SOFTWARE REPORT

OF

EXPENSE MANAGER

Version 1.0

MD Arman Sakif Chowdhury
(110 175 261)

December 12, 2024

Contents

1. Introduction

1.1. Purpose

The purpose of this project was to practically apply and enhance my Flutter development skills by building a functional application. Additionally, I wanted to create a personalized Expense Manager app to effectively track and manage my own expenses. This project also served as an opportunity to integrate a backend powered by AWS and Python, deepening my understanding of full-stack mobile app development.

1.2. Learning Resources

To build this application, I utilized knowledge gained from completing three comprehensive courses on Udemy, covering Flutter, AWS, and Python. These courses provided the foundational and advanced skills necessary to create a real-world application:

1. Flutter & Dart - The Complete Guide [2024 Edition]
2. Ultimate AWS Certified Solutions Architect Associate 2025
3. 100 Days of Code: The Complete Python Pro Bootcamp

1.3. Project Scope

The Expense Manager app was designed to provide a practical application of Flutter development while offering a functional tool for personal finance management. The scope of the project includes:

1. **Frontend Development:**
   - Implementing an intuitive and user-friendly interface using Flutter and Dart.
   - Designing features for adding, categorizing, and viewing expenses in a visually appealing manner.
2. **Backend Development:**
   - Creating a robust backend using Python to handle data processing and storage.
   - Leveraging AWS services for cloud-based data storage, authentication, and backend infrastructure.

3. **Integration:**
   - Seamlessly connecting the Flutter frontend with the AWS and Python-powered backend for real-time data synchronization.
4. **Features:**
   - Expense tracking with categorization and date filtering.
   - Visualization of expense data through graphs or summaries.
   - Secure data storage and user authentication mechanisms.
5. **Learning Outcomes:**
   - Applying theoretical knowledge from completed courses to develop a functional, real-world application.
   - Gaining hands-on experience with full-stack development using Flutter, AWS, and Python technologies.

1.4. Project Assumptions

The application is designed for personal use, with the assumption that I will be the sole user.

1.5. Deliverables

The following deliverables were achieved as part of the Expense Manager project:

1. **Flutter Frontend:**
   - A fully functional and user-friendly mobile application interface.
   - Features for adding, categorizing, and viewing expenses.
   - Expense data visualization through charts and summaries.
2. **Python APIs:**
   - Custom-built APIs to handle data processing and communication between the frontend and backend.
   - APIs for expense creation, retrieval, and filtering.
3. **AWS Backend:**
   - Cloud-based infrastructure for secure data storage.
   - Implementation of user authentication and data management systems.

2. Planning

2.1. Team

The project was developed as an individual effort, with myself as the sole team member.

2.2. Stakeholders

Currently, I am the sole stakeholder of the Expense Manager app. Potential future stakeholders may include:

- **Friends or Family:** Individuals who might use the app if it is expanded for multiple users.
- **Potential Users:** Broader audiences, should the app be scaled and made public.
- **Developers or Collaborators:** Individuals who might contribute to future development or maintenance.

2.3. Risk Assessment

The following risks were identified during the project:

- **Unexpected AWS Costs:** Improper management of AWS resources could lead to unforeseen expenses.
- **Time Management:** Balancing project development with other responsibilities was a challenge.
- **Technical Challenges:** Potential issues with integrating the Flutter frontend with AWS and Python APIs, which could delay development.
- **Limited Testing:** As the sole developer and user, the app may lack thorough testing and debugging.

2.4. Initial Cost Analysis

The initial cost analysis for the Expense Manager project is detailed below:

Item	**Quantity / Hours**	**Cost (CAD)**
Developer Salary	60 hours	1800
AWS Budget	-	25
Total	-	**1825**

Table 2.1.: Initial Cost Analysis

3. Design

3.1. System Architecture

The expense manager system is designed as a web application. The system architecture comprises the following components:

- **Frontend**: A mobile-based user interface built using Flutter to allow users to interact with the system.
- **Backend**: A server-side application built using Python with Flask to handle business logic, data access, and API requests.
- **Database and Authentication**: AWS services, including PostgreSQL for database and Cognito for user authentication.

3.2. Data Design

The database schema for the expense manager system consists of two main tables. An ERD (Entity Relationship Diagram) 3.2 and a level 0 DFD (Data Flow Diagram) 3.1 were drawn to conceptualize the flow of data.

3.3. System Functionality

The expense manager system provides the following core functionalities:

1. **User Registration and Login:** Users can create accounts and log in to the system.
2. **Expense Entry:** Users can add new expenses, specifying the amount, product name, and date of the expense.
3. **Expense Viewing:** Users can view their expenses in a list or Pie Chart view.
4. **Monthly Expense Summary:** The system can generate a monthly summary of expenses.

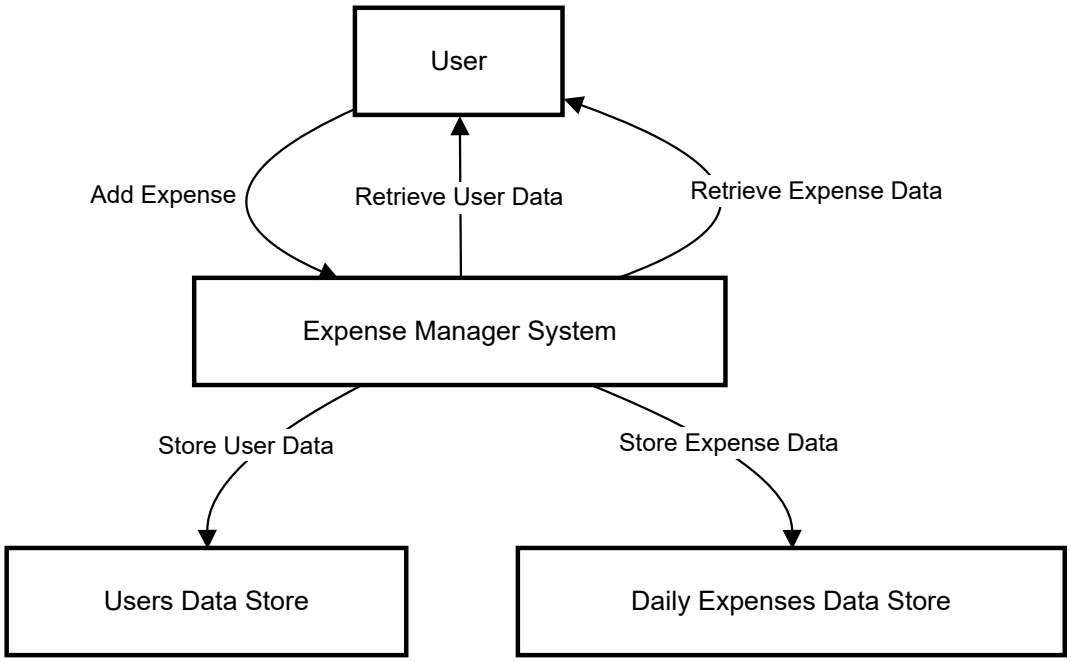


Figure 3.1.: level 0 DFD for Expense Manager

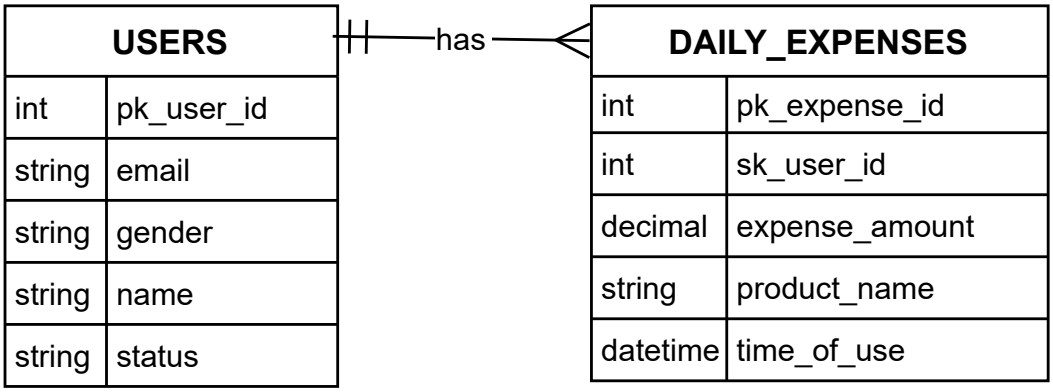


Figure 3.2.: Expense Manager ERD

4. Implementation

4.1. Actual Cost

Item	**Quantity / Hours**	**Cost (CAD)**
Developer Salary	60 hours	1800
AWS Bill	-	5.64
Total	-	**1805.64**

Table 4.1.: Current Cost Analysis

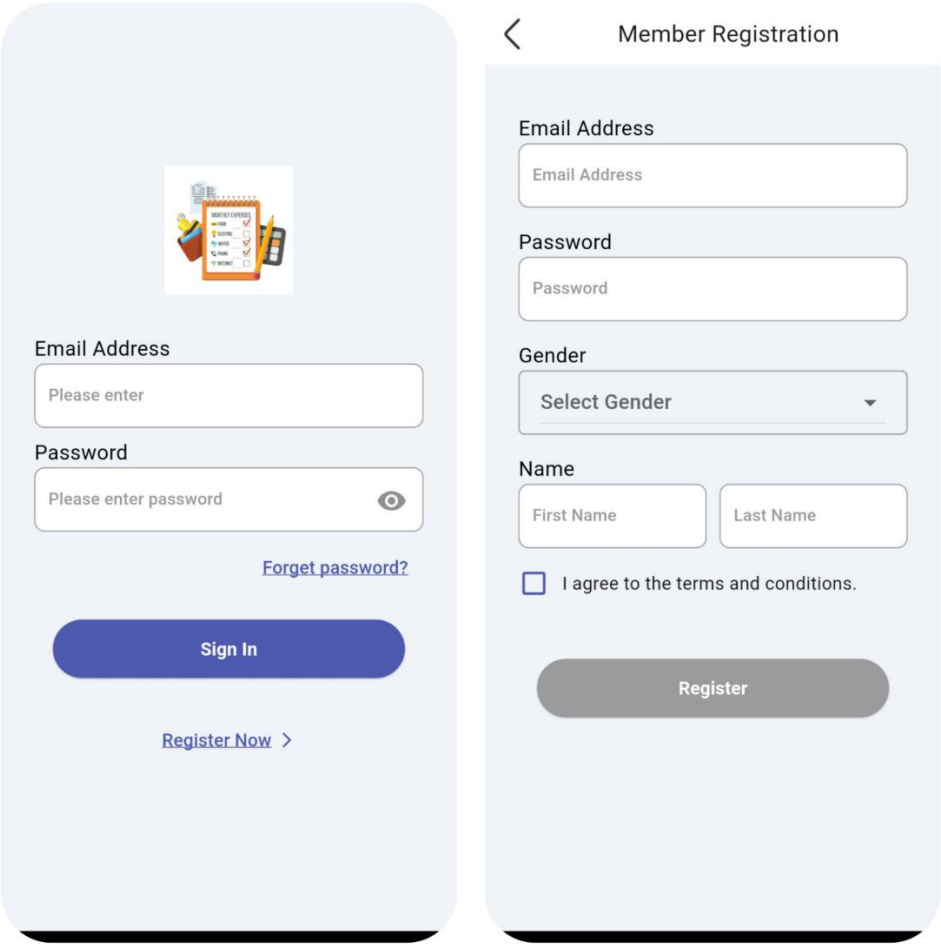


Figure 4.1.: Frontend Preview of Login and Signup

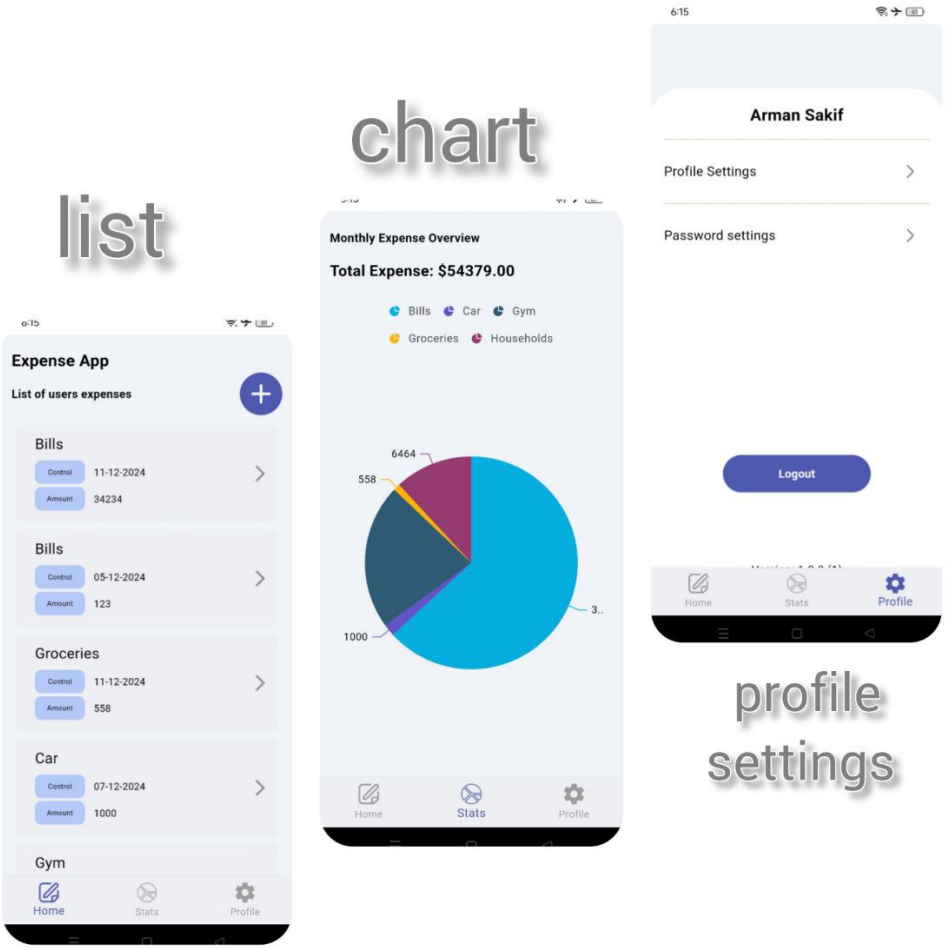
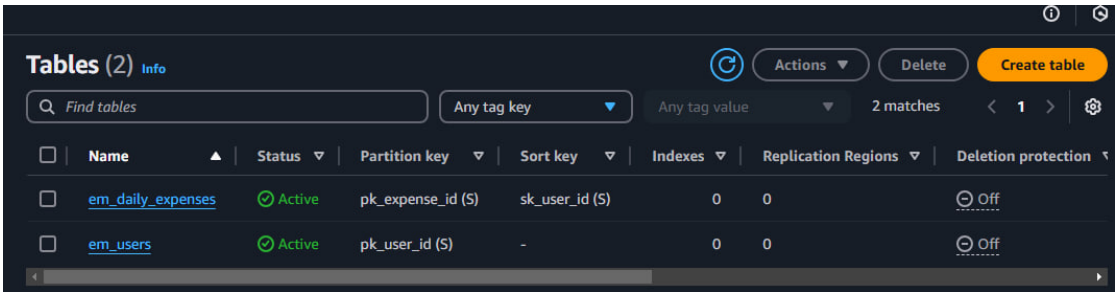


Figure 4.2.: Frontend Preview of Homepage and Features



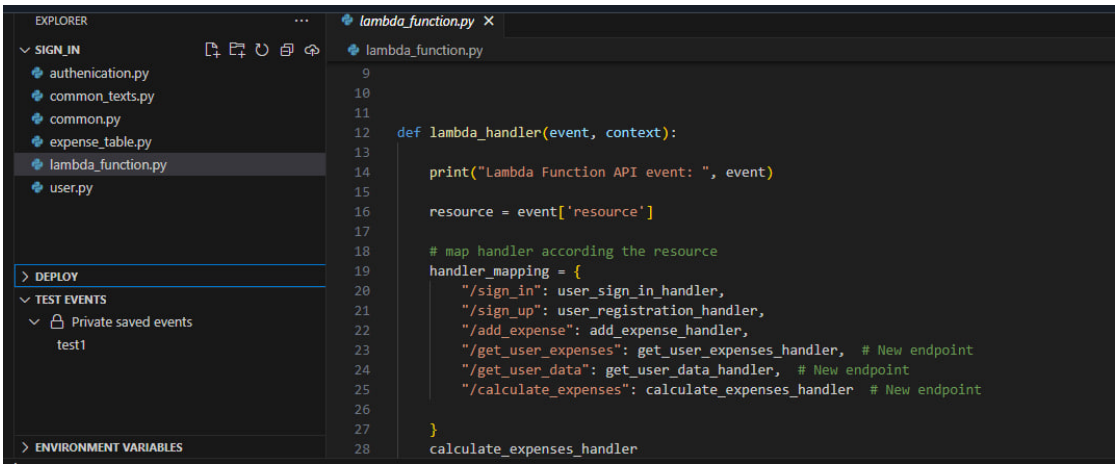
Tables (2) Info

Actions | Delete | Create table

Find tables | Any tag key | Any tag value | 2 matches | 1

Name	Status	Partition key	Sort key	Indexes	Replication Regions	Deletion protection
em_daily_expenses	Active	pk_expense_id (S)	sk_user_id (S)	0	0	Off
em_users	Active	pk_user_id (S)	-	0	0	Off

Figure 4.3.: Backend: Database Tables in AWS



EXPLORER

SIGN_IN

- authenication.py
- common_texts.py
- common.py
- expense_table.py
- lambda_function.py
- user.py

DEPLOY

TEST EVENTS

Private saved events

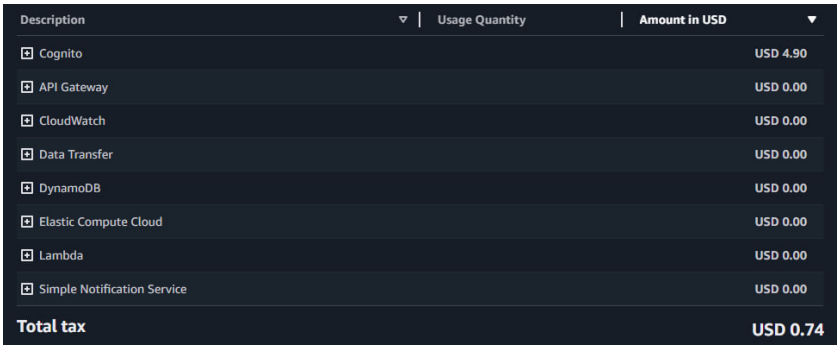
test1

ENVIRONMENT VARIABLES

lambda_function.py

```
def lambda_handler(event, context):

    print("Lambda Function API event: ", event)

    resource = event['resource']

    # map handler according the resource
    handler_mapping = {
        "/sign_in": user_sign_in_handler,
        "/sign_up": user_registration_handler,
        "/add_expense": add_expense_handler,
        "/get_user_expenses": get_user_expenses_handler,  # New endpoint
        "/get_user_data": get_user_data_handler,  # New endpoint
        "/calculate_expenses": calculate_expenses_handler  # New endpoint

    }
    calculate_expenses_handler
```

Figure 4.4.: Backend: Python APIs in AWS



Description	Usage Quantity	Amount in USD
Cognito		USD 4.90
API Gateway		USD 0.00
CloudWatch		USD 0.00
Data Transfer		USD 0.00
DynamoDB		USD 0.00
Elastic Compute Cloud		USD 0.00
Lambda		USD 0.00
Simple Notification Service		USD 0.00
Total tax		**USD 0.74**

Figure 4.5.: AWS Bill Breakdown

5. Conclusion

The Expense Manager project was undertaken to apply the knowledge gained from Udemy courses in Flutter, Python, and AWS to a real-world application. The primary objective was to develop a functional mobile application capable of tracking and managing personal expenses, which is stored in the cloud.

Through the development process, the project successfully implemented the core functionalities of expense tracking, categorization, and visualization. The Flutter framework was utilized to create an intuitive and user-friendly mobile interface. The backend, powered by Python AWS services, provided a robust and scalable infrastructure.

5.1. Future Scope

- including budgeting features
- using hashing to protect user data
- adding income
- net worth calculator
- financial advice based on income-expense
- voice inputs

A. Learning Flutter

- **Setup Instructions**
  - macOS Setup
  - Windows Setup
- **Introduction**
  - What is Flutter?
  - What is Dart?
  - The Concept of Widgets
- **Built-in Widgets**
  - Overview of Core Widgets
  - Custom Widget Creation
- **Debugging**
  - Debugging Tips and Tricks
- **Navigation**
  - Tab Navigation
  - Side Drawer Navigation
  - Stack-Based Navigation
- **State Management**
  - State Management Solutions (e.g., Provider, Riverpod, BLoC)
- **User Input**
  - Handling User Input
  - Input Validation
- **Backend Integration**
  - Sending HTTP Requests
- **User Authentication**
  - User Authentication Mechanisms
- **Google Maps Integration**

udemy

Certificate no: UC-83c93e6c-2e9a-48cb-8067-7ce48c8c7ab8
Certificate url: ude.my/UC-83c93e6c-2e9a-48cb-8067-7ce48c8c7ab8
Reference Number: 0004

CERTIFICATE OF COMPLETION

Flutter & Dart - The Complete Guide [2024 Edition]

Instructors **Academind by Maximilian Schwarzmüller, Maximilian Schwarzmüller**

Arman Sakif Chowdhury

Date **Dec. 6, 2024**
Length **30 total hours**

Figure A.1.: Udemy Flutter Course Certificate

  - Integrating Google Maps
- **Device Features**
  - Using the Device Camera
- **Animations and Transitions**
  - Creating Animations and Page Transitions
- **Image Upload**
  - Uploading Images to a Server
- **Push Notifications**
  - Manual Push Notifications
  - Automated Push Notifications
- **And Much More!**

[1]

[1] Flutter Course on Udemy

B. Learning AWS

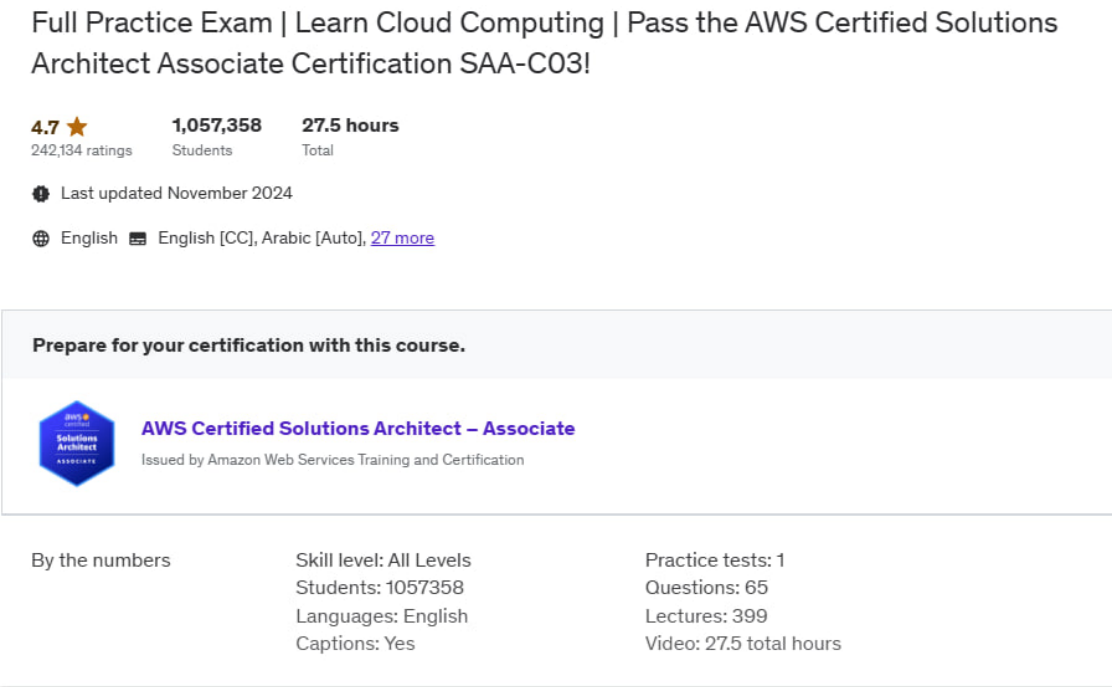


Figure B.1.: Learning AWS

[1]

[1] AWS Course on Udemy

C. Learning Python

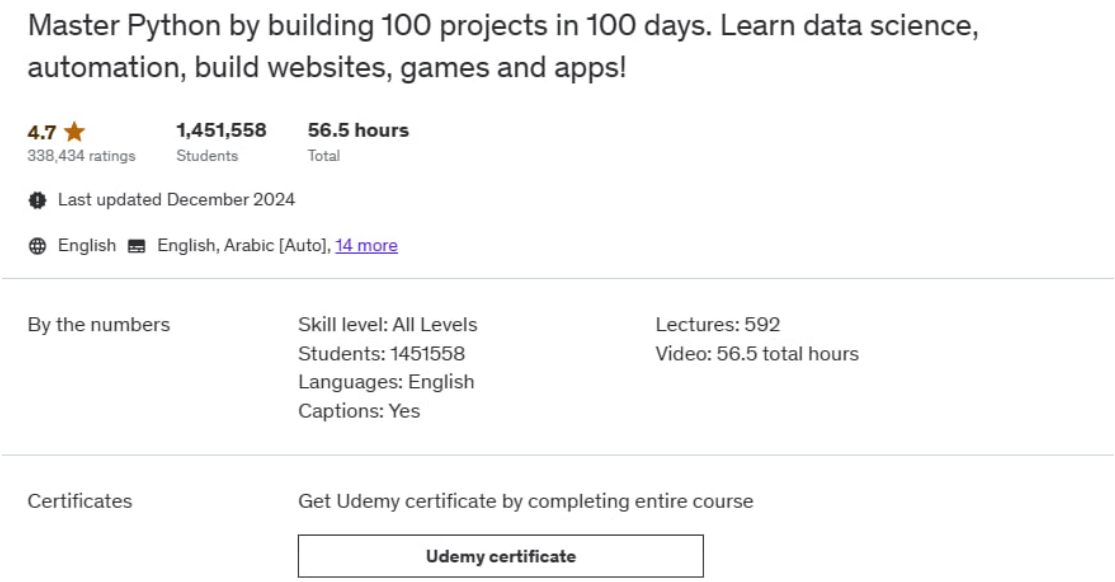


Figure C.1.: Learnig Python APIs

[1]

[1] Python Course on Udemy