

Information Technology — Capstone Project Documentation Template (2025)

1. Executive Summary

Small businesses are at high risk of losing important data because they usually don't have reliable or automated backup systems. When accidents like system crashes, file deletions, or cyberattacks happen, they have no easy way to recover their information. This can cause major downtime, financial loss, and serious disruptions to their business. Our solution is to use our cloud backup system. Which will be cheaper than using any other expensive cloud-based storage. We made a simple-to-use and effective app for these small businesses that allows them to browse files and upload them to the cloud. This app has a history of all backups and an option to share files with other users in the app.

2. Problem & Requirements (O1)

Small businesses usually rely on costly cloud services or manual local backups like USB drives. These options are either too expensive or not reliable enough, leaving businesses without a simple and affordable way to protect their data. We need an app that can store files and data in the cloud, and that is also much more cost-effective than other methods of storage. Either way, small businesses should have a local copy of data, a cloud copy, and a remote off-site copy for the best protection against losing that data. With our app, we want to ensure that small businesses have a reliable cloud system to back up all of their data and important files.

3. System Overview & Architecture (O2)

System Design:

A system where users can backup files, restore data, check connections

Supports multiple cloud providers and selects the best & cost efficient one

Built using Streamlit(UI) with python functions

Backup files are saved and read using AWS 3 API

Object Design:

Records

User access list

Backup records

System Services

Log of backup history

Controls

share info between employees

restore and backup control

backup history lists

5. Implementation Notes (O2)

cloud_backup_streamlit/

- |— app.py # Main application logic (UI + backend functions)
- |— cloud_storage/ # Local file storage (simulated cloud bucket)
- |— backup_history.json # Stores user activity logs
- |— README.md # Project overview and setup instructions
- |— architecture.png # System design diagram
- |— use_case.png # Use case diagram
- |— one_page_overview.pdf # Summary documentation

6. Testing & Evaluation (O3)

The system was tested using a combination of basic unit testing and manual testing through the Streamlit interface. We have features such as file upload, download, deletion and history tracking which were verified to work together correctly. Key metrics include upload/download success rate, response time, and accuracy of history logs. Some limitations were identified, such as no automated tests, risk of file overwrites, lack of concurrency handling, and potential issues with JSON file growth or corruption.

7. Security, Privacy, Accessibility & Compliance (O5)

The system has features that include login authentication and role based access control, but it doesn't have encryption. User activity is logged locally which limits exposure. The Streamlit is simple and easy to use and suitable for demonstration purposes. It highlights the importance of secure data handling which also showing the risks of minimal protection in prototype systems.

8. Deployment & Operations.

-The system is deployed as a Streamlit web application with a Python backend, accessible locally or on a cloud platform. Users can back up and restore files through a simple interface, while automated scripts handle scheduled backups. Detailed setup and usage instructions are provided in the appendices.

9. Limitations & Future Work.

-The system has limitations in scalability, advanced security features, and automation. Future improvements include adding stronger security (e.g., MFA), better scalability, automated configuration, real-time monitoring, and support for multiple cloud providers.

10. References

-Amazon Web Services, "AWS Backup Documentation," 2024.

Streamlit Inc., "Streamlit Documentation," 2024.

Python Software Foundation, "Python Documentation," 2024.

Appendices (Evidence & How-To)

E. Poster (36"x48" horizontal)

 Poster

Slides

 powerpoint

Video Links (Intro, Demo)

Intro: <https://youtu.be/MT2uo-9H8ns>

Demo: <https://youtu.be/Tzx-lBf47qs>

Scrum Evidence Index (links to Sprint Planning, Dailies, Reviews, Retros)

https://drive.google.com/drive/folders/1aXIUHaHR01IZUj99EVNDxTUvWg8VQwxr?usp=drive_link

4. Discipline-Specific Depth (O6)

• Architecture & Integration:

This system uses a Streamlit frontend that allows users to upload, download, and delete files. The Python backend handles file processing, authentication, and backup history logging. Instead of using real cloud storage in the current prototype, uploaded files are stored in a local directory named "cloud_storage", while backup actions are recorded in backup_history.json.

The interface, application logic, and file storage are integrated in a single Streamlit application. In a future version, the local storage layer can be replaced with AWS S3 or another cloud provider without changing the main user workflow.

- **DevOps & SRE:**

The current prototype includes basic operational logging through a backup history file that records uploads and deletions by user and timestamp. Error handling is present for file upload and delete actions. However, the project does not yet include full CI/CD pipelines, infrastructure as code, or production observability tools such as centralized logs, metrics dashboards, or distributed tracing.

- **Operations & Security:**

The prototype includes simple access control through login credentials and role-based permissions, where admins can view full history and standard users can only view their own records. Operationally, the system supports backup upload, download, deletion, and history tracking. However, it still lacks production-level security measures such as encrypted storage, secret management, IAM policies, hardened cloud configuration, and formal change management.