**Proposed Architecture for Email Sender Service**

**Overview**

This document outlines the proposed architecture for an Email Sender Service designed send different emails. The service will support dynamic email template rendering, logging of emails, and retry mechanisms for failed email deliveries. The architecture is built to be flexible, scalable, and reliable, making it suitable for use by different types of clients such as web applications, mobile applications.

**Architectural Components**

**Client Applications**

These are various types of client applications that will interact with the Email Sender Service via the API. Examples include:

* Web Applications
* Mobile Applications

**Multiple Layers**

* **API Layer:** This will be the entry point of the service to handle requests.
* **Application Layer**: Responsible for sending, templating, storing emails and retry mechanism.
* **Domain Layer:** Will have email related entities which will be used to store data in database
* **Infrastructure Layer:** For interacting with database and sending emails this will also have.

**Flow Diagram**

Send request

API

Client

calls

Return Response

Application Layer  
(Email Manager,Retry Scheduler)

Eamil Sender(for content and send email)

Rquest for send Email

Calls Repo

Smtp Email Sender

Insert into database

Data Base

Repository

**(Note:** In the above diagram Email manager inserts entries and Retry Service may update entries according to table structure**)**

**Architecture Diagram**

Application Layer   
 (EmailManager, EmailSenders, SchedulerService, RetryScheduler, TemplateService)

Domain Layer   
 (EmailRequest, EmailStatus, EmailDTOs.)

Client Applications   
 (Web Apps, Mobile Apps, Console Apps

, etc.)

Data Base Layer

Infrastructure Layer   
(Smtp Handler, Repositories)

API Layer  
(EmailController)