<u>Review Group 42: By A1851002</u>

I reviewed the code both manually and using a tool, the tool I used was bandit, bandit is a tool designed to find common security issues in Python code. I found the README to be helpful and it did a good job of providing a list of instructions on how to run the code. Furthermore, the README also did a good job of describing what exactly the implementation does and how it functions. Here are some of the things I found while reviewing the code:

Possible Security risks in client_app_vulnerable.py:

Issue: [B106:hardcoded_password_funcarg] Possible hardcoded password: 'olafclient'
Severity: Low   Confidence: Medium

Issue: [B608:hardcoded_sql_expressions] Possible SQL injection vector through string-based query construction.
Severity: Medium   Confidence: Low

Possible Security risk in server.py:

Issue: [B605:start_process_with_a_shell] Starting a process with a shell, possible injection detected, security issue.
Severity: High   Confidence: High

```

        os.system(f'echo "$(date) : {new_filename} : {filename}" >> ./local/upload.log')

```

Furthermore:

- The paths to sensitive files (e.g., private keys) are hardcoded, which might expose them if the code is shared or logged.
- Sensitive Data Exposure: The database password is exposed in the source code, which is a security risk.
- I personally found that most of the vulnerabilities in the vulnerable files involved sensitive data exposure.