CORE VAULT

Complete System Report, Installation Guide, and Operations Runbook

Product version	0.1.0
Prepared for	First Age Interactive
Production URL	https://vault.firstageinteractive.com
Report date	September 13, 2026
Classification	Private studio technical documentation

Document Control

This report describes the deployed Core Vault system as of version 0.1.0. It combines product documentation, installation instructions, daily workflows, administration, troubleshooting, and recovery guidance.

Item	Value
System owner	First Age Interactive
Application	Core Vault
Unity support	Unity 6 (6000.0 or newer)
Primary host	Relay9 VPS
VPS address	15.204.38.154
Public hostname	vault.firstageinteractive.com
Server installation	/opt/core-vault
Unity package ID	com.firstage.core-vault
Package source	https://vault.firstageinteractive.com/packages/core-vault.git#v0.1.0

Note: Do not place passwords, SSH private keys, database credentials, session tokens, or studio secrets in Unity projects or exported packages.

Contents

1. What Core Vault Does

Core Vault separates the studio's master asset library from active Unity projects. It gives developers a searchable web catalog and creates focused Unity packages containing selected files plus known dependencies.

System boundaries

• Core Vault is a separate product from Core Sync. Vault curates reusable source assets; Sync versions complete working Unity projects.

• The master library lives on the studio VPS and remains available to authorized users through the Studio Portal.

• A generated package is a delivery artifact. The original uploaded collection remains unchanged unless an authorized user edits or deletes it in Vault.

Main workflow

Acquire an asset pack, ingest it into Vault, inspect and categorize its contents, select only what the project needs, build a package, then import that package with Unity or the Core Vault Editor extension.

Important limitation

Vault can discover dependency GUIDs that exist inside an indexed collection. Always review a generated package in a test project when the source relies on custom render pipelines, packages, shaders, scripts, or external files.

2. Access and Security

Open the Studio Portal first. A valid 30-day studio session opens Core Vault without another password prompt.

Addresses

• Studio Portal: **https://studio.firstageinteractive.com**

• Core Vault: **https://vault.firstageinteractive.com**

• Use the **Studio Portal** button in Vault to return to the central developer hub.

Password management

Change the studio password from **Studio Portal - Account**. Passwords must contain 12 to 128 characters. The Portal is the single password-management location for the connected tools.

Session safety

• Do not share account passwords or browser session data.

• Sign out on shared machines. The normal session lasts up to 30 days.

• All public access should remain behind HTTPS. VPS storage and database backups should be restricted to studio administrators.

3. Library Structure

Vault uses folders, collections, and assets. This keeps large vendor packs manageable without mixing them into one flat directory.

Folders

Create a library folder to group related collections, such as Monster SFX, Environment Kits, or Character Packs. Set its name and classification information when created. Collections can be dragged into a folder or moved with the Move To action.

Collections

A collection represents one imported Unity package or uploaded asset folder. It stores name, category, publisher/author, source URL, tags, original source, indexed files, dependency relationships, and previews.

Assets

Individual assets include prefabs, models, materials, textures, animations, scenes, audio, video, code, and other files. Use search, category, tags, type, and motion filters to narrow the catalog.

Naming guidance

- Keep the vendor or pack name in the collection name.
- Use predictable categories such as Characters, Environments, Audio, UI, VFX, and Tools.
- Add practical tags: fantasy, undead, modular, URP, HDRP, humanoid, root-motion, or in-place.
- Store the Asset Store or publisher URL so licensing and updates can be traced.

4. Ingesting a Unity Package

Vault extracts a .unitypackage into its private master collection so the contents can be searched and curated. It does not install the package into an active Unity project.

Procedure

• Choose **+ Add Assets** and select **Unity Package**.

• Choose the .unitypackage file and enter a clear collection name.

• Optionally set category, publisher/author, source URL, and comma-separated tags.

• Start the upload. Keep the browser open until upload and indexing complete.

• Open the collection and verify the file count, paths, asset types, previews, and metadata.

Large uploads

Large files use resumable chunked upload sessions. The percentage may pause while the file count continues increasing during extraction or indexing. Do not restart solely because the displayed percentage is temporarily unchanged. Cancel only if the counters stop changing for an extended period or an error is shown.

Preserved data

Vault preserves the original uploaded package and stores extracted assets separately. Unity GUID and meta information are retained when present, allowing package reconstruction and dependency matching.

If previews are missing

A .unitypackage may contain Unity-generated preview PNGs. Raw folder uploads usually do not. Use the Core Vault Unity window's **Repair / Generate Prefab & Model Previews** command after the collection has been indexed and synced to a Unity project.

5. Uploading an Asset Folder

Folder upload is useful for loose vendor files or already-extracted content.

Procedure

• Choose **+ Add Assets**, then **Asset Folder**.

• Select the root folder. Browser folder selection includes the files beneath it.

• Enter collection details and begin upload.

• After indexing, inspect paths and verify that .meta files traveled with their matching assets.

GUID safety

For Unity content, upload each asset together with its .meta file. Missing meta files can cause new GUIDs and broken references when later imported.

When to use a package instead

Prefer the original .unitypackage when available because it is more likely to include preserved Unity metadata and embedded preview images. Use folder upload when the package is unavailable or when the source is already organized as a Unity Assets tree.

6. Searching, Previewing, and Reviewing

The web panel is the main place to inspect the master library before anything enters a production project.

Find assets

• Use the global search bar for collection names, publishers, tags, and indexed content.

• Use category and tag filters in the sidebar.

• Inside a collection, filter by asset type. Animation metadata can also be classified as In Place, Root Motion, or Unclassified.

Visual previews

Texture and image previews display directly. Prefab and model previews appear when the source package contains previews or after Unity generates and uploads them. A missing picture does not mean the source asset is missing.

Audio previews

Audio assets show a speaker control on their card and playback controls in asset details. Use these to audition sounds before adding them to a package. Browser codec support determines whether a particular source format can play directly.

Project locations

After a Unity project runs **Scan & Sync Project**, asset details can show whether the asset exists in that project, the project directory, Unity version, and saved scenes that reference it. An asset can be present without being referenced by a saved scene.

Animation note

Vault records animation files and motion classifications, but browser playback is not a substitute for Unity's rig and Animator evaluation. Validate final animation behavior inside Unity.

7. Building a Project Package

The Import Queue collects selected assets. Vault expands the selection to include indexed dependencies and exports a standard .unitypackage.

Build steps

• Open an asset or collection and add the required items to the Import Queue.

• Use the collection-level action when every asset in that collection is needed.

• Review the queue. Remove anything that should not enter the target project.

• Choose **Build Unity Package**, provide a useful package name, and wait for completion.

• Download the generated .unitypackage from package history or import it through the Unity extension.

Dependency closure

Vault follows known GUID links recursively. Selecting a prefab can therefore add its materials, textures, models, animation clips, and other indexed dependencies. Dependencies in another unindexed package cannot be inferred.

Package hygiene

• Build small purpose-specific packages instead of one enormous export.

• Use a test project to check shader compatibility, missing scripts, render-pipeline requirements, and import warnings.

• Do not modify vendor originals in Vault merely to solve a project-specific issue; make the project override inside the target project.

8. Unity Editor Extension

The Core Vault package adds **Window - Core Vault** to Unity 6. It downloads completed packages and reports project inventory to the web catalog.

Install by Git URL

In Unity, open **Window - Package Manager**, choose **+ - Install package from git URL**, and paste:

```
https://vault.firstageinteractive.com/packages/core-vault.git#v0.1.0
```

Install from ZIP

Download the ZIP from the Studio Portal, extract it outside the project, then use Package Manager's **Install package from disk** and select package.json. The Git method is preferred for clean upgrades.

Connect

• Open **Window - Core Vault**.

• Sign in with the studio account. The Editor session is remembered for 30 days.

• Use **Open Web Vault** to create and review packages in the browser.

• Use **Refresh Ready Packages** to load completed exports.

Import

Choose the ready package in the Editor window and download/import it. Unity's standard import dialog remains the final checkpoint before files are written to Assets.

Scan & Sync Project

This inventories the current Unity project and sends asset GUID/path data, project name, path, Unity version, and saved-scene references to Vault. It does not upload the entire project or connect Vault to Core Sync.

Generate previews

Use the preview repair command for indexed prefabs and models that lack thumbnails. Unity loads the assets, renders supported previews, and uploads them to the matching Vault GUID records.

9. Moving, Renaming, Deleting, and Recovery

Organization operations affect the master library, so confirm scope before destructive actions.

Move and rename

Drag a collection onto a folder or use Move To. Rename folders and collections through their management menu. Moving organizational records does not rewrite internal asset paths.

Trash

Deleting an asset or collection normally moves it to Trash first. Restore it when the removal was accidental. Permanently removing an item deletes its stored Vault copy and may affect future package builds.

Folder deletion

Deleting a folder deletes or trashes the collections contained inside it, according to the confirmation shown. Read the scope carefully; this is intentionally different from merely moving collections out of the folder.

Duplicates

Duplicate scanning should compare file content, but related texture maps are not duplicates simply because they share similar dimensions or names. Review each duplicate group before removing a copy.

10. Recommended Studio Workflow

Use Vault as the controlled intake gate between purchased/source assets and production Unity projects.

Asset intake

• Record publisher, store link, license context, render pipeline, Unity compatibility, and useful tags.

• Ingest the untouched source package or folder.

• Verify the collection before deleting local download copies.

• Keep any license files and documentation inside the collection when permitted.

Project selection

• Search Vault before importing a new vendor package directly into a game.

• Select only needed prefabs, materials, textures, models, audio, scripts, and dependencies.

• Build a named package for the task or feature.

• Test import, then submit the resulting project changes through Core Sync.

Ownership

Vault manages reusable assets. Core manages design documentation. Core Sync manages Unity project history. The Studio Portal provides the shared entry point and account password.

11. Troubleshooting

Most problems fall into upload state, missing metadata, preview generation, dependencies, or authentication.

Upload restarts or appears stuck

• Check whether the file number or byte counter is still increasing.

• Keep the tab open and avoid starting the same upload repeatedly.

• If the upload genuinely stops, cancel once and retry from a stable wired connection or from a machine with sufficient free temporary storage.

• For repeated failure, record the collection name, source size, time, and visible error before checking VPS logs.

No prefab/model image

• Confirm the collection has completed indexing.

• Open the same assets in the Core Vault Unity project.

• Run Repair / Generate Prefab & Model Previews.

• Refresh the web collection after preview upload completes.

Missing or broken dependencies

• Confirm source .meta files were included.

• Check that referenced assets exist in the indexed collection.

• Test the package in Unity and note missing GUIDs or scripts.

• Ingest the missing companion package or select the required asset manually.

Cannot sign in

Start from the Studio Portal. If necessary, use Portal - Account to change the password. Confirm the browser accepts secure firstageinteractive.com cookies and that the VPS health endpoint is online.

Package not visible in Unity

Build the package in the web Import Queue first, then click Refresh Ready Packages in Unity. Ensure the Editor extension is signed in to the same studio environment.

12. Administration and Recovery

The VPS is the authoritative Vault service. Protect both structured metadata and stored binary content.

What must be backed up

• PostgreSQL database: accounts, collections, assets, dependencies, tags, folders, queues, export history, and project inventory.

• Vault storage volumes: original uploads, extracted collections, previews, generated exports, trash, and staging where applicable.

• Application configuration: compose file, environment settings, reverse proxy configuration, and installer Git repository.

Recovery order

• Restore database and storage from the same recovery point.

• Restore configuration and secrets without exposing them in documentation or source control.

• Start database, Vault application, and proxy services.

• Verify health, sign-in, collection counts, previews, downloads, and one test package build.

Capacity planning

Master packages, extracted assets, previews, exports, and backups can temporarily multiply storage requirements. Monitor available VPS disk space before ingesting multi-gigabyte libraries. Remove obsolete generated exports according to studio policy, but retain originals and verified backups.

Security checklist

• Use HTTPS only.

• Restrict SSH and database access.

• Keep the studio SSO secret and database credentials outside source control.

• Review active studio accounts when team membership changes.

• Test a small restore after significant infrastructure changes.

Quick Reference

Task	Where
Change studio password	Studio Portal - Account
Ingest package/folder	Core Vault Web - + Add Assets
Build .unitypackage	Core Vault Web - Import Queue
Download/import export	Unity - Window - Core Vault
Report project usage	Unity - Core Vault - Scan & Sync Project
Return to developer hub	Studio Portal button in each tool

Installer Git URL

```
https://vault.firstageinteractive.com/packages/core-vault.git#v0.1.0
```

This guide documents Core Vault 0.1.0 as deployed for First Age Interactive. Update the guide when workflows, retention rules, installer versions, or access policies change.