between hub creation and analytics initialization, verifying that new hubs are properly tracked in the analytics system from creation.

5.3.1(b) Events and Hub Integration

The events and hub integration testing examines the relationship between hubs and each of the associated events, focusing on the bidirectional data flow and state management. The test suite validates the event creation process within hubs using EventForm.tsx, ensuring proper association with parent hubs and correct permission inheritance. We implement comprehensive testing of the event management lifecycle, including creation, updates, and deletion, verifying that changes properly cascade through the system with no race conditions. The testing includes

validation of the HubEventsList component's integration with data updates, ensuring that event changes are reflected immediately across all connected clients using proper invalidation and refetching. Particular attention is paid to testing the integration between event participation tracking and analytics data collection, verifying that metrics are accurately updated when members join or leave events. The test suite includes validation of event visibility rules based on role-based membership, ensuring that hub events are only visible to members. This integration testing also simultaneously covered the relationship between events and the hub's analytics dashboard, verifying that event metrics properly contribute to hub performance indicators.

5.3.1(c) Events, Markers and Participation Integration Testing

The integration testing for events, markers, and participation focuses on the geographical and interactive aspects of the hub system. The test suite validates the complete flow from event creation to marker placement on the map, ensuring proper geocoding and location data handling. Tests are implemented for the marker creation process, verifying that event locations are accurately represented on the map and that marker data is properly synchronized with event information at a one-to-one relationship. The testing also includes validation of the participation management system, ensuring that member RSVPs are properly tracked and reflected in both the event data and analytics metrics. The integration between the map component and event management system is crucial, verifying that marker interactions properly trigger event information displays on the MapSidebar and participation options are provided according to settings.

5.3.2 Hub Analytics Integration Testing

5.3.2(a) Analytics Data Flow Testing

The analytics data flow testing focuses on measuring and validating the performance of analytics updates across the system. The testing carried out made checks at critical data flow points, starting from user actions (like event creation or member joins) through to analytics dashboard updates. The latency in the calculation pipeline was roughly measured and guesstimated in order to find the best middle ground for cache times of data on the frontend, particularly in calculations.ts where metrics like calculateParticipationRate and calculateGrowthRate are heavy computations. The test suite heavily relies on asynchronous state management using the cache helper library for better performance and reliability. All

these efforts attempt to ensure that analytics updates are propagated to all connected clients within acceptable timeframes and balancing the database load. For certain scenarios with concurrent updates, the framework may suffer in system behavior under load when multiple hubs generate analytics events simultaneously. Therefore, a potential avenue discovered through this was the need for actual battle-tested big data technologies. Other critical test paths that were made during this testing included verification of the responsiveness in the analytics dashboard, ensuring that UI updates could maintain a good user experience while maintaining data consistency with backend calculations. The testing also revealed the crucial aspect missing from the app in its tested state, that is, a lack of complete observability. The testing framework revealed missing gaps in latency logging, alert mechanisms and other system-level analytical tools to identify potential performance degradation, with specific focus on the integration points between the analytics service and the main application components.

5.3.2(b) Chart Libraries and Data Processing Testing

Integration testing for chart libraries and data processing focuses on validating the accuracy and performance of data visualization components that use Recharts under the hood. The test suite covers all chart types implemented in the analytics module, including HubMetricsRadar, ReviewVolume, TopRatedHubs, and CategoryAnalysis. The main comparison to be made was in comparing the current library used to more specialized libraries like D3.js. The implementation of the tests showed to verify sufficiency for a low to medium scale data pipeline, ensuring that the lower amount of analytics data was correctly processed and formatted for each chart type like PieCharts and BarCharts. The testing notably discovered a lack of other types of graphs, which is in part of validating chart rendering performance. Potentially, an API/library could be integrated to measure time-to-interactive metrics and memory usage patterns. Something of note is attention paid to testing the responsiveness and accessibility of charts across different screen sizes and browsers. The test suite was only run on Firefox and desktop. So, the verification of chart interaction handlers, tooltips, zoom controls, and click events function could not maximize coverage. Next, the integration tests covered the chart configuration system provided by shadcn, validating that theme changes and custom styling are properly applied across all visualization components. The framework could improve by carrying out means like visual regression testing to ensure consistent rendering across different browsers and devices or with specific focus on animation performance and

SVG rendering optimization under the hood potentially provided by options like D3.js and Observable.

5.3.2(c) Analytics Chatbot Tool Calling Integrations

Meanwhile, the analytics chatbot integration testing focused on validating the interaction between the natural language interface and the analytics tools system. The test suite covers the complete flow from user query processing through tool selection and execution in tools.ts. The continuously updated Vercel AI SDK library APIs were comprehensively tested to enable each analytics tool (memberStatsTool, eventStatsTool, participationTool, hubMetricsAnalysisTool, reviewAnalysisTool) to function as expected, verifying correct parameter parsing and result formatting with the new additions like provider meta_data. The testing includes validation of the tool calling middleware, ensuring proper handling of rate limits and error conditions. Particular attention is paid to testing the integration between tool results and chart rendering, verifying that analysis outputs are correctly visualized in the chat interface by ensuring outputs are made in structured format. The test suite includes verification of context management, ensuring that multi-turn conversations maintain appropriate state and tool selection accuracy. A lot of console logging was done.

5.4 Security Testing in Applications with Web and LLM Integrations

Security testing in web applications with integrated large language models (LLMs) necessitates a two-pronged approach: traditional web security measures to ensure a robust shell against general threats, and LLM-specific undertakings to address the unique security challenges posed by generative AI. Pragmadic's security testing strategy combines these aspects, embedding (no pun intended) security at multiple levels to protect against unauthorized access, mitigate potential data leaks, and reduce vulnerability to adversarial attacks as outlined by the OWASP 2024 LLM AI Security and Governance Checklist as previously cited.

5.4.1 Rate Limiting

```
pragmadic-v1 - route.ts

const ratelimit = new Ratelimit({
  redis: kv,
  limiter: Ratelimit.fixedWindow(5, "30s"),
})

export async function POST(req: NextRequest) {
  // call ratelimit with request ip
  const ip = req.headers.get("x-forwarded-for") ?? "ip"
  console.log("ip", ip)
  const { success, remaining } = await ratelimit.limit(ip)
  console.log("remaining", remaining)
  // block the request if unsuccessful
  if (!success) {
    return new Response("RateLimited!", { status: 429 })
  }
}
```

Figure 5.1: Example of Upstash Rate Limit implemented to protect chat API endpoint

The simplest basis of Pragmadic's approach from a web security are rigorous rate-limiting measures that govern API endpoint consumption on the backend, applied to both authenticated and unauthenticated endpoints. These measures are crucial in mitigating denial-of-service (DoS) attacks and at the very least prevent brute-force attempts from a singular location, ensuring that application resources avoid catastrophic abuse.

5.4.2 LLM Guardrails

Furthermore, LLM-specific guardrails are implemented, addressing risks that emerge from LLM functionalities such as prompt injections, data leakage, and the potential misuse of generated outputs.

```

pragmatic-v1 - ragMiddleware.ts

// Classify the user prompt
const { object: classification } = await generateObject({
    model: openai("gpt-4o-mini", { structuredOutputs: true }),
    output: "enum",
    enum: ["question", "statement", "other"],
    system: "classify the user message as a question, statement, or other",
    prompt: lastUserMessageContent,
})

console.log("classification", classification)

// only use RAG for questions
if (classification !== "question") {
    messages.push(recentMessage)
    return params
}

// Generate hypothetical answer
const { text: hypotheticalAnswer } = await generateText({
    model: openai("gpt-4o-mini", { structuredOutputs: true }),
    system: "Answer the users question:",
    prompt: lastUserMessageContent,
})
console.log("hypotheticalAnswer", hypotheticalAnswer)
// Embed the hypothetical answer
const { embedding: hypotheticalAnswerEmbedding } = await embed({
    model: openai.embedding("text-embedding-3-small"),
    value: hypotheticalAnswer,
})

```

Figure 5.2: Example of Multi-step RAG Implemented for Accuracy

Testing for these unique security aspects includes simulation of adversarial attacks, real-time monitoring, and regular updates to security policies that integrate with the broader cybersecurity framework used across the organization (OWASP, 2024). As previously researched in the literature review, this potential pattern of wrapping LLM calls with middleware is now a widely recognized pattern, especially seen in implementations for logic and reasoning challenges. Therefore, the security posture was analysed which revealed a lack of observability and LLM tracing implemented, mainly out of lack of established resources for such a cutting-edge evolution of natural language processing technologies. Another avenue that was not able to be explored was the usage of web search integrations that allow AI agents to browse the web similar to products like Perplexity. It is not within the scope of this project but is a potential outlet to consider in making a secure LLM application.

5.4.3 Access Control

Access control is outlined by OWASP to be central to maintaining a secure application environment in general, and particularly in systems with role-based interactions and sensitive data operations such as DE Rantau handling sensitive PII. Pragmatic's access control framework is generally rudimentary when compared to enterprise solutions. Pragmatic opts to use session-based authentication managed by Supabase Auth and role-based access control (RBAC) through custom claims to secure knowledge base management. There are multiple vulnerabilities in the security posture employed at higher level when it comes to who are assigned roles and permissions. Separate access privileges for analytics, document selection, and administrative functions were not considered in the scope of the security aspect.

A good policy to take moving forward would be to granularize as much as possible the access to document selections and analytics tools. A key component is the use of observability tools, but it is also important to allow the system to enforce permissions at a detailed level at a base level, as recommended by OWASP's checklist for LLM applications. This setup not only protects sensitive data but also prevents unauthorized users from accessing LLM capabilities. Additionally, Pragmatic's access control architecture could support audit logging, recording all user interactions with LLMs, but was not implemented. This could help monitor access patterns, identify suspicious behaviour, and facilitate quick responses with security principles of non-repudiation implemented for potential breaches.

As a quick additional note, from research, security testing for access control especially in large scale enterprise tech companies is greatly multi-layered, incorporating penetration testing and tests that simulate unauthorized access attempts to LLM functionalities, checking whether the models are prone to unsafe behavior.

In total, rate limiting, guardrails, and access control are the basis for a robust security layer for Pragmatic's unique hybrid of being a web and LLM-integrated application. Through rigorous research and adherence to OWASP's latest guidelines, these mechanisms are guided in the right direction to directly contribute to a proactive security posture that addresses both traditional and LLM-specific threats effectively.

CHAPTER 6: DISCUSSION

6.1 Achievements

6.2 Limitations

6.3 Future Enhancements

6.4 Conclusion

CHAPTER 7: REFLECTION

7.1 Strengths

7.2 Challenges Overcame

7.3 Self-improvement

CHAPTER 8: CONCLUSION

REFERENCES

1. Amazon Bedrock, n.d., What is Amazon Bedrock? Amazon Web Services Documentation. Available at: <https://docs.aws.amazon.com/bedrock/latest/userguide/what-is-bedrock.html> (Accessed 26 May 2024)
2. Amazon Web Services, n.d., Welcome to AWS Documentation. Available at: <https://docs.aws.amazon.com/> (Accessed 26 May 2024)
3. Amazon Web Services, n.d., What Is Amazon Bedrock? - Amazon Bedrock (n.d.) available from <https://docs.aws.amazon.com/bedrock/latest/userguide/what-is-bedrock.html>
4. Amplify Docs, n.d., How Amplify Works, Amplify Docs Gen 2. Available at: <https://docs.amplify.aws/react/how-amplify-works/concepts/> (Accessed 26 May 2024)
5. Asai, A., Wu, Z., Wang, Y., Sil, A. and Hajishirzi, H., 2023. Self-rag: Learning to retrieve, generate, and critique through self-reflection. *arXiv preprint arXiv:2310.11511*.
6. Aziz, M. (2023a) Since its launch in Sep 2022, Malaysia's DE Rantau programme has been making strides [Post], LinkedIn. Available at: https://www.linkedin.com/posts/mahadhiraziz_workfrommalaysia-malaysia-sayadigital-activity-7085243026251386881-MTIn/ (Accessed: 14 April 2024).
7. Aziz, M. (2023b) Insight: Transforming Malaysia into ASEAN's Digital Nomad epicentre, The Edge Malaysia. Available at: <https://theedgemalaysia.com/node/689691> (Accessed: 14 April 2024).
8. Aziz, M. (2024). DE Rantau Nomad terima 2,600 permohonan sejak dilancarkan pada 2023 [online video] Interviewed by BERNAMA TV. 10 April, Available at: <https://www.youtube.com/watch?v=tSJyFFZv6FM> [Accessed 12/04/2024]
9. Bednorz, J., 2024. Working from anywhere? Work from here! Approaches to attract digital nomads. *Annals of Tourism Research*, 105, p.103715.
10. Boluda Chova, J. and von Ehrlich-Treuenstätt, K.T., 2023. Nomadism without borders: Exploring connections in digital nomad destinations: An ethnographic multiple-case study in Malaysia & Colombia
11. Business Advisory Group on Migration, 2024. Digital Nomads and Hybrid Work: A Deep Dive into the Benefits, Drawbacks and Possible Ways to Improve Uptake of Digital Nomad Visa Programmes. Available at: <https://www.ioe-emp.org/index.php?eID=dumpFile&t=f&f=159891&token=319e52a2beee5e0e3616a5919f2d1c418db11f0b> (Accessed 14 June 2024)
12. Citizen Remote, n.d., Citizen Remote Platform. Citizen Remote Website Available at: <https://citizenremote.com/> (Accessed 26 May 2024)
13. Firebase Genkit, 2024, Firebase Genkit Overview. Firebase Documentation. Available at: <https://firebase.google.com/docs/genkit?hl=en> (Accessed 26 May 2024)

14. Firebase, n.d., Firebase Documentation Overview. Firebase Documentation. Available at: <https://firebase.google.com/docs> (Accessed 26 May 2024)
15. Frontend Mastery, 2022, *The new wave of Javascript Web Frameworks* (September 2022) *Frontend Mastery*. Available at: <https://frontendmastery.com/posts/the-new-wave-of-javascript-web-frameworks/> (Accessed: 26 May 2024).
16. Hempenius & Osmani, 2024, What's new in JavaScript Frameworks (May 2024), Chrome for Developers. Available at: <https://developer.chrome.com/blog/frameworks-may-2024> (Accessed: 26 May 2024)
17. Jeong, S., Baek, J., Cho, S., Hwang, S.J. and Park, J.C., 2024. Adaptive-rag: Learning to adapt retrieval-augmented large language models through question complexity. *arXiv preprint arXiv:2403.14403*.
18. LangChain, n.d., Overview of LangGraph. LangChain Docs. Available at: <https://langchain-ai.github.io/langgraph/#why-langgraph>
19. Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., Küttler, H., Lewis, M., Yih, W.T., Rocktäschel, T. and Riedel, S., 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in Neural Information Processing Systems*, 33, pp.9459-9474.
20. Litania, A. and Marsan, G.A., 2023. Attracting Global Talents: Bringing Digital Nomads and the Highly Skilled into ASEAN (No. PB-2023-11).
21. Lybrand, B., 2024, Merging Remix and React Router (May 2024), Remix Blog. Available at: <https://remix.run/blog/merging-remix-and-react-router> (Accessed: 26 May 2024)
22. Malaysia Digital Economy Corporation (2024). De Rantau Pass: Frequently Asked Questions (FAQ). Available at: https://mdec.my/static/pdf/derantau/DE%20Rantau%20Pass%20FAQ_SST.pdf (Accessed: 09 April 2024)
23. Manner, J., 2023, July. A Structured Literature Review Approach to Define Serverless Computing and Function as a Service. In *2023 IEEE 16th International Conference on Cloud Computing (CLOUD)* (pp. 516-522). IEEE.
24. Moore, A. A., 2024, The foundations of the Frontend Cloud. Vercel Blog. Available at: <https://vercel.com/blog/the-foundations-of-the-frontend-cloud>. (Accessed 26 May 2024)
25. Noor, J.H., 2024. The Effects of Architectural Design Decisions on Framework Adoption: A Comparative Evaluation of Meta-Frameworks in Modern Web Development.
26. Ollama, n.d., Ollama Documentation. Ollama GitHub Repository. Available at: <https://github.com/ollama/ollama> (Accessed 26 May 2024)
27. OpenAI, n.d., OpenAI API Reference, OpenAI Documentation. Available at: <https://platform.openai.com/docs/api-reference/introduction> (Accessed 26 May 2024)

28. Remix, 2024, Introduction, Technical Explanation (2024), Remix Docs. Available at: <https://remix.run/docs/en/main/discussion/introduction#introduction-technical-explanation> (Accessed: 26 May 2024)
29. Republic of Estonia, n.d., e-Residency vs digital nomad visa. Official Republic Of Estonia E-Residency Website. Available at: <https://www.e-resident.gov.ee/nomadvisa/>
30. Roger, S.P. and Bruce, R.M., 2015. Software engineering: a practitioner's approach.
31. Schmidhuber, J., 2022. Annotated history of modern ai and deep learning. *arXiv preprint arXiv:2212.11279*.
32. SolidJS Core, 2024. SolidStart 1.0: The Shape of Frameworks to Come. SolidJS Blog. Available at: <https://www.solidjs.com/blog/solid-start-the-shape-frameworks-to-come> (Accessed: 26 May 2024)
33. SolidStart, 2024. SolidStart Fine-grained reactivity goes fullstack. SolidStart Home Page. Available at: <https://start.solidjs.com/> (Accessed: 26 May 2024)
34. Supabase, n.d., Supabase Architecture. Supabase Documentation. Available at: <https://supabase.com/docs/guides/getting-started/architecture>
35. Yan, S.Q., Gu, J.C., Zhu, Y. and Ling, Z.H., 2024. Corrective Retrieval Augmented Generation. *arXiv preprint arXiv:2401.15884*.
36. OWASP (2024) *LLM AI Cybersecurity & Governance Checklist*, OWASP. Available at: https://owasp.org/www-project-top-10-for-large-language-model-applications/llm-top-10-governance-doc/LLM_AI_Security_and_Governance_Checklist-v1.pdf (Accessed: 06 November 2024).
- 37.

APPENDICES

Appendix A: DE Rantau Statistics JUN2024



Figure A.1: Statistics of DE Rantau Program cumulative to June 2024, sourced from a community post in [Malaysia's DE Rantau Digital Nomad Member Community](#)



Figure A.2: Picture of Post captured 20th June 2024

Appendix B: Questionnaire

The screenshot shows the first page of a questionnaire titled "Digital Nomad (DE Rantau Nomad Pass Program) Requirements Analysis Questionnaire". It is a Microsoft Word document with a blue header bar. The title is centered at the top. Below the title, there is a paragraph of text about the purpose of the questionnaire and the researcher's background. A numbered list follows, detailing the aims of the questionnaire. At the bottom, there is a note about the time required, a link to a pre-filled version, and a reference to a FAQ document.

Section 1 of 3

Digital Nomad (DE Rantau Nomad Pass Program) Requirements Analysis Questionnaire

Hello, I'm Matthew, an undergraduate student from Penang, Malaysia. I'm conducting this questionnaire to validate the DE Rantau Digital Nomad community's pain points and also collect insights for the requirements analysis phase as part of my final year project.

My undergraduate studies is categorized as a Bachelor of Science (Hons) in Computer Science in collaboration with Coventry University under the School of Computing of INTI International College Perang.

1. The questionnaire aims to validate the problem and focus of my project through gathering real-world input from the DE Rantau digital nomad community of Malaysia.
2. Insights from the active community will be gathered to serve as the foundation of an onboarding support platform incorporating generative AI features tailored to digital nomad workflows.
3. This questionnaire is carried out solely for academic purposes and data collected will be treated confidentially up to ethics standards expected by the overseeing academic institutions.
4. This questionnaire aims to target in terms of respondents:

- digital nomads interested in applying for the DE Rantau Nomad Pass, a specialized Professional Visit Visa recently tailor-made to allow qualified foreign digital nomads to travel and work in Malaysia [1].
- digital nomads who are currently in the submission / approval process phase of the DE Rantau program.
- digital nomads who have been approved for the DE Rantau Nomad Pass but are yet to enter Malaysia
- digital nomads that are holders of the DE Rantau Nomad Pass and currently settled in Malaysia as a place of operations for the foreseeable moment.
- digital nomads who have successfully completed their stay in Malaysia with the DE Rantau Nomad Pass.

This questionnaire should not take more than 10 minutes of your valuable time. Thank you very much.

You may access a pre-filled matrices version of the form using the link:
<https://bit.ly/derantauquestionnaireprefilled>

[1] - Malaysia Digital Economy Corporation (2024). De Rantau Nomad Pass: Frequently Asked Questions (FAQ) available at https://index.mysticid/pdf/derantau/DE%20Rantau%20Pass%20FAQ_SST.pdf

Figure B.1: Cover of Questionnaire

The screenshot shows the "Brief Project Title Explanation and Motivation" section of the questionnaire. It includes logos for INTI International University & Colleges and Coventry University. The section starts with a brief project title explanation and motivation, followed by a bulleted list of project objectives. At the bottom, there is a link to a pre-filled version of the form and a list of statements for the respondent to choose from.

Overseeing Academic Institutions

INTI International University & Colleges™ in collaboration with **Coventry University**

INTI International College Penang

Brief Project Title Explanation and Motivations

- PRONOMADS is a web application attempting to address challenges of exhaustive information in the digital nomad visa landscape and provide solutions incorporated with generative AI for digital nomads who want to engage in the DE Rantau program and explore Malaysia for career and lifestyle opportunities.
- My final year project is the development of a full-fledged platform that plans to utilize large language models capabilities and retrieval-augmented generation (RAG) to aid in aggregating and curating information for the querying and planning purposes of digital nomads.
- The platform's target audience include those who are interested in applying for the Malaysia Digital Economy Corporation's DE Rantau Nomad Pass visa program and current active DE Rantau Nomad Pass holders.
- Another objective aside from personalizing and expanding the experience of digital nomad support in Malaysia is to facilitate a suite of services for DBS to onboard themselves and get a soft landing through locally-based networking opportunities and events at locations such as coworking spaces that are centralized under the DE Rantau Nomad Pass program.
- Finally, one of the goal is to be able to lead crowdsourcing these entities to provide a community-driven platform to increase visibility of curated information and provide real-time feedback for queries digital nomads may have regarding the spaces' services and establishment to ultimately reap the economic spillover benefits.

You may access a pre-filled matrices version of the form using the link:
<https://bit.ly/derantauquestionnaireprefilled>

Firstly, which of these statements best describes your status currently as a digital nomad? *

I'm a digital nomad interested in applying for the DE Rantau Nomad Pass program or in the submiss...

I'm a digital nomad whose DE Rantau Nomad Pass application has been approved and currently pl...

I'm a digital nomad who has already entered Malaysia with the DE Rantau Nomad Pass / complete ...

Figure B.2: Explanation of Motivation

Full questionnaire available at:

<https://bit.ly/derantauquestionnaireprefilled>

<https://forms.gle/FZDHW1MGFzxzA92X9>

Appendix C: GitHub Repository Link

<https://github.com/matthewloh/pragmatic-v1>

Appendix D: Login and Registration Module Black Box Testing Results

Appendix I: Sign Up Form Validation Successful Email Link and Login

Sign up to Continue

Use your email or another service to continue

First Name

Last Name

Email

Invalid email address

Password
 

One number

Confirm Password
 

Create account

 Continue with Google

 Continue with GitHub

Email

Send Magic Link

Already have an account? [Sign in](#)

Sign up to Continue

Use your email or another service to continue

First Name

Last Name

Email

Password
 

Confirm Password
 

The passwords did not match

[Create account](#)

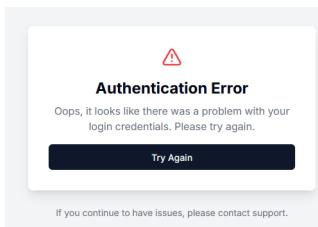
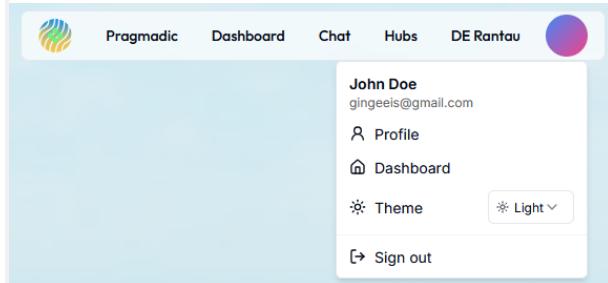
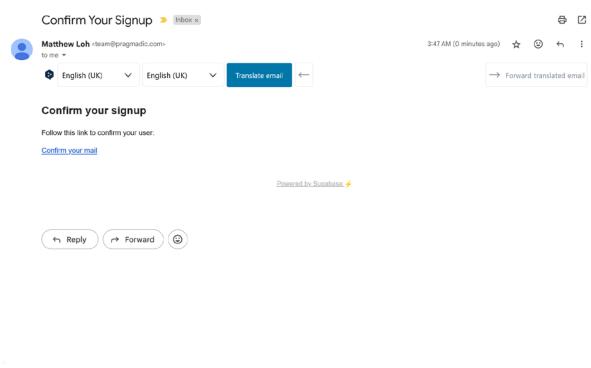
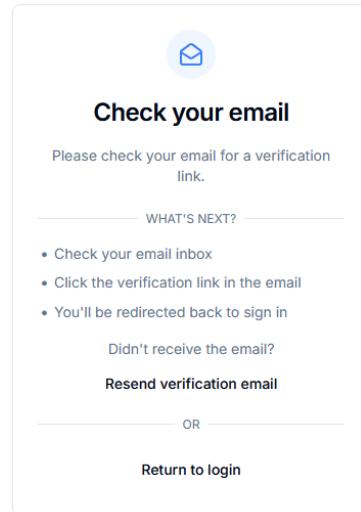
 [Continue with Google](#)

 [Continue with GitHub](#)

Email

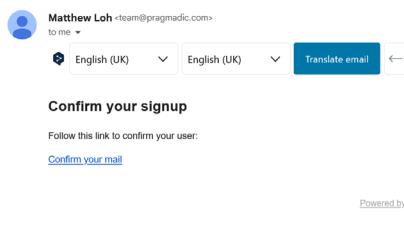
[Send Magic Link](#)

Already have an account? [Sign in](#)



Error is given upon second use of Login Verification Email

Appendix II: Duplicate Email



Returns Magic Link if exist

Appendix III: Google OAuth Login

The screenshot shows a browser window with the Google 'Sign in with Google' logo at the top. Below it, the text 'Choose an account' is displayed, followed by 'to continue to iwoygsdyztqrbbbyg.supabase.co'. To the right, there is a vertical list of user profile icons. A sidebar on the left contains a table with the following data:

	UID	Providers	Display name	
g	6941ae16-6055-473c-a3ff-d5a5d883e1fa	Email	-	(green)
g	15f7d703-9fa0-4cbe-94c8-f32aa85acb9f	Email	Gingee Loh	(green)
g	81574ab8-e68a-430f-9fe6-8314840bfc19	Email	Matthew Loh	(brown)
g	363d45e4-21e6-4a62-a2a8-87ac06b7eb95	Email	-	(brown)
M	d18de68d-7a26-42c5-bc40-0442e39290fb	Google	Matthew Loh Yet Marn	(blue)
a	2d7c06cc-a575-45ba-b8ae-b0c762e35bbe	Google	anybody want diluc lol	(blue)
g	fc076d78-ce49-48e6-9f3a-5db4a5244870	Google	Matthew Loh	(green)

At the bottom of the sidebar, it says 'English (United Kingdom) ▾'.

Appendix IV: Login with Credentials Errors and Rate Limiting

The image displays two side-by-side screenshots of a login interface.

Left Screenshot: A 'Sign In to Continue' page. It includes fields for 'Email' (gingeels@gmail.com) and 'Password' (redacted). A red error message 'Invalid credentials entered.' is displayed above the password field. Below the fields are buttons for 'Sign In', 'Continue with Google', and 'Continue with GitHub'. At the bottom, there is another 'Email' field (you@example.com) and a 'Send Magic Link' button.

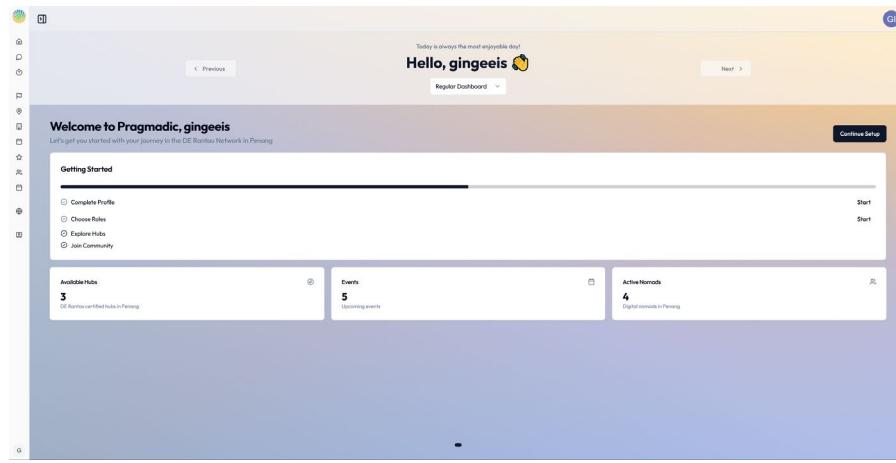
Right Screenshot: A 'Sign In to Continue' page. It shows an 'Email' field (you@example.com) and a 'Sign In' button. Above the email field, a red message says 'Invalid credentials entered. 1 attempts remaining.' To the right of the email field is a 'Forgot password?' link. Below the sign-in area are buttons for 'Continue with Google' and 'Continue with GitHub'. At the bottom, there is a 'Send Magic Link' button and a note 'Don't have an account? [Sign up.](#)'

Appendix V: Magic Link Sign in and Sign Up

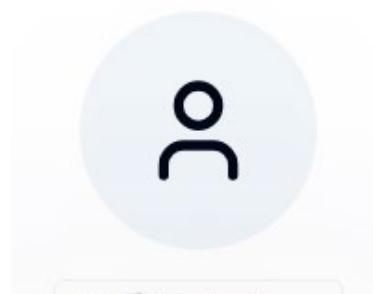
The composite screenshot illustrates the Pragmatic sign-up and sign-in process:

- Sign-up Step:** A user attempts to sign up with the email "gingeeis@gmail.com", which is flagged as an invalid email address. A button labeled "Send Magic Link" is present.
- Sign-in Step:** An email from "Matthew Loh <team@pragmadic.com>" is shown, containing a "Confirm your signup" link. The email includes standard message controls like reply, forward, and delete.
- Profile View:** The user logs in and views their profile page. The sidebar shows "gingeeis" and "gingeeis@gmail.com". Options include "Profile", "Dashboard", "Theme" (set to "Light"), and "Sign out".
- Email Confirmation:** The user receives an email titled "Your Magic Link" from "Matthew Loh" with a "Log In" button. The email includes "Powered by Supabase" branding.

Appendix VI: Initial Login and Basic Profile Setup



The screenshot shows the "Complete Your Profile (Optional)" section. It includes fields for "Display Name" (gingeelis), "Bio" (Tell us about yourself), "Occupation" (Occupation), "Location" (Location), "Website" (Website), and "Contact Number" (Contact Number). A "Continue" button is at the bottom right.



! Failed to update user profile, please check you have uploaded an image for your profile

Welcome to Pragmatic! 🎉

Let's get started by setting up your profile.



Display Name *

Complete Your Profile (Optional)

Tell us more about yourself

Bio

Tell us about yourself

Occupation

Location

Website

Contact Number

Social Links

Profile picture updated!



Profile Details

📍 Batu Ferringhi, Penang
🌐 https://www.yotubakoawai.com/
📞 +6012-2233445
👤 @ilam_markets

Joined 08/11/2024

Activity Overview

Connections 0	Events Attended 0	Reviews 0	Hub Visits 0
---------------	-------------------	-----------	--------------

Successful base profile picture upload and creation

Appendix VII: Digital Nomad Profile Creation

Pragmatic Features

- Dashboard
- RAG Chatbot
- Getting Started

DE Rantau Platform

- DE Rantau Regions
- DE Rantau States
- Hubs
- Hub Events
- Hub Reviews
- Communities
- Community Events

Onboarding Support

- Penang Digital Nomad Guide

Profile Management

- User Profile
- Nomad Profile

Full-stack developer with a background in AI. Penang's tech ecosystem has drawn me here, and I'm looking forward to collaborating with local tech enthusiasts.

Tokyo, Japan

@yotsubakoiwai

Skills

JavaScript Python Machine Learning

Interests

Coding AI Entrepreneurship

1
2
3

Basic Info Choose Role Complete Profile

[Back to Basic Info](#)

Choose Your Role

We found your existing Digital Nomad profile!

Select how you'll primarily use Pragmatic to get started with your personalized experience

Digital Nomad

Join vibrant hubs, connect with fellow nomads, and discover opportunities in the digital nomad community.

- Join exclusive digital nomad hubs
- Connect with local communities
- Access nomad-friendly resources

Click to edit your Digital Nomad profile

Hub Owner

Create and manage your own hub, build your community, and connect with digital nomads.

- Create and manage your hub
- Engage with nomad community
- Track hub analytics and growth

You can always change your role or add additional roles later

Successful nomad profile creation + given nomad role.

Appendix VIII: Hub Owner Profile Creation

Update Your Hub Owner Profile

Review and update your business information

Company Name: Freelance Hub Penang

Registration Number: SSM334455667

Business Description: Freelance Hub Penang provides a welcoming coworking environment with all the amenities needed by digital nomads. Our hub features quiet zones, collaborative spaces, and a strong community spirit for freelancers in Penang.

Business Contact: +6016-334-5567

Business Email: support@freelancehubpg.com

Business Location: 47 Jalan Sultan, 10000 George Town, Penang, Malaysia

Residing Location: 22 Jalan Burma, 10350 George Town, Penang, Malaysia

Social Media: @freelancehubpg on LinkedIn, Facebook

Website: https://www.freelancehubpg.com

Update Profile

Analytics

Profile Management

User Profile

Nomad Profile

Hub Owner Profile

< Back to Basic Info

Choose Your Role

We found your existing Digital Nomad and Hub Owner profiles!

Select how you'll primarily use Pragmatic to get started with your personalized experience

Digital Nomad

Join vibrant hubs, connect with fellow nomads, and discover opportunities in the digital nomad community.

- Join exclusive digital nomad hubs
- Connect with local communities
- Access nomad-friendly resources

Hub Owner

Create and manage your own hub, build your community, and connect with digital nomads.

- Create and manage your hub
- Engage with nomad community
- Track hub analytics and growth

You can always change your role or add additional roles later

Successful rendering of hub owner details and analytics is loaded in the sidebar. User roles acquired properly

Appendix IX: RAG Chatbot Black Box Testing

The screenshots illustrate the AI Assistant's responses to different queries:

- Query 1:** "Hello" "What is DE Rantau?"
Response: "I don't have enough information to answer that question. I'm unfamiliar with "DE Rantau." Please provide more context."
- Query 2:** "Hello" "What is DE Rantau?"
Response: "I don't have enough information to answer that question. I'm unfamiliar with "DE Rantau." Please provide more context."
- Query 3:** "What is DE Rantau??"
Response: "The DE Rantau Nomad Pass is a Professional Visit Pass designed for foreign digital nomads who want to work remotely from Malaysia. It's available for both tech and non-tech professionals who meet specific income requirements (USD 24,000/year for tech talent and USD 60,000/year for non-tech talent). The pass is valid for 3 to 12 months, renewable for an additional 12 months. (DE Rantau Pass FAQ Version 4.0)"
- Query 4:** "Google - Gemini" "20240808_DE Rantau Pass FAQ_ChinaBangladesh_MAF.pdf"
Response: "Enter a message..." (This is a search interface, not a direct AI response.)
- Query 5:** "What model are you running?"
Response: "I'm currently running on the Gemini family of models."
- Query 6:** "What model are you running?"
Response: "I am based on OpenAI's GPT-4 model."
- Query 7:** "OpenAI - GPT-4o"
Response: "Enter a message..." (This is a search interface, not a direct AI response.)