SimpleStorage.cloud

CUSTOMER PRODUCT BRIEF

File sharing, made simple again.

Product capabilities, customer experience, cloud architecture and roadmap for a modern temporary file-transfer platform.

Prepared for prospective customers and partners

Product snapshot · 28 July 2026

DOCUMENT GUIDE

What this document covers

This brief explains the current SimpleStorage.cloud implementation, how customers experience the product, how its serverless AWS backend works, and which capabilities are planned next.

Accuracy note

“Implemented” means functionality exists in the current repository. “Foundation” means the interface or data model exists but a production integration is still required. “Roadmap” identifies a proposed feature and is not represented as available today.

01 · EXECUTIVE OVERVIEW

A focused, modern file-transfer experience

SimpleStorage.cloud is a lightweight file-transfer product designed around a straightforward promise: select files, send secure temporary links, and manage retained content without unnecessary workflow.

5 files
Maximum files in one transfer

100 MB
Current maximum per individual file

12 characters
Random public file identifier

Customer value

- No framework-heavy interface or complex onboarding for basic transfers.
- Direct browser-to-object-storage uploads avoid routing file bodies through Lambda.
- Temporary links obscure private bucket keys and original object paths.
- Registered users receive a dashboard, longer retention and Recycle Bin controls.
- Recipients receive branded email messages with individual download buttons.

Commercial direction

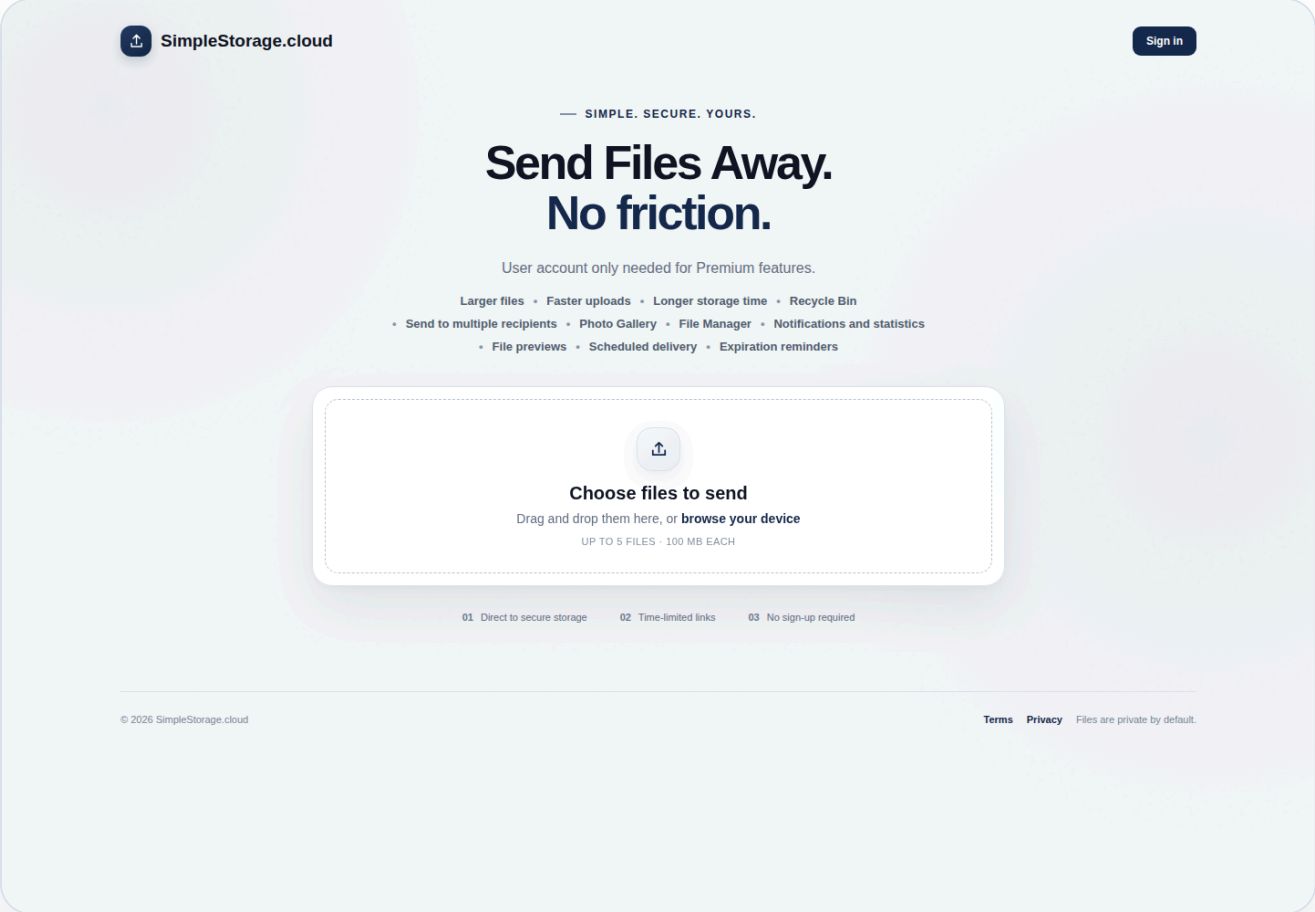
- A free transfer tier can use economical object storage and shorter retention.
- A Premium tier can use AWS S3, optimized upload techniques and richer management.
- The architecture separates frontend, identity, metadata and storage concerns.
- Administrative controls support account operations and customer support workflows.
- The product can evolve into a branded or white-label file-delivery platform.

Current product position

The repository is a functional serverless prototype with real upload, email, authentication, dashboard, preview, Recycle Bin and administrator foundations. Payment processing, subscription entitlement validation, production quota enforcement and several advertised Premium features remain roadmap work.

PRODUCT INTERFACE

A minimalist landing and upload experience



Current local build rendered at desktop resolution.

Clear entry point

The file-selection area supports drag-and-drop and traditional browser selection.

Visible constraints

Users see the maximum file count and per-file limit before starting a transfer.

Responsive identity

Navigation adapts to anonymous, signed-in and administrator sessions.

The interface uses plain HTML, CSS and readable JavaScript. There is no React, Vue, Angular or other client framework dependency.

02 · MAIN FEATURES

Capability map

CAPABILITY	STATUS	CUSTOMER OUTCOME
Multiple-file upload	Implemented	Select, remove and transfer up to five files in one operation.
Recipient email delivery	Implemented	SES sends branded HTML and plain-text email with individual links.
Temporary download links	Implemented	Random IDs resolve through DynamoDB to short-lived signed downloads.
Cognito accounts	Implemented	Sign-up, confirmation, sign-in, recovery and password changes.
User file manager	Implemented	Lists uploaded files, storage usage, expiry and common actions.
Recycle Bin	Implemented	Recoverable deletion, restoration and permanent removal.
Image preview	Implemented	Protected modal preview for JPEG, PNG, GIF and WebP.
Bulk operations	Mixed	Bulk download, send, recycle, restore and permanent delete foundations.
Administrator dashboard	Implemented	Account listing, disable/enable, reset, sign-out and user file visibility.
Subscription management	Foundation	Account interface exists; payment provider is not connected.
Scheduled delivery and reminders	Roadmap	Future time-based customer communications.
Advanced analytics	Roadmap	Download notifications, statistics and activity history.

03 · FRONTEND EXPERIENCE

Purpose-built browser workflows

Upload page

- Drag-and-drop or browser file selection.
- Five-file batch limit and 100 MB per-file validation.
- Per-file removal before transfer begins.
- One recipient for anonymous transfers; up to five for authenticated users.
- Terms acceptance and reCAPTCHA v3 verification.
- Progress feedback, generated links and email-delivery status.

Authentication pages

- Create account with name, email and password.
- Email-code confirmation and confirmation-code resend.
- Sign-in with optional remembered session.
- New-password challenge handling.
- Forgot-password and confirmation-code recovery.
- My Account password change through Cognito.

User dashboard

- File count and aggregate displayed storage.
- Scrollable file list with upload date, size and expiration.
- Active and Recycle Bin views.
- Download, recycle, restore and permanent deletion actions.
- Image-preview icon for supported formats.
- Contextual bulk-operation toolbar.

Administrator dashboard

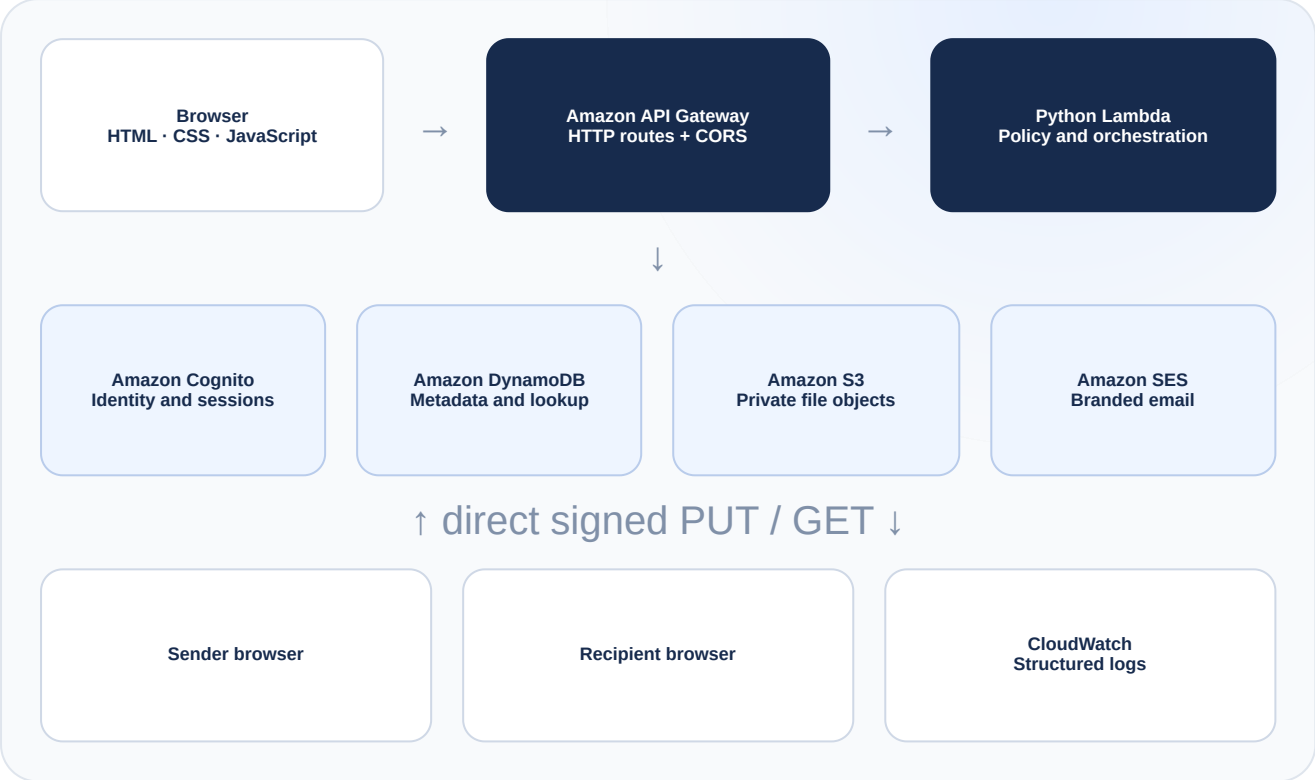
- Protected by immutable Cognito administrator identity.
- Searchable list of up to 100 Cognito users.
- Enable/disable access and revoke sessions.
- Send a password-reset code.
- Review all, active and recycled records for a selected user.
- Administrator entry point appears only after backend verification.

Readable implementation

The browser code intentionally favors named functions, explicit state and straightforward DOM manipulation. This reduces framework overhead and makes the project approachable for maintenance and customer-specific customization.

04 · SYSTEM ARCHITECTURE

Serverless control plane, direct data plane



Control plane

API Gateway and Lambda authenticate users, enforce limits, allocate identifiers, create signed URLs, update records and trigger email. File bytes do not normally travel through Lambda.

Data plane

The browser transfers file bodies directly to private object storage using short-lived signed requests. This reduces Lambda execution time, avoids payload limits and scales independently.

The design can later route free and Premium users to different S3-compatible storage providers while preserving the same metadata and public-link abstraction.

05 · UPLOAD WORKFLOW

From file selection to recipient delivery

1

Validate the transfer in the browser

JavaScript checks file count, individual size, recipient syntax and terms acceptance, then requests a fresh reCAPTCHA token.

2

Request upload preparation

The browser sends metadata—not file content—to **POST /upload-url**, optionally with a Cognito access token.

3

Create the private mapping

Lambda verifies reCAPTCHA and identity, generates a 12-character file ID, creates a random S3 key and stores a pending DynamoDB record.

4

Upload directly to object storage

The browser uses a 15-minute signed PUT URL. The original filename is metadata in DynamoDB, not part of the S3 object key.

5

Verify completion

The browser calls **POST /complete**. Lambda performs S3 HeadObject, verifies actual size and marks the record uploaded.

6

Deliver temporary links

Public IDs resolve through Lambda to five-minute signed GET URLs. SES can send each recipient a branded email containing the links.

7

Apply retention

Application checks enforce exact expiry while S3 Lifecycle and DynamoDB TTL perform asynchronous storage and metadata cleanup.

06 · LAMBDA BACKEND

One Python function, several trusted workflows

The Lambda entry point distinguishes Cognito lifecycle events from API Gateway requests and delegates each operation to a focused handler.

Transfer and recipient routes

- `POST /upload-url · validate + reserve`
- `POST /complete · verify S3 object`
- `POST /send-links · validate + SES`
- `GET /files/{file_id} · signed redirect`

User file-management routes

- `GET /my-files · owner GSI query`
- `GET /my-files/{id}/preview`
- `DELETE /my-files/{id} · recycle`
- `GET /recycle-bin`
- `POST /recycle-bin/{id}/restore`
- `DELETE /recycle-bin/{id} · permanent`

Administrator routes

- `GET /admin/check`
- `GET /admin/users`
- `GET /admin/user-files`
- `POST /admin/users/action`

Cognito triggers

- `PreSignUp_SignUp · reCAPTCHA`
- `PostConfirmation_ConfirmSignUp · admin email`

Authentication

Accepts API Gateway JWT claims or validates a Cognito access token with GetUser.

Authorization

Matches ownership for customer routes and immutable Cognito sub for administrator routes.

Observability

Emits structured security and lifecycle events without logging tokens or signed URLs.

Failure handling

The backend returns controlled JSON errors, performs compensating S3 cleanup when record updates fail, and keeps administrator-notification email failures from blocking account confirmation.

07 · DATA MODEL

Opaque links, private object keys, useful metadata

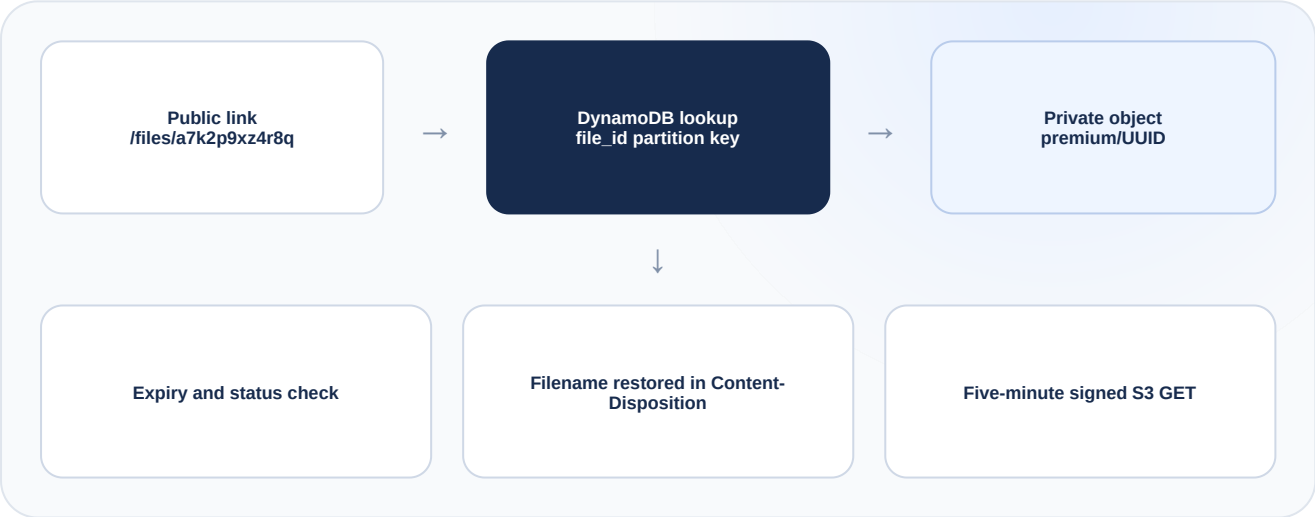
Representative DynamoDB record

```
file_id: a7k2p9xz4r8q
owner_id: Cognito sub
filename: proposal.pdf
s3_key: premium/<random UUID>
content_type: application/pdf
size_bytes: verified object size
uploaded_at / completed_at
expires_at / active_expires_epoch
retention: free | premium
status: pending | uploaded | recycled
ttl_epoch: asynchronous record expiry
```

Why the indirection matters

- The public 12-character ID is not an S3 object key.
- The private key contains no user-supplied filename.
- Original filenames remain in DynamoDB for display and downloads.
- Storage providers can change without changing public-link format.
- Owner ID enables dashboard queries through a global secondary index.
- State fields support pending upload, active file and Recycle Bin workflows.

Record-to-object relationship



Public IDs provide obscurity and convenience, but access control still depends on private storage, application validation and short-lived signed URLs. Link possession currently grants recipient download access.

08 · STORAGE LIFECYCLE

Retention designed around temporary delivery

Anonymous transfer

Upload
Random object beneath the free/ prefix.

Available
Public mapping remains usable for approximately 24 hours.

Expired
Lambda blocks access at the exact expiry timestamp.

Removed
S3 Lifecycle and DynamoDB TTL delete asynchronously.

Registered-user transfer

Upload
Random object beneath the premium/ prefix.

Active
Current configured lifetime is 30 days.

Archived
Lifecycle can transition content to a lower-cost class.

Managed
DynamoDB mapping remains for future restore and governance.

Recycle Bin

Recycle
Object is copied to recycle/ and its record becomes recycled.

Retain
Recoverable for up to 30 days.

Restore
Copied to a new premium/ key with refreshed dates.

Delete
Permanent action removes both S3 object and record.

Operational nuance

S3 Lifecycle and DynamoDB TTL are asynchronous. The application does not rely on physical deletion timing to enforce access: Lambda evaluates the stored expiry timestamp before creating a download URL.

09 · AWS SERVICES

Managed services and their responsibilities

AG

Amazon API Gateway

Publishes HTTP routes, handles CORS and can validate Cognito JWTs before invoking Lambda.

λ

AWS Lambda

Python control plane for signed URLs, validation, metadata, lifecycle actions, email and administration.

S3

Amazon S3

Private encrypted object storage, direct signed transfers and prefix-based lifecycle policies.

DB

Amazon DynamoDB

File mapping, owner index, status, expiry, retention and asynchronous TTL deletion.

ID

Amazon Cognito

User registration, email confirmation, sign-in, recovery, tokens and administrative account controls.

SES

Amazon SES

Branded recipient messages and optional administrator notification after confirmed signup.

CW

Amazon CloudWatch

Lambda logs, error investigation, storage metrics and potential alarms routed through SNS.

AMP

AWS Amplify Hosting

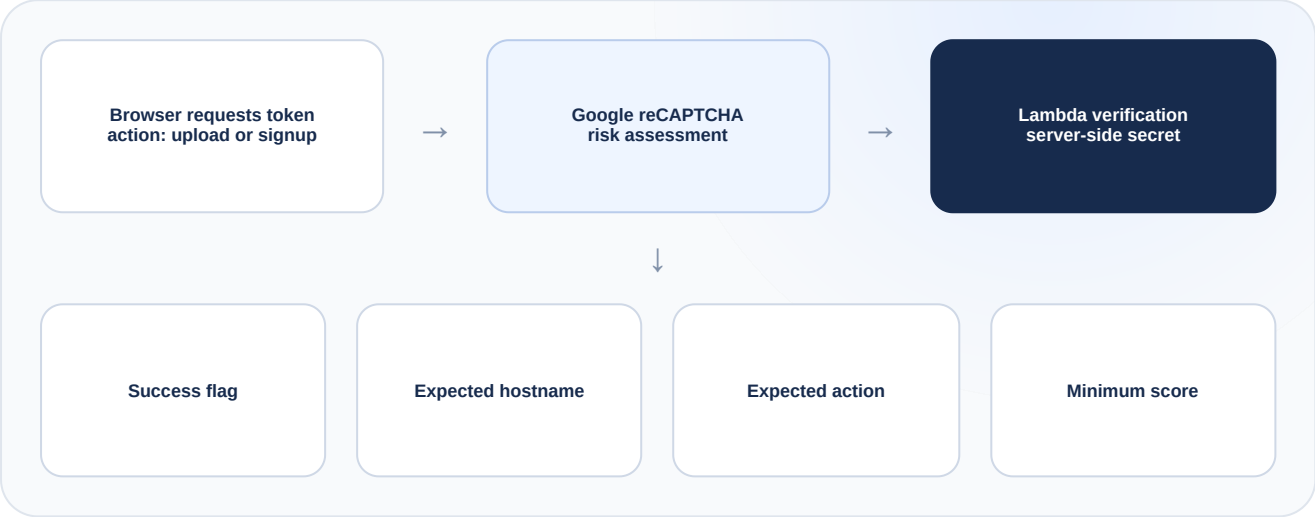
Proposed static frontend hosting with CDN delivery, SSL, Git-based builds and custom domains.

Cost profile

The static frontend is small and inexpensive to host. File storage and download bandwidth are expected to become the principal variable costs, followed by email, API and execution usage as traffic grows.

Invisible abuse scoring for uploads and signup

The product uses Google reCAPTCHA v3. Instead of asking every legitimate visitor to solve a visual challenge, the browser obtains a short-lived token associated with a named action.



Upload protection

The token is included with the upload-preparation request. Lambda refuses to generate signed upload URLs when validation fails.

Signup protection

The token travels in Cognito ClientMetadata and is verified by the Pre Sign-up Lambda trigger before the user is created.

Configuration boundary

The site key is public in browser configuration. The secret key, expected hostname and score threshold remain in Lambda environment variables.

Deployment detail

localhost, 127.0.0.1, the apex domain and www are different hostnames. Production should use one canonical hostname.

reCAPTCHA is one control, not a complete abuse strategy. Production hardening should also include API throttling, quotas, monitoring, sender reputation controls and potentially AWS WAF.

Layered controls for a public transfer service

Storage protection

- Private bucket with public access blocked.
- Random object keys without customer filenames.
- Short-lived signed PUT and GET URLs.
- Exact content-length verification after upload.
- Lifecycle rules by free, premium and recycle prefix.

Identity and authorization

- Cognito access tokens for authenticated operations.
- Owner ID comparison before customer file actions.
- Immutable Cognito sub check for administrator operations.
- Disable, enable and global sign-out through Cognito.
- JWT authorizers recommended on protected API routes.

Application safeguards

- File count, file size and recipient validation in browser and backend.
- Allowlisted preview MIME types.
- Uploaded SVG and HTML are not rendered as inline previews.
- Filename sanitization for Content-Disposition.
- Controlled errors avoid exposing storage keys and signed URLs.

Operational visibility

- Structured upload and administrator-action logs.
- Timestamp, file ID, owner ID and request context for investigations.
- Thirty-day log retention recommended by the current privacy notice.
- S3 BucketSizeBytes alarms can warn at 1 GiB, 10 GiB and higher levels.
- SES bounce, complaint and delivery monitoring remains production work.

Privacy posture

The application includes Terms and Privacy pages and documents limited security logging. Before commercial launch, counsel should review the controller identity, retention statements, lawful bases, processor disclosures, user rights process and liability language.

From prototype to commercial service

PRIORITY	FEATURE	WHY IT MATTERS
Near term	Subscription and entitlement service	Connect payment status to real Premium authorization instead of treating every authenticated user as Premium.
Near term	Quota enforcement	Enforce the planned maximum file count and 10 GB account storage in Lambda.
Near term	Multipart parallel upload	Improve throughput and retry behavior for larger Premium files.
Near term	Multi-provider storage routing	Use economical S3-compatible storage for free transfers and optimized AWS infrastructure for Premium.
Near term	Malware scanning	Quarantine suspicious objects before recipient access.
Product	Password-protected transfers	Add recipient-level access control beyond link possession.
Product	Custom expiration and download limits	Give senders stronger transfer governance.
Product	Notifications and statistics	Show views, downloads and delivery confirmation.
Product	Scheduled delivery and expiration reminders	Support professional campaign and delivery workflows.
Product	Expanded previews and gallery	Add PDF, audio and video previews plus thumbnail generation.
Business	Custom branding and domains	Enable agency, studio and white-label use cases.
Business	Teams, roles and audit history	Support multi-user customer accounts and enterprise oversight.

Performance

Measure before promising “faster”; combine provider routing, multipart concurrency and optional S3 Transfer Acceleration.

Reliability

Add idempotent upload reservations, incomplete-upload cleanup, counters and reconciliation jobs.

Governance

Formalize retention, account deletion, abuse response, backups, incident response and support procedures.

CUSTOMER SUMMARY

A credible foundation for secure file delivery

SimpleStorage.cloud combines a deliberately simple user interface with a serverless backend that keeps file bytes in object storage, metadata in DynamoDB and identity in Cognito. The result is a modular foundation that can be extended without rebuilding the complete product.

What is compelling today

- Clean, professional and responsive interface.
- Real direct-to-S3 transfer path.
- Opaque public links and verified completion.
- Branded recipient email.
- Authenticated dashboard and Recycle Bin.
- Protected administrator account management.
- Clear AWS-native operational model.

What completes the commercial story

- Payment and entitlement integration.
- Strict production quotas and rate limiting.
- Malware scanning and abuse operations.
- Download analytics and notifications.
- Multi-provider cost optimization.
- Automated deployment and environment separation.
- Final legal, privacy and security review.

SIMPLESTORAGE.CLOUD

Send files away.
No friction.

A concise customer experience, backed by a flexible serverless architecture.

This document is a product and technical overview, not a security certification, service-level agreement or legal warranty. Capabilities should be revalidated against the deployed production environment before contractual use.