NASA HUNCH Data Repository: Project Requirements, File Organization, Access Rules

---

1. Project Requirements

1.1. General Overview

The NASA HUNCH Data Repository will serve as a centralized, secure, and scalable platform for collecting, organizing, and managing student project submissions across multiple programs and years. The repository must support periodic updates (PDR, CDR, FDR), handle large media files, and enforce strict access controls to protect project confidentiality while enabling oversight by NASA HUNCH staff and educators.

1.1.1 Special Requirement - Operational

Different from most of our projects, we plan to have this project working 4 weeks after the CDR.

This project will be the model for the NASA HUNCH Data Repository, and we will use this to start collecting our Project Data:

1- This means we must try to use existing systems.
2- Minimize any code or customer being written.
3- Minimize cost.

Measure of success should also include ease of use, ease of Maintenance, and easy training for the operators (Provide a manual and videos for training.

NASA HUNCH Data Repository

Table of Contents

1.2. Functional Requirements

1.2.1. Submission Workflow

- **Three Submission Phases:**
  - **PDR (Preliminary Design Review)**
  - **CDR (Critical Design Review)**
  - **FDR (Final Design Review)**

- **Submission Content by Phase:**
  - **PDR:**
    - 60%+ Working Prototype (Pictures)
    - Presentation Board and Team
    - Engineering Notebook
    - Client/Project Storyboards
    - Website or Google Site
    - 3-Minute Prototype Video
    - 3-Minute Presentation Video
    - etc.

  - **CDR:**
    - 90%+ Working Prototype (Pictures)
    - Presentation Board
    - Engineering Notebook (D.A.A.N.)
    - Improved Website or Google Site
    - Client/Project Storyboards
    - New 3-Minute Prototype Video
    - New 3-Minute Presentation Video
    - CDR Video Testimonial(s)
    - Marketing Brochure (Front/Back)
    - Technical Presentation (Charts/Slides)
    - Etc.

- **FDR:**
  - 100%+ Working Prototype
  - Presentation Board
  - Engineering Notebook
  - Improved Website or Google Site
  - FDR Marketing Brochure (Front/Back)
  - FDR Technical Presentation (Charts/Slides)
  - FDR New 3-Minute Prototype Video
  - FDR New 3-Minute Presentation Video
  - All Code Files (organized by system/sub-system)
  - All Executable Files
  - FDR New Video Testimonial(s)
  - USB with all the above, code, executables, and platform scenarios
  - etc.

1.2.2. File Organization

- **Hierarchical Structure:**
  - Top-level: Year
  - Second-level: Program (e.g., BIO Health, Software, Design-Prototype, etc.)
  - Third-level: Project (unique per team)
  - Fourth-level: Submission Phase (PDR, CDR, FDR)
  - Fifth-level: Submission Content (as per phase requirements)

- **Support for 5–15 projects per program per year**
- **Support up to 150 Project Teams Folders per Year (Note: All Folders and Files need to start with the Team Number, 9 Alphanumeric Characters)**
- **Ability to upload individual files or folders (batch upload)**
- **Support for large media files (videos, images) using chunking, resumable uploads, and/or cloud storage solutions.**

1.2.3. Access Control

- **NASA HUNCH Staff:**
  - Full access to all programs, projects, and files across all years
- **Project Teams:**
  - Access only to their project folder(s) and submission content
  - Ability to upload/update files for their project
- **Teachers (per school/project):**
  - Access to all projects submitted by their own school/team
  - Ability to review and download files
- **No cross-project visibility for student teams**
- **Role-based access control (RBAC) with resource hierarchy**

1.2.4. Usability and Collaboration

- **User-friendly interface for uploading, organizing, and retrieving files**
- **Metadata and tagging for searchability (e.g., project name, year, program, phase, file type)**
- **Version control for file updates and tracking changes**
- **Automated notifications for submission deadlines and updates**
- **Great user experience**
- **Documentation /README for files in each project folder**
- **Lables**

1.2.5. Security and Compliance

- **Secure authentication and authorization**
- **Data encryption at rest and in transit**
- **Audit logs for file access and changes**
- **Compliance with NASA and educational data privacy standards**

1.2.6. Scalability and Maintenance

- **Scalable storage to accommodate growing data volume**
- **Regular data curation and quality control**
- **Archival and backup procedures**

---

2. File Organization Diagram (Character Diagram)

Below is a textual representation of the hierarchical file organization and access rules. (A visual diagram can be created using tools like Lucidchart or Visio as per .)

```
NASA_HUNCH_Repository/
│
├── 2025/
│   ├── BIO_Health/
│   │   ├── Project_Alpha/
│   │   │   ├── PDR/
│   │   │   │   ├── Prototype_Pictures/
│   │   │   │   ├── Presentation_Board/
│   │   │   │   ├── Engineering_Notebook/
│   │   │   │   ├── Storyboards/
│   │   │   │   ├── Website/
│   │   │   │   ├── Prototype_Video.mp4
│   │   │   │   └── Presentation_Video.mp4
│   │   │   │   |----- ETC.
│   │   │   ├── CDR/
│   │   │   │   └── ...
│   │   │   └── FDR/
│   │   │       └── ...
│   │   ├── Project_Beta/
│   │   │   └── ...
│   │   ├── Software/
│   │   │   └── ...
│   │   └── Design_Prototype/
│   │       └── ...
├── 2026/
│   └── ...
```

NASA HUNCH Data Repository

Access Rules (Character Diagram):

- **NASA HUNCH Staff:**
  - Access: All folders and files (read/write)
- **Teacher (School X):**
  - Access: All projects under their school/team (read)
- **Project Team (Team Y):**
  - Access: Only their own project folder (read/write)
- **Other Project Teams:**
  - No access to other teams' folders

Legend:

- 🔒 = Restricted (team/teacher only)
- 🔓 = Open to NASA HUNCH staff

---

3. Access Control Model

- **Role-Based Access Control (RBAC) with Resource Hierarchy:**
  - Roles: NASA Staff, Teacher, Project Team Member
  - Resources: Year > Program > Project > Phase > Files
  - Permissions:
    - NASA Staff: All resources (read/write)
    - Teacher: All projects for their school/team (read)
    - Project Team: Their project only (read/write)
- **Inheritance:**
  - Permissions assigned at the project level propagate to all subfolders/files within that project.

---

4. Best Practices for File Management

- **Consistent naming conventions** for folders and files (e.g., [ProjectName]*[Phase]*[ContentType]_[Date])
- **Metadata tagging** for search and retrieval
- **Version control** for iterative submissions
- **Chunked and resumable uploads** for large files (especially videos)
- **Cloud storage integration** for scalability and reliability
- **Regular data curation and archiving**

---

5. Role-Based Access Control (RBAC) Support

Role-Based Access Control (RBAC) is **a security mechanism that restricts system access based on a user's role within an organization**. Instead of assigning permissions directly to individual users, RBAC assigns permissions to roles, and then users are assigned to those roles. This approach simplifies access management, enhances security, and improves efficiency, especially in large organizations. [1, 2, 3, 4]

Here's a more detailed explanation:

- **Roles:** Roles represent different job functions or levels of responsibility within an organization (e.g., administrator, manager, employee). [5, 6]
- **Permissions:** Permissions define the specific actions a user can perform on a system, such as read, write, or delete. [5]
- **Assignment:** Users are assigned to roles based on their job responsibilities, and they inherit the permissions associated with that role. [5, 6]
- **Benefits:** RBAC offers several advantages, including:
    - **Simplified Access Management:** Managing permissions becomes easier as changes are made at the role level, affecting all users assigned to that role. [2, 4]
    - **Enhanced Security:** Restricting access based on roles minimizes the risk of unauthorized access to sensitive information. [1, 4]
    - **Improved Efficiency:** RBAC streamlines the process of granting and revoking access, saving time and effort. [2, 4]
    - **Compliance:** RBAC helps organizations meet regulatory requirements by ensuring that users only have access to the resources they need. [3, 4]

[1] https://www.imperva.com/learn/data-security/role-based-access-control-rbac/
[2] https://auth0.com/docs/manage-users/access-control/rbac
[3] https://www.strongdm.com/rbac
[4] https://www.splashtop.com/blog/role-based-access-control
[5] https://workos.com/blog/what-is-rbac-how-it-works-and-when-to-use-it
[6] https://delinea.com/what-is/role-based-access-control-rbac

6. Visualizing File Organization and Access Rules

Recommendation:
Use a hierarchical tree diagram (e.g., in Lucidchart or Visio) to represent the folder structure, with color-coded nodes for access levels (e.g., blue for NASA staff, green for teachers, yellow for project teams). Edges can represent inheritance of permissions. This approach aligns with best practices for data classification diagrams.

---

7. Platform and Tool Recommendations

- **Cloud-based repository with RBAC support** (e.g., Open Science Framework, Harvard Dataverse, or a custom solution with AWS S3 and IAM policies)
- **Support for large file uploads** (chunking, resumable uploads, CDN integration)
- **User-friendly web interface with batch upload and metadata tagging**
- **Automated notifications and version control**

---

8. Provide Feedback from Teacher and NASA Staff - Program Leader or Regional Mentor

With all this information provided by the student, there is a big opportunity to provide feedback at different stages, especially at the PDR and maybe at the CDR, with the goal that either the teacher or the NASA hunt staff member, or the Regional Mentor can provide feedback to help the student. grow in their projects.

9. Summary

By implementing a hierarchical, role-based, and scalable data repository with robust access controls, NASA HUNCH can efficiently manage annual student project submissions, ensure data security, and facilitate collaboration among staff, teachers, and students. The outlined requirements, file organization, access rules, and storyboard scripts provide a comprehensive blueprint for system design and user experience.