# Dishant Parikh - C0934360

# Sana Khan - C0937931

# Zoheb Siddiqui - C0933790

# Groot Growth: A Human-Centric AI Financial Companion

## Abstract

Groot Growth reimagines wealth management by wrapping a fine-tuned causal language model in a simple, privacy-conscious chat interface. Unlike sterile, data-heavy fintech tools, Groot Growth delivers clear, personalized financial guidance in natural language—powered by Streamlit on the frontend, a Transformers-based text-generation pipeline on the backend, and seamless Docker-based deployment via Hugging Face Spaces. This report details the actual system architecture, core capabilities, implementation choices, and roadmap for future enhancements.

## 1. Introduction

Traditional financial planning apps present users with charts and spreadsheets that often feel impersonal. At Groot Growth, we believe lasting prosperity grows like a forest—organically, patiently, and in tune with human nature. Our mission is to “demystify wealth building through intuitive technology that speaks human”—even if that means a single phrase of guidance from your AI companion, Groot. By reframing every decision as part of a living ecosystem, we turn transactions into narratives, deadlines into seasons, and milestones into growth rings.

## 2. Business Context & Purpose

Individual investors today face three core challenges:  
1. Emotional Disconnect – data overload leads to decision paralysis.  
2. Generic Advice – one-size-fits-all recommendations ignore personal goals.  
3. Privacy Concerns – mistrust of platforms that harvest sensitive data.

Groot Growth addresses these by:  
- Emotional Engagement: a conversational UI that feels like talking to a trusted guide.  
- Hyper-personalization: context-aware prompts drawn from your own financial profile.  
- Data Minimalism: no external analytics tracking—user inputs are processed in-memory and not stored server-side.

## 3. System Architecture

Frontend – Streamlit UI:  
- Layout & Configuration: st.set\_page\_config and st.title in app.py  
- Input Widgets: st.text\_area and st.button  
- Output Display: st.markdown and st.write for responses

Backend – Python & Transformers:  
1. Environment Prep: cleanup of stale lock files and configuration of HF cache directories.  
2. Model Initialization: cached init\_generator() loading AutoTokenizer and AutoModelForCausalLM.  
3. Session Management: storing the generator in st.session\_state after loading.  
4. Inference Flow: prompt assembly, pipeline call with max\_new\_tokens=100, do\_sample=False, temperature=0.0, and post-processing.

Infrastructure – Docker & CI/CD:  
Dockerfile based on python:3.12-slim configures /cache and .streamlit directories, installs dependencies, and launches the app on port 8501. Deployment uses Hugging Face Spaces CI/CD: git push triggers Docker build and instant container deployment at the public URL.

A close-up of a computer screen

AI-generated content may be incorrect.

## 4. Core Features

1. Conversational Guidance: natural-language Q&A powered by a causal language model.  
2. Custom Training Corpus: domain knowledge from training.txt guides responses.  
3. Deterministic Generation: greedy decoding for consistent advice.  
4. In-Session Caching: model and tokenizer cached to reduce latency after first load.

## 5. Technical Implementation

Modeling & Inference:  
Uses Hugging Face Transformers pipeline with AutoTokenizer and AutoModelForCausalLM from deepseek-ai/DeepSeek-Coder-1.3B-base. Prompt engineering includes a system preface, the training corpus, and user input.

Dependencies & Environment:  
requirements.txt pins streamlit, transformers, and torch. Docker ensures reproducible environments.

Containerization & Deployment:  
Docker builds a minimal container running Streamlit. Hugging Face Spaces CI/CD automates build and deploy on git push.

Performance:  
Cold-start latency: approx. 30–60 seconds on first load. Generation time: under 0.5 seconds per 100-token output.

## 6. User Journey & Engagement

1. Onboarding: sample questions and brief description.  
2. Interactive Q&A: type a question and receive tailored advice.  
3. Session Continuity: follow-up questions use cached model.  
4. Iterative Learning: update training.txt to refine responses.

## 7. Conclusion & Future Work

Groot Growth delivers a human-centric financial advisor in under 200 lines of code. Future enhancements include fine-tuning on opt-in user logs, richer prompt templates, and latency optimizations via quantization or GPU support.