National College of Ireland

Configuration Manual

MSc Research Project
MSCCLOUD

Harish Challa
Student ID: 23417498

School of Computing
National College of Ireland

Supervisor: Shreyas Setlur Arun

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name: ……. ……Harish Challa……

Student ID: ……23417498……..……

Programme: ……MSCCLOUD…… **Year:** …2025-26….

Module: ……MSc Research Project……..……

Lecturer: …… Shreyas Setlur Arun……..……

Submission Due Date: ……11 Dec 2025……..……

Project Title: Optimised Latency in Cloud Systems: Multi-Level Caching through AWS-Based FDN

Word Count: ……959…..…… **Page Count:** ……6……..……..……

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.
ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature: ……Harish Challa……..……

Date: ……10 Dec 2025……

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	□
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	□
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	□

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Configuration Manual

Harish Challa
Student ID: 23417498

1 Introduction

This project presents a Function Delivery Network (FDN) designed to enhance serverless computing performance through an optimized multi-level caching architecture. It integrates an in-memory LRU cache as the first layer and DynamoDB as the second layer to reduce redundant computation and minimize end-to-end latency. When a user uploads a file through the Flask-based web interface, the system generates a unique hash and checks both cache layers before invoking AWS Lambda for processing. By comparing execution time across no-cache, single-cache, and multi-cache setups, the project demonstrates how strategic caching significantly improves efficiency, scalability, and response time in AWS-based applications.

2 Hardware Requirements

· Dual-core processor

· 8 GB RAM

· 20 GB free storage

· Stable internet connection

· System capable of running VS Code or Cloud9

3 Software Requirements

- Operating System: Windows / Linux / macOS
- Python 3.8+
- AWS CLI
- AWS Account
- VS Code or AWS Cloud9 IDE
- Browser (Chrome/Firefox)
- Flask
- boto3
- pip and virtual env

4 Technologies Used

- Backend: Python
- Frontend: HTML, CSS, JavaScript
- Framework: Flask
- Cloud Environment: AWS
- Services: Lambda, API Gateway, EC2, DynamoDB, S3, Cloud9
- IDE: VS Code / Cloud9

5 Libraries

boto3 : A Python SDK for interacting with AWS services programmatically. It enables operations such as invoking Lambda functions, accessing S3, managing DynamoDB, and more within Python applications.

base64 : A Python module used for encoding and decoding data in Base64 format. It is commonly used for safely transmitting binary data (like files) over text-based protocols.

hashlib : Provides secure hashing algorithms such as SHA-256 and MD5. It is used to generate unique, consistent hash values for files, passwords, or other data inputs.

numpy : A numerical computing library providing support for multi-dimensional arrays and high-performance mathematical operations. It is widely used for data processing, scientific computing, and array manipulation.

requests ; A simple and popular HTTP library for Python that allows sending GET, POST, and other HTTP requests easily. It simplifies communication with APIs and web services.

flask : A lightweight Python web framework used for building web applications and APIs. It is flexible, modular, and ideal for small to medium-sized projects that require custom routing and backend logic.

pypdf2 : A Python library used to read, extract text, merge, split, and manipulate PDF files. It enables easy processing of PDF content without requiring external tools.

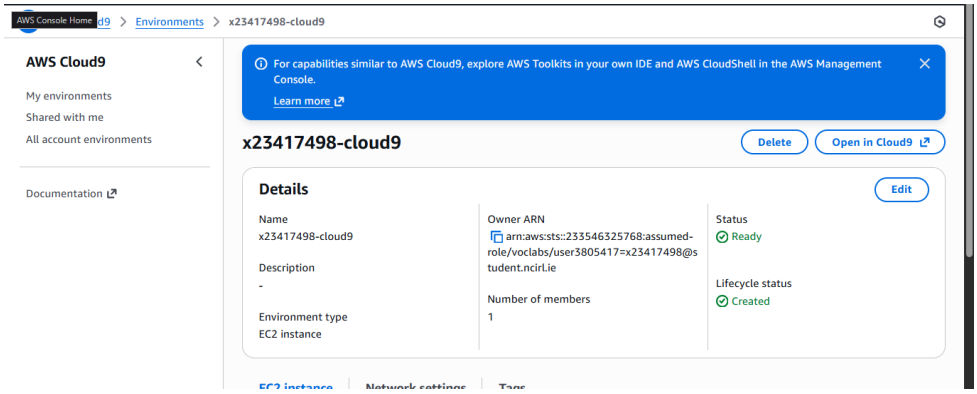
6 Steps

Create Flask Web app for

1. Extracting text from pdf
2. Extracting info of txt file like no. Of words, no. of lines
3. For managing and checking the requests and responses

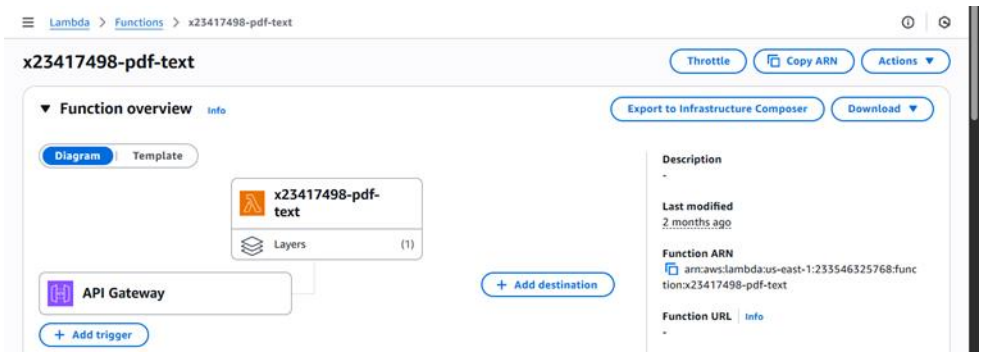
Configure Cloud Services

1. Cloud9 : For coding, testing and deployment

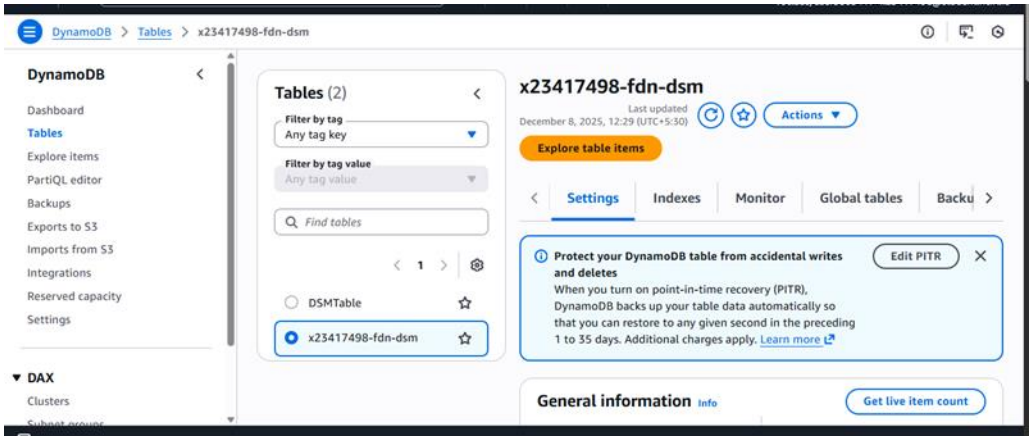


2. EC2 : created while creating cloud9

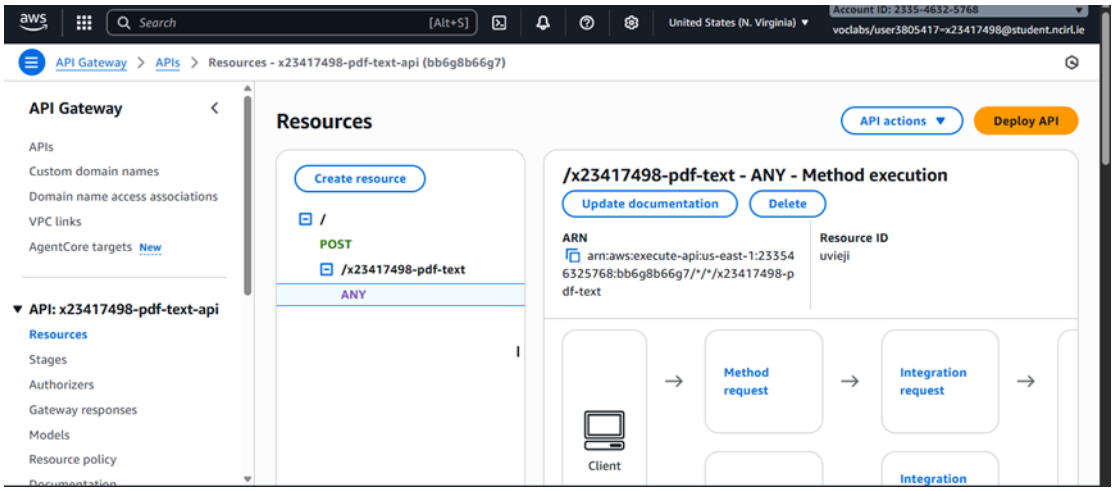
3. Lambda : running the logical code online



4. DynamoDB : for storing the cache data of requests and response



5. API Gateway : for triggering the lambda function



7 Project Flow

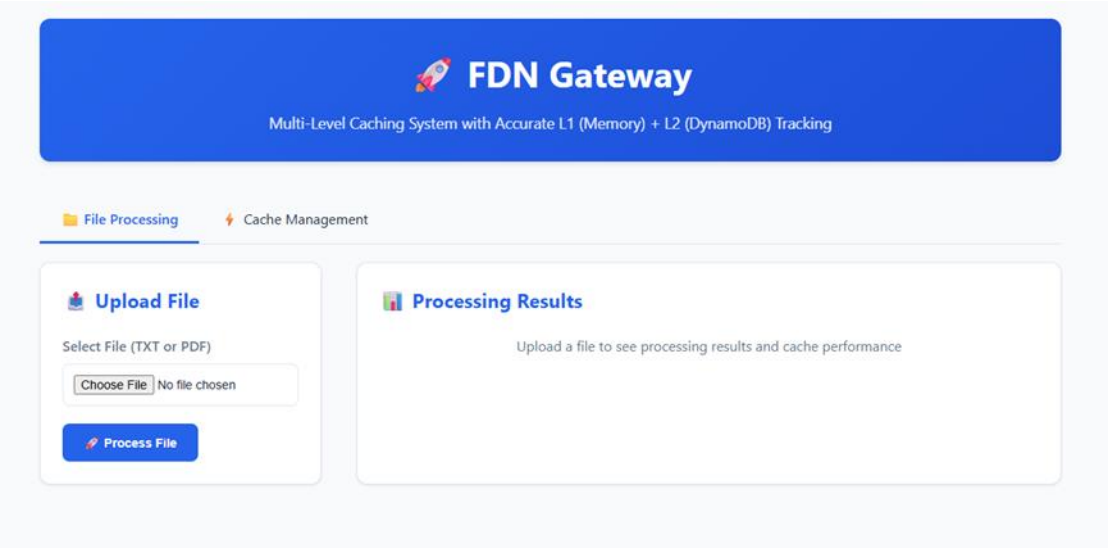
Experiment 1

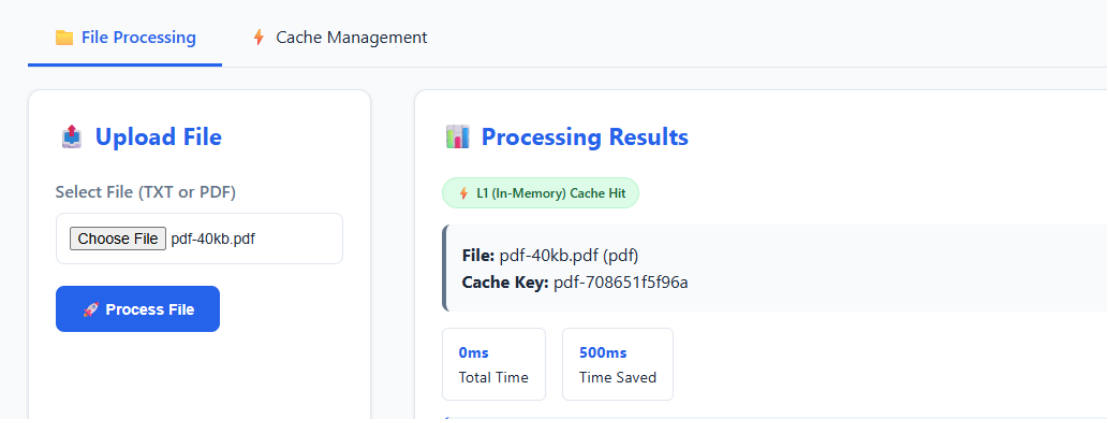
1. Client open flask web app
2. upload txt doc file to extract text
3. generate a key from file and hash
4. it will check the key in level1 chache (in-memory) using LRU(Least Recently Used) for available response
5. if it gets the response it will return it from there if not it will go to level2 cache (DynamoDB)
6. if it gets response in level2 cache it will return it
7. if not it will send request to the lambda function to process the file and return the response
8. get the metrics

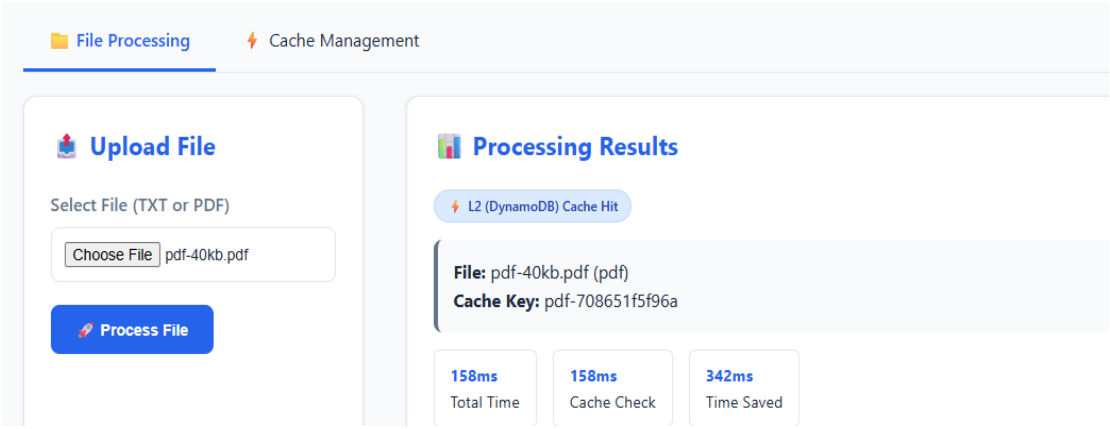
Experiment 2

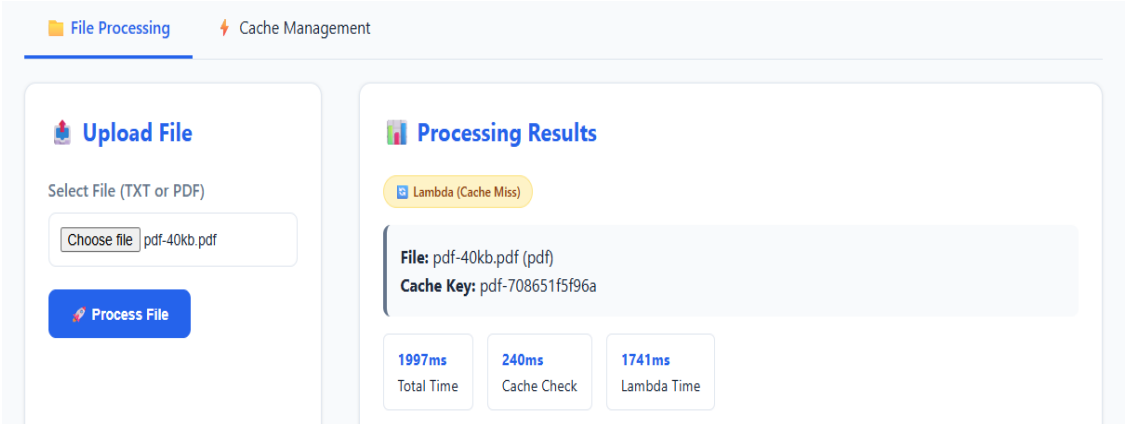
1. Client open flask web app
2. upload pdf doc file to extract text
3. generate a key from file and hash
4. it will check the key in level1 chache (in-memory) using LRU(Least Recently Used) for available response
5. if it gets the response it will return it from there if not it will go to level2 cache (DynamoDB)
6. if it gets response in level2 cache it will return it
7. if not it will send request to the lambda function to process the file and return the response
8. get the metrics

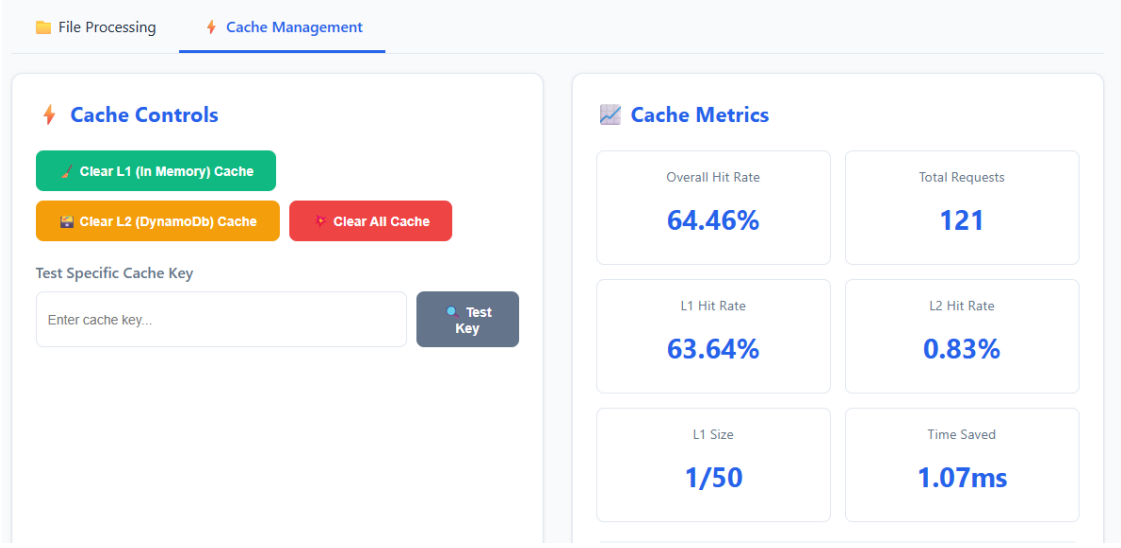
8 Output






File Processing
Cache Management
Upload File
Select File (TXT or PDF)
Choose File
pdf-40kb.pdf
Process File
Processing Results
L2 (DynamoDB) Cache Hit
File: pdf-40kb.pdf (pdf)
Cache Key: pdf-708651f5f96a
158ms
Total Time
158ms
Cache Check
342ms
Time Saved


File Processing
Cache Management
Upload File
Select File (TXT or PDF)
Choose file
pdf-40kb.pdf
Process File
Processing Results
Lambda (Cache Miss)
File: pdf-40kb.pdf (pdf)
Cache Key: pdf-708651f5f96a
1997ms
Total Time
240ms
Cache Check
1741ms
Lambda Time


File Processing
Cache Management
Cache Controls
Clear L1 (In Memory) Cache
Clear L2 (DynamoDb) Cache
Clear All Cache
Test Specific Cache Key
Enter cache key...
Test Key
Cache Metrics
Overall Hit Rate
64.46%
Total Requests
121
L1 Hit Rate
63.64%
L2 Hit Rate
0.83%
L1 Size
1/50
Time Saved
1.07ms