Semantic Integrity API demo

This demo provides a simple test case to explain how the semantic integrity (SI) API should work between IABG and Secublox.

Explanation

There are two scripts contained in this folder:

* `client_mock.py`: represents a simplified version of the semantic integrity (SI) evaluation on IABG side.
This script retrieves SI evaluation requests from the Secublox SI API, and posts results back as soon as they are ready.

* `server_mock.py`: represent a simplified version of the Secublox SI API handling the semantic integrity part.
This script keeps track of pending evaluation requests (for now for simplicity, stored into a file `pending_requests.txt`)
and posts the evaluation requests to a public endpoint (`/requests_pending_semantic_integrity`) to that the IABG client
can retrieve them. It also exposes a public endpoint (`/semantic_integrity_results`) for enabling
the IABG client to post results back.

The workflow is the following:

0) A user of the Secublox platform wants to verify if an image has been semantically manipulated or not.
1) The user sends a request for SI in the UI. The user has an `candidate_image` as input. As a first step, the candidate image must be associated to a `watermarked_image` through image verification. When the image is verified, we have access to both `candidate` and `watermarked` image. For simplicity, in this demo this association is already present in `pending_requests.txt`.
2) The Secublox platform frontend receives the request of SI verification and adds it to a queue / DB. In our scenario, as said before we use a file
`pending_requests.txt`
3) The SI API on Secublox side checks the content of the queue / DB (file in our case), and determines there is a pending request to signal
to the IABG client. The endpoint now exposes this pending request under `/requests_pending_semantic_integrity`. Both images are ready to be downloaded under this endpoint.
4) The IABG SI client periodically checks the status of the SI API on Secublox end (e.g. every minute), using a GET method.
After a while, it receives a new pending request. It processes the request by computing the SI
prediction for the image pair. The final `prediction` is a boolean (Manipulated/Non-manipulated) plus some `metadata_info` about the prediction (e.g. confidence score). These results are posted to the

`/semantic_integrity_results` on Secublox SI end, using a POST method.
5) The SI API on Secublox side receives the SI results from IABG and updates the UI and DB / queue to take note of the result.
6) This loop (4–6) continues till all requests have been handled.

Setup

```
pip install uvicorn fastapi pydantic
cd deepshield_semantic_integrity
```

Usage

Start the mock server:

```bash
python server.py
```

A FastAPI endpoint gets started to post pending requests and receive semantic integrity results.

Now test the mock server with the mock client:

```
python client_mock.py
```

The client will retrieve pending requests from the server and post results as soon as they are ready.

After you have implemented the actual SI API on the Secublox platform, you can use our mock client
to test the correct working of the SI API.