🔐 University and Student Server Applications

This repository hosts two **Flask-based web applications** designed to simulate secure interactions between a university administration and students. It provides functionalities such as secure student registration, issuing digitally signed grade reports, and degree certificates.

📂 Project Structure

Here's a detailed breakdown of each component:

```
project/
├── app.py # University Server Flask Application
│   ├── Manages student registrations and credentials securely.
│   ├── Automatically generates RSA keys for signing documents.
│   ├── Issues digitally signed PDFs (grades, degrees).
│
├── user_app.py # Student Server Flask Application
│   ├── Allows secure registration and login for students.
│   ├── Securely requests documents from the University Server.
│   ├── Verifies digitally signed academic documents.
│
├── rsa_utils.py # RSA Utilities for Encryption/Signing
│   ├── Generates RSA key pairs (public/private keys).
│   ├── Implements encryption, decryption, and digital signatures.
│   ├── SHA-256 hashing for secure signature creation and verification.
│
├── database.json # JSON Database for Persistence
│   ├── Stores university details (public/private keys).
│   ├── Stores student details (registration info, keys, grades).
│   ├── Courses information managed centrally.
│
├── static/
│   └── certificates/ # PDFs created by the University Server
│
└── pdfs/ # PDFs stored by the Student Server for verification
```

🚀 Setup and Installation Guide (Step-by-Step)

1. Prerequisites

Ensure your system has:

1. Python 3.x installed.
2. `pip` (Python package installer).

2. Install Required Libraries

Run the following commands in your terminal:

pip install flask requests reportlab

1. **Flask**: For web application creation.
2. **Requests**: To handle HTTP requests securely.
3. **ReportLab**: To dynamically create PDF certificates.

🛠️ Running the Applications

Each application runs independently:

University Server

Run the following command in your terminal from the project root:

python app.py

1. The server runs at `http://localhost:5000`.

Student Server

In another terminal window, execute:

python user_app.py

1. The server runs at `http://localhost:5001`.

📌 Functionalities (Deep Dive)

🔑 RSA Encryption and Digital Signatures

1. **Automatic RSA Key Generation**:
2. Keys automatically generated for both university and each registered student.
3. **SHA-256 Hashing**:
4. Ensures strong data integrity and authenticity.
5. **Encryption & Decryption**:
6. Secure communication of student data and document requests.

📃 Dynamic PDF Generation

1. PDFs generated using ReportLab for academic documents.

2. PDFs digitally signed by the university server using RSA signatures.
3. Students download and verify the authenticity of these PDFs.

🌐 Secure API Communication

1. JSON-based requests and responses encrypted using RSA.
2. Secure endpoints established for sensitive operations like registration and document requests.

📡 API Endpoints (Detailed)

University Server (`localhost:5000`)

Method	Endpoint	Description
GET	`/api/public_key`	Retrieves university's RSA public key.
POST	`/api/register`	Registers a new student securely.
POST	`/api/request_grades`	Provides digitally signed grade reports.
POST	`/api/request_degree`	Provides digitally signed degree certificates.

Student Server (`localhost:5001`)

Method	Endpoint	Description
GET/POST	`/register`	Handles student registration securely.
POST	`/login`	Authenticates students securely.
POST	`/request_grades`	Requests and verifies grade reports.
POST	`/request_degree`	Requests and verifies degree certificates.

🔐 Security Implementation (In-Depth Explanation)

The application leverages RSA encryption to provide robust security:

1. **RSA Key Generation**:

2. Random primes used to create secure RSA key-pairs for both university and students.
3. **Digital Signatures**:
4. Data integrity ensured through SHA-256 hashing combined with RSA encryption.
5. **Encrypted Communication**:
6. Sensitive student data is encrypted during transmission to prevent unauthorized access.

📂 File Descriptions (Detailed)

1. `app.py`:
2. Manages student registration, authentication, RSA key management.
3. Issues digitally signed PDF certificates.
4. `user_app.py`:
5. Handles student interactions, login, and document verification.
6. Securely communicates with the University Server.
7. `rsa_utils.py`:
8. Contains essential cryptographic functions including:
9. Prime number generation.
10. RSA key-pair generation.
11. Encryption/decryption mechanisms.
12. Digital signature creation and verification.
13. `database.json`:
14. Persistent JSON database holding university info, student credentials, RSA keys, and course details.