INTERNSHIP REPORT

SUBMITTED IN THE
PARTIAL FULFILLMENT OF THE
REQUIREMENT FOR THE AWARD OF THE
DEGREE OF BACHELOR IN TECHNOLOGY
(COMPUTER SCIENCE AND
ENGINEERING)

FACULTY OF ENGINEERING & TECHNOLOGY,
AGRA COLLEGE,
AGRA

SUBMITTED TO

DR. ANURAG SHARMA
HEAD OF THE DEPARTMENT
(COMPUTER SCIENCE & ENGG.)

SUBMITTED BY

SHIKHER JAIN
2200020100075
4TH YEAR

Table Of Contents

7. System Architecture

- Architecture Pattern
- Data Flow
- Pipeline Architecture
- NLP Engine Architecture
- LLM Fine-Tuning Architecture

8. Features & Functionality

- FAQ Extraction System
- Context Analyzer
- System Prompt Generator
- Fine-Tuning Automator
- Dataset Builder
- Chatbot Engine
- Evaluation Framework

9. Database / Data Design

- Dataset Format
- Data Models
- Indexing & Optimization

10. API Architecture

11. Development Process

- Methodology
- Version Control
- Development Setup
- Testing Approach

13. Challenges & Solutions

14. Key Learnings & Outcomes

15. Conclusion

Acknowledgement

I would like to express my profound gratitude to **Dr. Anurag Sharma**, Head of the Department of Computer Science & Engineering, Faculty of Engineering & Technology, Agra College, Agra, for granting me the opportunity to pursue this internship. His academic vision, constant encouragement, and guidance have played a vital role in shaping my technical and professional development.

I extend my heartfelt appreciation to my mentors at **Novas Arc Consulting Pvt. Ltd.**:

- **Ms. Ashiya Syed (Manager)** — for her exceptional mentorship, strategic guidance, and deep insights into AI pipeline architecture, NLP system engineering, and data-driven solution design. Her clarity of thought, problem-solving approach, and structured feedback significantly elevated the quality of my work.
- **Ms. Roja Velpuri (Data Science Mentor)** — for her expertise in dataset engineering, prompt optimization, LLM fine-tuning, and evaluation methodologies. Her continuous support and practical insights greatly enhanced my understanding of applied NLP and large-scale AI workflows.

I also thank the entire **Novas Arc team** for fostering a collaborative, innovative, and research-centric environment. Working alongside experienced professionals allowed me to refine my engineering mindset, strengthen my analytical thinking, and gain hands-on experience with real-world AI development processes. This internship has been instrumental in expanding my technical capabilities and shaping my aspirations in the field of Artificial Intelligence.

Abstract

This internship report documents the complete technical, architectural, and developmental work carried out at **Novas Arc Consulting Pvt. Ltd.** as part of my **AI/Data Science Internship** from **11 August 2025 to 15 November 2025**.

The focus of the internship was the development of an **end-to-end AI automation system** consisting of:

- Automated **FAQ Extraction Pipeline** capable of scraping and structuring FAQs from any website.
- **Context Analyzer** to classify tone, domain, persona, intent, and target audience using NLP.
- **Dynamic System Prompt Generator** for precise LLM-context instructions.
- Automated workflows for **fine-tuning OpenAI GPT-3.5 Turbo** on curated datasets.
- A fully functional **domain-specific AI Chatbot** trained on the processed datasets.
- Evaluation workflows for response quality, consistency, and domain alignment.

This report covers system design, data pipelines, NLP methodologies, LLM training processes, reasoning frameworks, and performance optimization strategies. The internship experience strengthened my practical understanding of ML, LLMs, prompt engineering, automation, and scalable AI system development.

1. Introduction

Modern artificial intelligence systems demand robust data pipelines, scalable architectures, and intelligent automation strategies to consistently achieve high accuracy and practical real-world applicability. As organizations increasingly rely on AI-driven insights, the ability to efficiently extract, process, and utilize domain-specific knowledge has become essential. During my internship at **Novas Arc Consulting Pvt. Ltd.**, I was involved in developing a comprehensive end-to-end AI system designed to meet these requirements. The system integrated multiple components, including automated data extraction from websites, linguistic and semantic analysis using NLP techniques, optimized prompt-generation logic, fine-tuning of large language models, and the deployment of intelligent chatbots tailored to domain-specific tasks.

This project not only exposed me to advanced methodologies in NLP and LLM engineering but also helped me understand the real-world challenges associated with building scalable AI systems. By working closely with the team, I gained hands-on experience in designing modular architectures, implementing iterative pipelines, and applying research-driven approaches to solve practical problems.

Purpose

The primary objective of this report is to document the complete architecture and engineering workflows developed throughout the internship. Specifically, the report aims to:

- Present a detailed breakdown of the AI system architecture, describing how each component interacts within the overall workflow.
- Explain the NLP, text-processing, and LLM-driven methodologies applied at different stages.
- Highlight the challenges faced during development and the technical decisions made to overcome them.
- Demonstrate the work I contributed to—ranging from data pipeline automation to model improvement strategies and deployment processes.
- Summarize the technical and professional learnings gained during the internship, including insights into industry practices and collaborative development.

Scope

This report includes:

- System design and component-level architecture
- Dataset extraction, preprocessing, and NLP-driven classification
- LLM fine-tuning using OpenAI APIs
- Prompt engineering logic and rules framework
- Evaluation workflows and performance metrics
- Automation and iterative model improvement strategies

Methodology

The project followed:

- Research-driven development for NLP logic
- Incremental architecture design
- Pipeline automation methodology for repeatability
- Dataset-driven model fine-tuning
- Iterative testing + evaluation cycles

Internship Context

Company: Novas Arc Consulting Pvt. Ltd.

Role: AI / Data Science Intern

Domain: NLP, LLM Fine-Tuning, Chatbot Engineering

Duration: 11 August 2025 – 15 November 2025

Mentors: Ms. Ashiya Syed, Ms. Roja Velpur

2. COMPANY PROFILE

NovasArc Overview

NovasArc Consulting is an IT consulting and software development company specializing in delivering enterprise-grade solutions and custom software products. The company operates with a modern engineering culture that prioritizes collaborative team environments, continuous learning, skill development, and adherence to industry best practices.

Service Offerings:

- Custom software development tailored to client requirements
- Web and mobile application development services
- Cloud-based solution architecture and implementation
- Enterprise consulting and digital transformation services
- Agile development practices and project management

Company Culture: The engineering culture at NovasArc emphasizes collaborative team environments where developers work together on complex technical challenges. The organization prioritizes continuous learning through knowledge sharing sessions, code reviews, and exposure to diverse technologies. Agile and iterative development practices are core to project execution, enabling rapid iteration and quality improvements. Code quality is maintained through strict review processes, automated testing, and adherence to design patterns. Mentorship and professional growth are active initiatives, with senior developers providing guidance to junior team members.

Internship Environment

The internship was conducted within a professional team-based environment where I worked alongside experienced developers on production-grade projects. This setup provided invaluable exposure to real-world project experience with production-grade code standards, enterprise-level architecture patterns and design decisions, peer code review practices and collaborative feedback, and industry-standard development workflows and best practices.

The collaborative environment fostered not only technical skill development but also professional growth through exposure to professional communication, project management practices, debugging methodologies, and problem-solving approaches used in enterprise settings.

3. INTERNSHIP OVERVIEW

My internship at **Novas Arc Consulting Pvt. Ltd.** (11 August 2025 – 15 November 2025) provided hands-on experience in **Artificial Intelligence, Data Science, and NLP**, with active involvement in real-world AI workflow automation projects. I worked with senior AI engineers and mentors to build production-ready systems, gaining exposure to enterprise practices, architecture design, and iterative model improvement.

Internship Objectives

1. Build Real-World AI Systems
Worked on end-to-end workflows including FAQ extraction, NLP context analysis, dataset generation, LLM fine-tuning, chatbot building, and evaluation frameworks.

2. Strengthen NLP & Data Science Skills
Gained practical experience in data preprocessing, prompt engineering, NLP pipelines, rule-based/model-based classification, and API-driven automation.

3. Develop LLM Training Expertise
Learned to create fine-tuning datasets, structure instructions/responses, improve model accuracy, and evaluate outputs using defined metrics.

4. Understand AI Deployment Workflows
Worked with model hosting, versioning, model updates, dataset management, and pipeline organization.

5. Gain Professional & Collaborative Skills
Enhanced research-driven problem solving, technical communication, and team collaboration using code versioning tools.

Overall, the internship blended research, development, and engineering, offering a complete understanding of the AI system life cycl

4. PROJECT DESCRIPTION

Project Overview

This project builds an end-to-end AI automation pipeline that can:

- Extract FAQs and knowledge from any website URL
- Clean and structure the extracted content
- Analyze linguistic context (tone, domain, persona, intent, audience)
- Auto-generate optimized system prompts
- Create structured training datasets
- Fine-tune OpenAI GPT-3.5 Turbo for domain-specific behavior
- Deploy a context-aware chatbot
- Evaluate chatbot performance through automated tests

The system enables organizations to **extract knowledge, build datasets, train models, and deploy chatbots automatically**—with no manual effort.

Problem Statement

Companies typically struggle with:

1. **Manual Knowledge Extraction** – Copying or formatting website content is slow and error-prone.
2. **Weak Context Awareness** – Chatbots lack correct tone, domain alignment, persona, and intent classification.
3. **Generic LLM Behavior** – Responses are inconsistent and not tailored to organizational needs.
4. **No Fine-Tuning Pipeline** – Most lack automation to create datasets and fine-tune models.
5. **No Evaluation Framework** – Hard to measure chatbot accuracy or consistency.

This project solves these issues with a unified, automated pipeline.

Key System Modules

1. FAQ Extraction Engine

- Takes any URL
- Scrapes FAQs
- Auto-detects Q/A patterns
- Produces clean structured JSON

2. NLP Context Analyzer

Analyzes:

- Domain
- Tone
- Persona
- Intent
- Target audience

Ensures accurate context alignment.

3. System Prompt Generator

Creates domain-optimized prompts based on NLP results:

- Tone-specific
- Instruction-focused
- Style-aligned

4. Dataset Builder

Converts extracted data into training-ready **JSONL**:

- Instruction → Input → Response
- Balanced for consistent fine-tuning

5. Fine-Tuning Automation

Uses OpenAI APIs to:

- Upload datasets
- Configure jobs
- Train models
- Track logs and manage versions

6. Chatbot Engine

A domain-specialized chatbot with:

- Improved accuracy
- Context-aware behavior
- Responses aligned to extracted knowledge

7. Evaluation Framework

Automated testing for:

- Accuracy
- Consistency
- Error detection
- Instruction adherence

5. Technology Stack

A. AI & NLP Technologies

- **Python** – Core language for automation, NLP logic, pipelines, and evaluation.
- **BeautifulSoup + Requests** – Extracted FAQ content through HTML parsing and pattern-based Q/A detection.
- **spaCy** – Tokenization, NER, POS tagging, dependency parsing; powered tone, persona, and domain classifiers.
- **NLTK** – Text cleaning: stopwords, lemmatization, segmentation, normalization.
- **Hugging Face Transformers** – Embeddings, semantic similarity, classification, and context scoring.
- **OpenAI GPT-3.5 Fine-Tuning** – Built domain-specific models, improved structure, behavior, and accuracy.

B. Backend & Data Processing

- **Python Automation Scripts** – End-to-end pipeline: extraction → preprocessing → dataset creation → evaluation.
- **JSON / JSONL / CSV** – Dataset storage, fine-tuning formats, and logs.
- **Regex** – Detection of questions, answers, patterns, and text cleanup rules.

C. Evaluation & Testing

- **Custom Evaluation Framework** – Measured accuracy, relevance, consistency, tone, and instruction-following.
- **Prompt-Based Testing** – Blind tests, multi-turn conversations, and edge-case scenarios.
- **Error Categorization** – Identified hallucinations, tone mismatches, context drift, and generalization issues.
- **Model Comparators** – Compared base vs. fine-tuned models and dataset variations.

Overall Purpose

Enabled automated AI workflows, high-accuracy context understanding, clean datasets, reliable evaluations, and scalable chatbot deployment.

6. System Architecture

The system follows a **modular AI pipeline architecture**, ensuring scalability, automation, and clean separation between extraction, NLP processing, dataset preparation, fine-tuning, and deployment. Each module is independent, reusable, and connected through structured JSON interfaces.

A. Architecture Pattern

The project uses a **Modular Pipeline Architecture** consisting of:

1. **Extraction Layer** – Scrapes and collects FAQ content
2. **Processing Layer** – Cleans text and applies NLP
3. **Context Analysis Layer** – Determines tone, domain, persona, and intent
4. **Dataset Layer** – Builds structured training and evaluation datasets
5. **Fine-Tuning Layer** – Trains customized LLM models
6. **Deployment Layer** – Serves the finalized chatbot model
7. **Evaluation Layer** – Performs accuracy and quality checks

This pattern supports automation, maintainability, and versioned improvements.

B. Data Flow Architecture

The data flow proceeds through clearly defined stages:

1. User enters a URL
2. FAQ extractor scrapes and structures Q/A pairs
3. Preprocessing removes noise and normalizes text
4. NLP engine analyzes domain, tone, persona, and intent
5. System prompt generator injects contextual rules
6. Dataset generator maps content into JSONL training format
7. Fine-tuned model is created using OpenAI APIs
8. Model is deployed and tested via chatbot interface

C. Pipeline Architecture

The system consists of five core pipelines:

- **FAQ Extraction Pipeline:** Scrapes and structures FAQs from websites
- **NLP Processing Pipeline:** Performs POS tagging, NER, keyword extraction
- **Context Analyzer Pipeline:** Classifies tone, domain, persona, and intent
- **Dataset Generation Pipeline:** Prepares fine-tuning and evaluation datasets
- **Fine-Tuning Pipeline:** Automates model training, versioning, and logging

D. NLP Engine Architecture

The NLP engine includes:

1. **Preprocessing Layer** – cleanup, tokenization
2. **Linguistic Layer** – POS tagging, dependency parsing, NER
3. **Semantic Layer** – embeddings, similarity scoring
4. **Classification Layer** – rule-based + statistical classifiers

E. LLM Fine-Tuning Architecture

Fine-tuning operates in four phases:

- **Dataset Preparation:** Convert cleaned data to OpenAI JSONL
- **Training:** Create model versions (v1, v2…)
- **Evaluation:** Check accuracy, tone, consistency
- **Deployment:** Expose model via API with domain-aware system prompts

F. Summary

The architecture ensures:

- High modularity
- Automated workflows
- Accurate NLP-driven classification
- Clean dataset creation
- Reliable domain-specific LLM behavior
- Scalable deployment

7. Features & Functionality

The entire system developed during the internship consists of multiple intelligent components that work together to form a fully automated AI pipeline. Each module performs a specific function while contributing to the end-to-end workflow of extracting knowledge, generating datasets, fine-tuning LLMs, and deploying domain-aware chatbots.

A. FAQ Extraction System

This module automatically extracts FAQs from any website provided by the user.

Key Features

- **URL-based scraping** using BeautifulSoup
- **Automatic detection** of FAQ patterns (e.g., Q-tags, headings, keywords)
- **Noise removal** such as advertisements, scripts, or repetitive text
- **Q/A structuring** into clean dictionary or JSON format
- **Fallback rules** for websites without traditional FAQ structures

Outcome

Creates a **high-quality knowledge base** for downstream NLP pipelines.

B. Context Analyzer

Analyzes linguistic and semantic attributes from extracted content.

Capabilities

- **Domain classification** (education, travel, finance, health, etc.)
- **Tone detection** (formal, friendly, instructional, marketing-driven)
- **Persona mapping** (brand voice, user type)
- **Intent identification** (informational, transactional, problem-solving)
- **Audience detection** (student, customer, professional, general public)

Purpose

Helps generate prompts and fine-tuned datasets that match the **style and expectations** of the target domain.

C. System Prompt Generator

A dynamic prompt engineering engine that produces context-rich instructions for LLMs.

Functionality

- Builds **rule-based prompts** using attributes from the Context Analyzer
- Adds **tone, style, domain, persona, and format instructions**
- Embeds **constraints**, such as accuracy rules or response limits

- Ensures the chatbot behaves consistently with the domain

Example Output

A domain-aware instruction like:
"Behave as an academic advisor. Use a formal and informative tone. Provide precise answers strictly from the given FAQ dataset."

D. Dataset Builder

Prepares structured datasets suitable for fine-tuning large language models.

Key Features

- Converts extracted FAQs into **instruction–response pairs**
- Adds **context tags** generated from NLP analysis
- Supports JSON, JSONL, and CSV formats
- Handles **dataset balancing**, deduplication, and normalization
- Creates **evaluation datasets** for model testing

Outcome

A clean dataset ready for fine-tuning and evaluation workflows.

E. Fine-Tuning Automator

Automates the model training workflow using OpenAI APIs.

Functions

- Prepares JSONL dataset and uploads it to OpenAI
- Initiates fine-tuning runs automatically
- Tracks job status and training metrics
- Stores model versions such as v1, v2, v3…
- Supports iterative testing and retraining

Purpose

Creates a **domain-specialized LLM** that understands the extracted knowledge deeply.

F. Chatbot Engine

The chatbot is built on top of the **fine-tuned LLM**, enabling accurate domain-specific responses.

Capabilities

- Multi-turn conversation support
- Context retention
- Domain-specific reasoning

- Clean, structured answer formatting
- Controlled tone based on prompt generator output

User Applications

Can be integrated into:

- Websites
- Internal tools
- Customer support systems
- Knowledge-base interfaces

G. Evaluation Framework

Ensures that the model is reliable, accurate, and aligned with expected outputs.

Functionality

- Compares **base model vs fine-tuned model**
- Tests for **accuracy**, **consistency**, and **context alignment**
- Identifies:
    - Hallucinations
    - Missing answers
    - Tone/format mismatches
    - Incorrect reasoning
- Generates evaluation score sheets
- Supports multiple testing methods:
    - Blind tests
    - Prompt-based scoring
    - Error classification

Purpose

Ensures the chatbot meets professional quality standards before deployment.

Overall Impact of Features

Together, these features deliver a complete AI solution capable of:

- Extracting knowledge from any website
- Understanding linguistic patterns
- Generating optimized prompts
- Preparing structured datasets
- Fine-tuning LLMs
- Deploying a functional chatbot
- Evaluating AI performance scientifically

This ecosystem transforms unstructured website FAQs into a **production-ready AI assistant**.

8. Data / Dataset Design

The project does **not use any traditional database** such as MySQL, MongoDB, or PostgreSQL. All components work entirely through **file-based datasets**, which are sufficient and optimal for LLM fine-tuning workflows.

Instead of tables or collections, the system uses structured **JSON / JSONL datasets**, each representing a different processing stage in the pipeline.

A. Dataset Layers

1. Raw Extracted Dataset

Contains unprocessed website FAQ data.

```
{
  "question": "...",
  "answer": "...",
  "source_url": "..."
}
```

2. Processed Dataset (After NLP Analysis)

Includes tone, domain, persona, and intent metadata.

```
{
  "question": "...",
  "answer": "...",
  "domain": "...",
  "tone": "...",
  "intent": "..."
}
```

3. Fine-Tuning Dataset (OpenAI JSONL Format)

Used for GPT-3.5 training.

```
{
```

```
  "messages": [
    {"role": "system", "content": "..."},
    {"role": "user", "content": "..."},
    {"role": "assistant", "content": "..."}
  ]
}
```

B. Why No Database Was Required

- The entire workflow is **pipeline-driven**, not CRUD driven
- Data is static and only used for **model training & evaluation**
- JSONL is the recommended structure for LLM fine-tuning
- Versioning is easier with file-based datasets
- Faster processing for NLP and training tasks

C. Optimization Methods

To ensure clean and high-quality training data:

- Duplicate removal
- Regex-based cleaning
- Length normalization
- Dataset balancing
- Removing invalid or incomplete entries

D. Summary

A database was **not necessary** because the work involves:

- NLP pipelines
- Data transformation
- LLM fine-tuning
- File-based datasets

This approach keeps the system simple, fast, reproducible, and fully aligned with OpenAI's recommended training format.

9. API Architecture

This project relies entirely on **OpenAI's official API ecosystem**. No custom backend, database, or server framework is used. All workflows—training, evaluation, and inference—are executed through **Python scripts calling OpenAI APIs**.

A. Architecture Pattern

Client (Python Scripts) → OpenAI API → Model Server → Chat/Test Interface

No backend services are required.
The Python scripts directly manage training, model retrieval, and inference.

B. Key OpenAI API Endpoints

1. File Upload API

`POST /v1/files`
Used to upload training and evaluation JSONL datasets.

2. Fine-Tuning Jobs API

`POST /v1/fine_tuning/jobs`
Creates and manages fine-tuning jobs and model versions.

3. Model Access (Chat Completion) API

`POST /v1/chat/completions`
Used by the final chatbot to generate responses with the fine-tuned model.

4. Model Retrieval API

`GET /v1/fine_tuning/jobs/{job_id}`
Returns job status, logs, and the final model identifier.

C. Authentication

All requests use:
`Authorization: Bearer <OPENAI_API_KEY>`

API key is stored in environment variables, never in code, and rotated regularly.

D. Request / Response Examples

Fine-Tuning Request

```
{
  "training_file": "file-xyz",
  "model": "gpt-3.5-turbo",
```

```
  "suffix": "faq-domain-model"
}
```

Chat Completion Response

```
{
  "id": "chatcmpl-xyz",
  "choices": [
    { "message": { "role": "assistant", "content": "..." } }
  ]
}
```

E. Error Handling

Handled through Python exception logic:

- Retry on rate limits/timeouts
- Validate datasets before upload
- Manage token constraints

Common issues: invalid JSONL formatting, token limit errors, or network issues.

F. Summary

The **entire backend** is OpenAI's API.
No additional infrastructure was needed—only:

- Python automation scripts
- OpenAI API endpoints
- Fine-tuned model outputs

10. Development Process

The system was developed using an **incremental, research-driven workflow**, ensuring each module—scraping, NLP analysis, dataset creation, fine-tuning, and chatbot deployment—was validated and optimized.

1. Requirement Analysis

- Automatic FAQ extraction
- NLP-based tone, domain, persona detection
- Dynamic system prompt generation
- JSONL training dataset creation
- OpenAI fine-tuning workflow
- Model evaluation and testing

2. Research Phase

Studied:

- Web scraping patterns
- NLP classification techniques
- OpenAI fine-tuning architecture
- Prompt engineering strategies
- Dataset formatting best practices

This established the technical blueprint for the project.

3. Prototype Development

Created small prototypes for:

- Scraper
- NLP Analyzer
- Prompt Builder
- Dataset Generator
- Fine-tuning Trigger
- Chatbot Test Console

Each validated one functional block.

4. Pipeline Integration

All modules were combined into a **seamless pipeline**:
`Extraction → NLP Analysis → Prompt Rules → Dataset → Fine-Tuning → Testing`.

5. Version Control

Used Git for:

- Script management
- Dataset versions
- Model versions (v1, v2, v3…)

6. Environment Setup

- Python
- Requests, BeautifulSoup, spaCy/NLTK
- JSON/JSONL datasets
- VS Code
- OpenAI fine-tuning API

7. Testing Approach

- Module testing (scraper, NLP logic)
- Pipeline testing
- Model evaluation (accuracy, tone consistency, relevance)

8. Optimization

- Cleaned noisy FAQ entries
- Balanced datasets
- Refined prompt rules
- Improved NLP attributes
- Adjusted token limits

9. Final Deployment

- Fine-tuned model deployed via OpenAI endpoint
- Chatbot connected and tested
- Evaluation completed
- Documentation prepared

Summary

A structured approach involving **research, prototyping, integration, testing, and optimization** resulted in a stable, scalable, and domain-accurate AI automation pipeline.

11. Challenges & Solutions

During the development of the AI automation pipeline, several technical and practical challenges arose. Each challenge was addressed systematically to ensure stability, accuracy, and reliability of the final system.

1. Noisy & Unstructured FAQ Data

Challenge:
Websites follow different HTML patterns, causing inconsistent and noisy FAQ extraction.

Solution:

- Applied regex-based cleaning
- Built heuristic rules for FAQ identification
- Removed duplicate and irrelevant entries
- Structured outputs into clean JSON

2. Ambiguous Tone, Domain & Persona Detection

Challenge:
NLP classification struggled with pages that mixed multiple tones or unclear context.

Solution:

- Combined rule-based + model-based classification
- Added keyword frequency scoring
- Implemented fallback logic when ambiguity was detected

3. Dataset Quality Issues for Fine-Tuning

Challenge:
Raw extracted data was not directly suitable for OpenAI fine-tuning (inconsistent structure, uneven sample lengths).

Solution:

- Standardized Q/A formatting
- Balanced dataset length
- Added system-level metadata
- Ensured strict JSONL compliance

4. Fine-Tuning Instability

Challenge:
Initial fine-tunes produced inconsistent or generic responses.

Solution:

- Improved prompt structure
- Added more contextual metadata
- Removed overly long / noisy samples
- Performed multiple model iterations (v1, v2, v3…)

5. Handling API Rate Limits & Errors

Challenge:
OpenAI API occasionally returned rate-limit or timeout errors.

Solution:

- Implemented retry logic
- Added cooldown delays
- Split large datasets into smaller batches

6. Ensuring Consistent Chatbot Behavior

Challenge:
Maintaining tone, persona, and domain alignment across responses was difficult.

Solution:

- Strengthened system prompt rules
- Added tone consistency checks
- Evaluated using fixed benchmarking questions

7. Multi-Stage Pipeline Coordination

Challenge:
Managing multiple interconnected steps (scraping → analysis → dataset → fine-tune → evaluation) created workflow complexity.

Solution:

- Modularized each system component
- Added logging at each stage
- Implemented step-by-step verification before progressing

8. Evaluation & Testing Accuracy

Challenge:
Evaluating chatbot responses manually was time-consuming and subjective.

Solution:

- Created structured evaluation templates
- Added test scenarios across domains
- Used scoring rubrics (accuracy, completeness, tone)

Summary

Through systematic debugging, pipeline restructuring, dataset improvement, and iterative model optimization, all challenges were successfully overcome, resulting in a stable, accurate, and domain-specific AI system.

12. Key Learnings & Outcomes

The internship provided strong practical exposure to real-world AI development and significantly enhanced my technical and analytical skills.

1. Full AI Pipeline Understanding

Gained hands-on experience across the entire workflow—from scraping and NLP preprocessing to dataset creation, fine-tuning, deployment, and evaluation.

2. Dataset Engineering Skills

Learned to clean, structure, balance, and convert raw data into JSONL formats suitable for OpenAI fine-tuning, improving model performance and consistency.

3. LLM Fine-Tuning Expertise

Worked extensively with OpenAI APIs:

- Uploading datasets
- Running fine-tuning jobs
- Monitoring models
- Testing output quality

Developed the ability to iteratively improve model accuracy.

4. Applied NLP Knowledge

Improved skills in tone, persona, domain, and intent detection, combining rule-based logic with statistical NLP methods.

5. Prompt Engineering Proficiency

Learned to craft dynamic, rule-driven system prompts that control tone, structure, and domain alignment for high-quality model responses.

6. System Design & Automation

Developed modular pipelines with clean architecture, error handling, logging, and version management—essential for scalable AI systems.

7. Strong Problem-Solving Ability

Solved challenges involving noisy data, ambiguous context, API issues, and fine-tuning instability, strengthening analytical thinking.

8. Professional Growth

Enhanced collaboration, documentation discipline, and technical communication while working with mentors in a real project environment.

Outcome Summary

Successfully built a complete AI automation system capable of:

- FAQ extraction
- NLP context analysis
- Dataset generation
- LLM fine-tuning
- Chatbot deployment
- Model evaluation

This internship significantly advanced my readiness for professional roles in **AI, NLP, and Data Science**.

13. Conclusion

The internship at **Novas Arc Consulting Pvt. Ltd.** provided invaluable, hands-on experience in designing and implementing real-world AI systems using **modern NLP and LLM technologies**. Working across the entire AI pipeline—including **FAQ extraction, context analysis, prompt engineering, dataset preparation, OpenAI fine-tuning, chatbot development, and evaluation**—gave me a holistic understanding of how intelligent systems are **architected, optimized, and deployed at scale**.

This experience significantly enhanced my **technical skills**, particularly in:

- **Data processing and dataset engineering**, ensuring high-quality inputs for LLMs.
- **NLP modeling and LLM customization**, including fine-tuning strategies and prompt design for domain-specific tasks.
- **System engineering**, integrating multiple components into a seamless, automated workflow.
- **Evaluation and testing**, measuring model performance and iteratively improving outputs.

Beyond technical expertise, the internship strengthened my **problem-solving abilities, workflow discipline, and technical communication skills**. Collaborating in a professional environment allowed me to navigate real-world challenges such as API limitations, error handling, and optimizing AI pipelines for efficiency and reliability.

Moreover, the project deepened my appreciation for **AI automation and scalable solutions**. I learned the importance of creating context-aware systems that not only provide accurate outputs but also handle user interactions gracefully, maintain robustness, and adapt to evolving requirements.

Overall, the internship **shaped my professional and technical growth**, preparing me to contribute effectively to advanced AI projects in industry. It also reinforced my enthusiasm for **building reliable, scalable, and intelligent AI solutions**, and has motivated me to continue exploring innovations in **LLM applications, NLP research, and AI-driven automation**.

This experience will serve as a strong foundation for future projects and career development, equipping me with both the **skills and confidence** to tackle complex AI challenges in professional environments.

Internship Certificate

15th Oct 2025,

Novas Arc
Address:Oval Building, Plot no.18,
iLabs Hyderabad Technology Park,
Inorbit Mall Rd Hyderabad,
Telangana India

TO WHOMSOEVER IT MAY CONCERN

This is to certify that Mr. Shikher Jain has successfully completed his internship at Novas Arc as a Data Science Intern. He demonstrated strong technical skills, professionalism, and a proactive attitude during his tenure.

His service record is as follows:

Name	:	Shikher Jain
Designation	:	Data Science Intern
Duration		11 Aug 2025 – 15 Oct 2025

Shikher Jain actively contributed to development projects, collaborated effectively with the team, and displayed a keen ability to adapt to new challenges. His work was valuable to our operations, and we appreciate his dedication.

We wish him the very best in his future endeavors.

On behalf of Novas Arc,

NOVAS ARC CONSULTING PVT. LTD. Hyderabad

KC Joe
HR Manager

Novas Arc Consulting Pvt. Ltd.
Level 2, Oval Building, Plot no.18, iLabs
Hyderabad Technology Park, Inorbit Mall Rd
Hyderabad, Telangana
India - 500081

Phone : +91 40 44334213
Email : connect@novasarc.com
Know about us at www.novasarc.com
CIN : U72900TG2022PTC161828