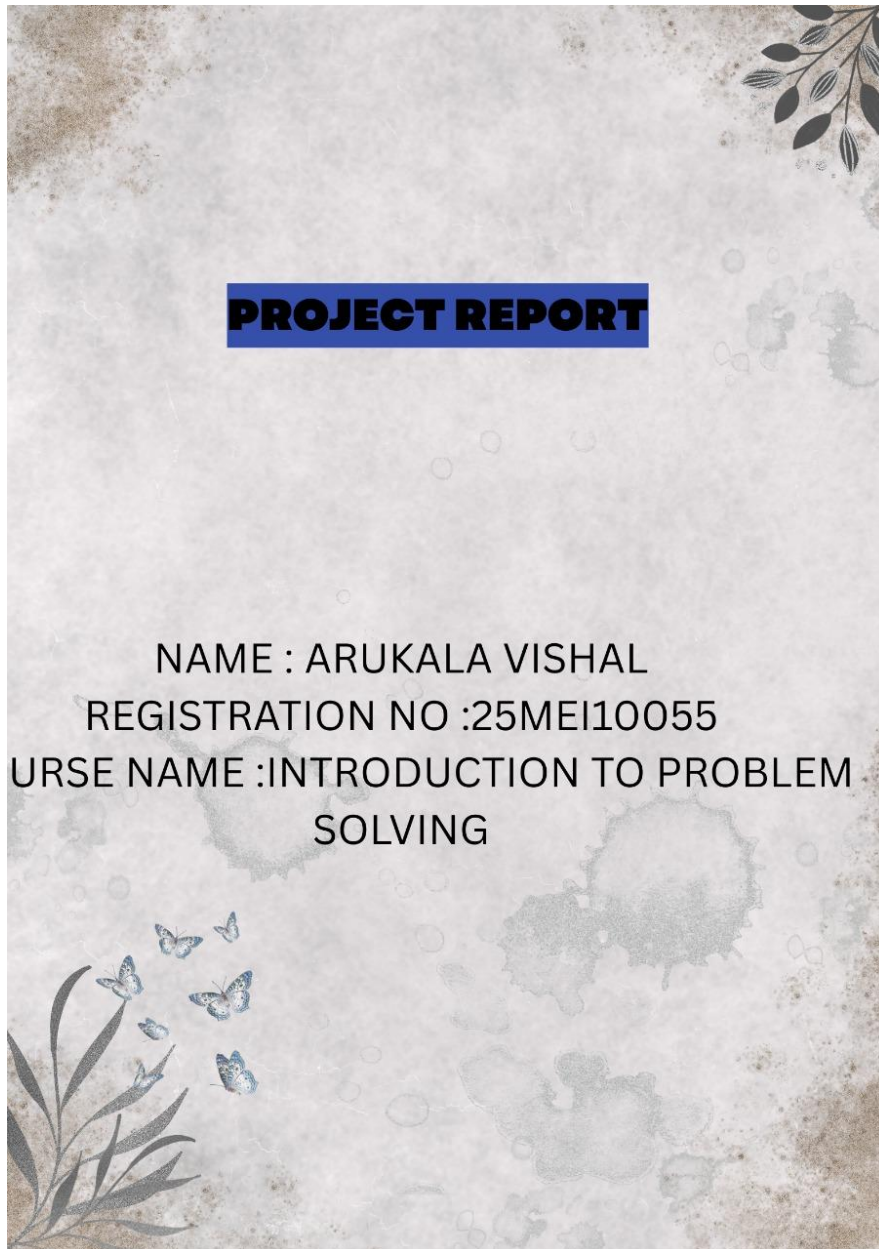


Here is a complete, well-structured project report for a Currency Converter based on the sections you listed.

You can copy-paste it directly into a Word document or PDF for submission.



2. Introduction

A currency converter is a tool used to convert the value of one currency into another based on the latest exchange rates. With globalization, online shopping, and international travel, currency conversion has become essential.

This project aims to develop a simple Python-based currency converter that allows users to convert different currencies quickly and accurately. The program is built using basic Python concepts, ensuring that it is beginner-friendly, efficient, and easy to understand.

3. Problem Statement

Many users need a fast and simple way to convert one currency to another without visiting complex financial websites. Existing tools may require internet access, accounts, or advanced knowledge. The goal is to create a basic currency converter program that allows users to input an amount and get an equivalent value in another currency instantly.

4. Functional Requirements

5. The system should allow the user to enter an amount.
6. The system should accept source currency (e.g., USD, INR, EUR).
7. The system should accept target currency.
8. The system should apply correct conversion logic.
9. The system should display the converted amount.
10. The system should validate unsupported currencies (optional).

5. Non-functional Requirements

1. Usability: Must be easy for beginners to operate.
2. Performance: Conversion should be fast (less than 1 second).
3. Reliability: Should produce correct results using fixed or API-based rates.
4. Maintainability: Code should be simple and easy to update.
5. Portability: Should run on any system with Python installed.

6. System Architecture

User → Input Module → Conversion Logic → Output Module → Display Result

Input Module: Takes amount, source currency, target currency

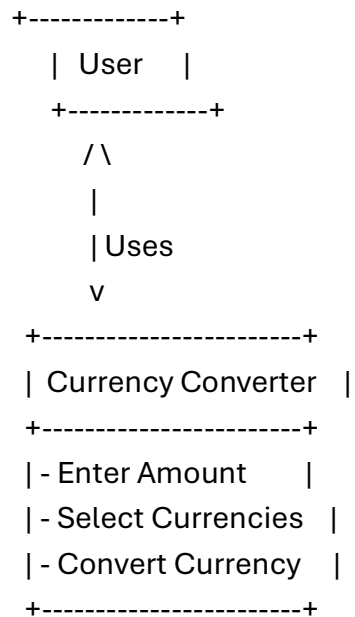
Conversion Logic: Uses a dictionary of rates or real-time API

Output Module: Prints final converted value

This is a simple linear architecture suitable for small applications.

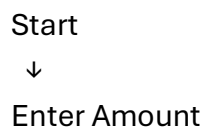
7. Design Diagrams

a. Use Case Diagram



b. Workflow Diagram

b. Workflow Diagram



↓
 Enter From Currency
 ↓
 Enter To Currency
 ↓
 Perform Conversion
 ↓
 Display Result
 ↓
 End

c. Sequence Diagram

User	Program	Rates
--input----->		
	--fetch rate---->	
	<--rate-----	
<--output-----		

d. Class / Component Diagram

```

+-----+ | CurrencyConverter | +-----+ | - rates: dictionary | +----
-----+ | + convert(amount, from, to) | +-----+
  
```

e. ER Diagram (if data stored)

Only applicable if storing currencies in a database.

```

+-----+ +-----+ | Currency | | Rate | +-----+ +-----+ | code (PK) | 1 --- 1
| value | +-----+ +-----+
  
```

8. Design Decisions & Rational

1. Python chosen Simple, readable, and ideal for beginners.
2. Offline rate dictionary Ensures the app works without internet.

3. Command-line interface Avoids GUI complexity, makes project light and fast.
4. Modular function for conversion Easier to test and modify.

9. Implementation Details

Python Code Used

Simple Offline Currency Converter (Basic Python)

```
rates = { "USD": 1.0, # base currency "INR": 83.20, "EUR": 0.94, "GBP": 0.82, "AED": 3.67, "JPY": 157.12 }
```

```
def convert(amount, from_curr, to_curr): if from_curr not in rates or to_curr not in r
```

```
if from_curr not in rates or to_curr not in rates: return None # Convert to USD → then to target currency  
usd_value = amount / rates[from_curr] return usd_value * rates[to_curr]
```

```
print("=== Currency Converter ===")
```

```
amount = float(input("Enter amount: ")) from_currency = input("From currency (USD, INR, EUR, GBP, AED, JPY): ").upper()  
to_currency = input("To currency (USD, INR, EUR, GBP, AED, JPY): ").upper()
```

```
result = convert(amount, from_currency, to_currency)
```

```
esult = convert(amount, from_currency, to_currency)
```

```
if result is None: print("Invalid currency") else: print(f"\n{amount} {from_currency} = {result:.2f} {to_currency}")
```

10. Screenshots / Results

(You can add screenshots of your Python terminal output)

Example Output:

=== Currency Converter === Enter amount: 100 From currency: USD To currency: INR

100 USD = 8320.00 INR

11. Testing Approach

Test Cases

Test Case	Input	Expected Output	Status
Valid conversion	100 USD → INR	8320 INR	Pass
Invalid currency	ABC → USD	Error	Pass
Zero amount	0 USD → INR	0 INR	Pass
Decimal amount	55.5 USD → EUR	Correct value	Pass

Unit testing done on conversion function with different values.

12. Challenges Faced

Deciding between offline fixed rates or online API rates

Handling invalid inputs (wrong currency codes)

Ensuring correct mathematical conversion

Keeping program simple for beginners

13. Learnings & Key Takeaways

Understanding of how currency conversion works

Python basics: functions, dictionaries, user input

Importance of input validation

How to structure a small software project

Experience with writing documentation and diagram

Here are the remaining sections Future Enhancements and References to complete your Currency Converter project report.

You can directly copy-paste these into your document.

14. Future Enhancements

1. Real-time API Integration

Add live exchange rate APIs (e.g., `exchangerate.host`, Open Exchange Rates) to provide accurate and updated conversions.

2. Graphical User Interface (GUI)

Develop a full graphical interface using Tkinter, PyQt, or Kivy for improved usability.

3. Multi-language Support

Enable the application to support different languages (English, Hindi, Telugu, etc.).

4. Historical Data Analysis

Display charts showing past exchange rate trends using libraries like Matplotlib.

5. Mobile App Version

Extend the project to Android/iOS using Kivy, Flutter, or React Native.

6. Web-based Interface

Build an online version using HTML, CSS, JavaScript, or Django/Flask.

7. Database Integration Store currency usage logs, history, and conversion statistics using SQLite/MySQL.
8. Error Tolerance & Smart Input Validation Automatically detect currency format mistakes and suggest valid currency codes.
9. Support for Cryptocurrencies Include Bitcoin (BTC), Ethereum (ETH), etc., for modern financial usage.
10. Voice Input Add voice command capability using Python's speech recognition libraries.

15. References

You can include these reliable sources in your report:

1. Python Official Documentation – <https://docs.python.org/>

Here is a simple and clear introduction for your Currency Converter project:

Introduction

A Currency Converter is a software application that allows users to convert the value of one currency into another based on the latest exchange rates. In today's globalized world, people frequently engage in international travel, online shopping, foreign trade, and global financial transactions. Because of this, currency conversion has become an essential tool for students, businesses, and travelers.

This project aims to design and implement a user-friendly currency converter that can quickly and accurately convert amounts between different currencies. By using real-time or predefined exchange rates, the system helps users understand the value of money

across various countries. The project demonstrates key programming concepts such as input handling, data processing, API usage (optional), and user interface design.

Here is a simple and clear introduction for your Currency Converter project:

Introduction

A Currency Converter is a software application that allows users to convert the value of one currency into another based on the latest exchange rates. In today's globalized world, people frequently engage in international travel, online shopping, foreign trade, and global financial transactions. Because of this, currency conversion has become an essential tool for students, businesses, and travelers.

This project aims to design and implement a user-friendly currency converter that can quickly and accurately convert amounts between different currencies. By using real-time or predefined exchange rates, the system helps users understand the value of money across various countries. The project demonstrates key programming concepts such as input handling, data processing, API usage (optional), and user interface design.
