# Implementation of a Basic JWKS Server

**Overview**

This FastAPI-based server provides JWT authentication using RSA key pairs. It dynamically generates and rotates RSA keys, exposes a JWKS (JSON Web Key Set) endpoint, and issues JWT tokens signed with the latest keys.

**Features**

-> Generates RSA key pairs (public/private) dynamically

-> Key rotation every 10 minutes

-> Expired key cleanup in a background process

-> JWKS endpoint (/.well-known/jwks.json) for public key retrieval

-> JWT issuance with support for valid & expired tokens

**Technologies Used**

-> FastAPI (API framework)

-> PyJWT (JWT handling)

-> Cryptography (RSA key management)

-> Threading (Background key cleanup)

**Installation**

**Clone this repository:**

-> git clone <https://github.com/TheCodingWizard27/3550-Project.git>

**Install the required libraries:**

-> pip install -r requirements.txt

**Running the Server**

Start the FastAPI server using uvicorn:

-> uvicorn project1:app --host 127.0.0.1 --port 8080 --reload

**Once running, access:**

**-> JWKS Endpoint:** http://127.0.0.1:8080/.well-known/jwks.json

**-> Authentication Endpoint:** POST /auth

**API Endpoints**

**1. Get JSON Web Key Set (JWKS)**

**Endpoint:** GET /.well-known/jwks.json

-> Returns the public keys used to verify JWTs.

**2. Authenticate & Get a JWT Token**

**Endpoint:** POST /auth

-> Issues a JWT token, signed with an RSA private key.

## **Running Tests**

This project includes a test suite using pytest.

To run the tests, execute:

-> pytest --cov=project1 --cov-report=term-missing

This ensures **80%+ test coverage**, validating key functionality such as:

* RSA key generation and rotation.
* JWT issuance with valid and expired keys.
* JWKS endpoint response.
* Expired key cleanup.