Cost-Benefit Analysis

FileSender

SURF
Product Manager: William van Santen
Mentor: Rogier Spoor

Windesheim University of Applied Sciences
Supervising Teacher: Rob Kaesehagen
Student: Aaron Jonk (s1170298)

Date: 29/06/2025

Version: 1.0

Version management

Version	Date	What
1.0	29/06/2025	Creation of this document

Distribution

Version	Date	Recipient
1.0	29/06/2025	Published on Codeberg

Contents

1 Introduction

With the initial development of the proof of concept now complete, the FileSender community faces an important decision; how to proceed with future development. This document aims to provide a clear analysis of the potential costs and benefits associated with further investment in the project. It has been prepared based on the current set of requirements, as outlined in version 1.1.

This analysis assumes that the reader is already familiar with the accompanying documentation and understands the core objectives of the project. One of the primary goals is to make FileSender as secure as possible by emphasizing simplicity and eliminating external dependencies. At the same time, ease of installation remains a key requirement for broader adoption of this new version.

2 Baseline

Before exploring future decisions, it is important to establish what has already been achieved in the FileSender proof of concept. Based on the current requirements document, the following components have been implemented to a certain degree:

1. **MVP (all “must-have” features)**
2. **“Should-have” features:**
    a. Support for file encryption and decryption combined with chunking, enabling transfers up to 1 TB
    b. Download resumption after errors
    c. SAML and OIDC authentication, configured via an HTTP proxy server
    d. Pre-built Docker image for deployment
    e. Fedore package (thanks to François)

3 Estimated Costs

Estimating development costs in euros or any other currency would introduce inconsistency and require speculative assumptions about developer salaries. Instead, this section presents a time-based estimate of the effort needed to complete the remaining features. The estimates are expressed in weeks, providing a clearer and more neutral basis for planning.

The following is a list of features that still need to be implemented, along with an approximate number of weeks required for each.

Requirement	What	Effort
Client-side upload resume	Support resuming uploads on a different day or from a different device	~4 weeks
Multi-file transfers	Enable upload and download of multiple files in a single transfer	~6 weeks
FileSender CLI	A command-line interface for uploading and downloading files without using a browser	~8 weeks
Website templating, emails, email templating	Add support for customizable templates for the web interface and emails, including email sending functionality	~6 weeks
Guest vouchers	Allow users to invite others to upload files via guest vouchers	~6 weeks
Extended logs, usage and data metrics	Provide insights for both uploaders and FileSender administrators	~2 weeks
Built-in S3 backend	Enable automatic upload and download of files to and from S3	~4 weeks
Built-in SAML or OIDC and optional username/password	Provide built-in authentication using SAML or OIDC, or via username and password, without requiring a proxy	~6 weeks
Debian package	Package FileSender for Debian-based systems	~2 weeks
Scalability and distribution	Improve scalability through options like Helm charts and documentation for distributed deployment	~4 weeks

Total estimated time until full requirements are implemented: 48 weeks.

4 Expected Benefits

Implementing the remaining features outlined in the previous section is expected to significantly increase the value, usability, and adoption potential of this FileSender. The benefits can be grouped into several key areas:

4.1 Improved User Experience

Features such as multi-file transfers, upload resumption, and the command-line interface will make FileSender more convenient and flexible for end users. These enhancements reduce friction in daily use, especially for users handling large or time-sensitive files.

4.2 Greater Accessibility and Adoption

Guest vouchers and built-in authentication options remove barriers for external collaborators and simplify deployment for organizations that prefer not to use a reverse proxy. A broader range of users will be able to use the system securely with minimal setup.

4.3 Enhanced Customizability

Support for website and email templating allows institutions to better align the platform with their branding and communication needs. This improves trust and coherence across user touchpoints.

4.4 Visibility and Accountability

Extended logging and usage metrics are designed to provide users with clear insights into who downloaded which files and when. This level of transparency enhances trust, supports auditing needs, and offers valuable feedback for users managing sensitive or important file transfers.

4.5 Deployment Flexibility and Scalability

Features such as the Debian package, Helm-based deployment options, and built-in S3 support will make FileSender more attractive to a wide range of IT environments. These improvements lower the barrier to adoption for large institutions and make scaling more manageable.

4.6 Broader Integration and Automation Potential

With the addition of the FileSender CLI, users can now incorporate file transfers into automated workflows and scripts, opening new use cases such as schedules backups, research data pipelines, or CI/CD integrations.

5 Conclusion

The FileSender proof of concept has successfully delivered on its initial objectives, laying a strong technical foundation while aligning closely with the project's core principles of security through simplicity and ease of deployment. With the majority of "must-have" and several "should-have" features already implemented, the platform is well-positioned for its next phase of development.

Completing the remaining features, which are estimated to require approximately 48 weeks of work, would significantly enhance FileSender's functionality, flexibility, and appeal across a wide range of use cases. These improvements would not only elevate the end-user experience but also support wider institutional adoption by offering better integration, easier deployment, and more operational insight.

The decision to move forward with full implementation should consider both the relatively modest development effort still required and the substantial long-term benefits this investment would unlock. If the community prioritizes a secure, simple, and extensible file transfer platform, then advancing to the next stage of development is a logical and valuable next step.