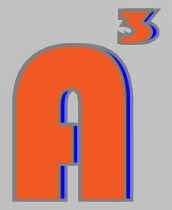


framework

Aggregation and Archiving of Artifacts

"A Repository of Reliable Resources for Academia"

PROTOTYPE PRESENTATION

CS 411, SUMMER 2020

TEAM CRYSTAL

TEAM CRYSTAL

Team Crystal, comprised of computer science students at Old Dominion University, is developing the A^3 framework.

Aaron Berman

Team Lead/Webmaster

Joshua Murphy

Documentation Manager/ Database Developer

Mike Campbell

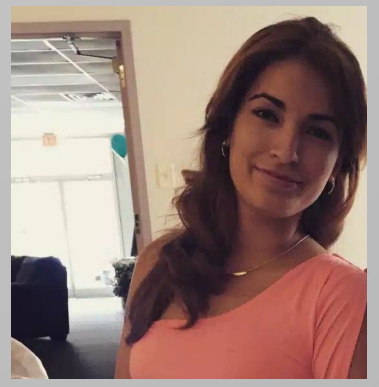
Database Architect/Testing

Stephen Ayers

Algorithms Developer/Testing

Noah Jennings

Algorithms Developer-UI/UX Developer



Rosalie Oliva

UI/UX Developer

TABLE OF CONTENTS

PROBLEM STATEMENT

Educators and students lack a framework to aggregate and archive fragmented and domain-specific artifacts for the purpose of academic knowledge management.

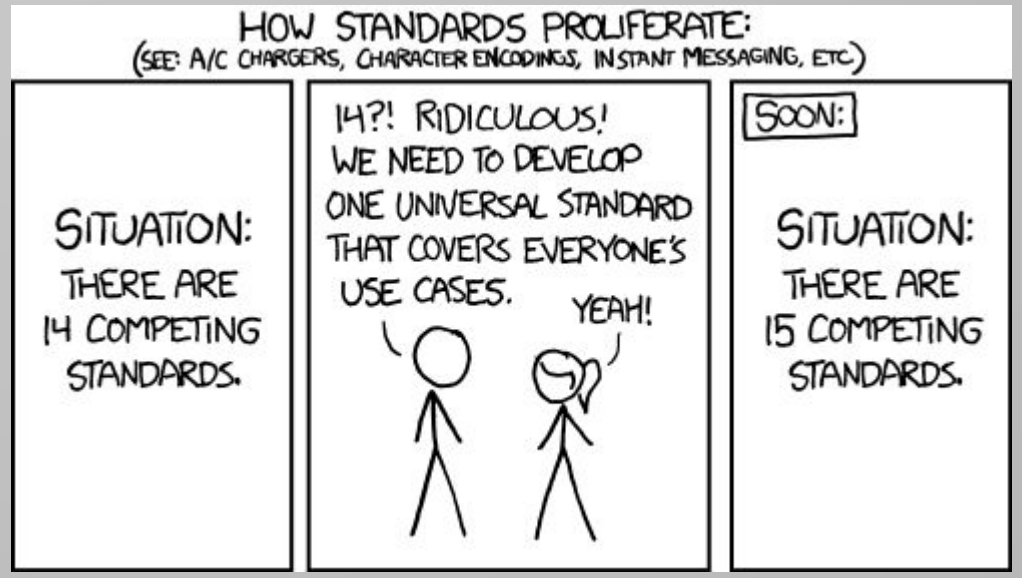


Image Credit:
(XKCD.com 2020)

DEFINITIONS:

Aggregate: Data that is composed of smaller pieces that form a larger whole.

Artifact: Refers to a file or document.

Knowledge Assets: The accumulated intellectual resources of an organization codified in electronic form.

Repository: Central location in which data is stored and managed.

TRADITIONAL SHORTCOMINGS:

Knowledge Repositories

- Formal artifact aggregation in traditional academic environments does not exist [3]
- The aggregation that does currently exist does not support tracking of changes through time [2]
- Current aggregation is not strong enough to be considered centralized

Knowledge Accessibility

- Knowledge is isolated by specialization [16]
- Access is often restricted by course or major [16]
- Artifact format preference by instructor can vary wildly and may not be functional to others [17]

TRADITIONAL SHORTCOMINGS:

Knowledge Asset Management

- Instructor artifacts are often created in a variety of formats
- Individual instructors must often be petitioned for information [17]
- Loss of artifacts from reassignment of responsibilities [16]
- ODU CS department syllabus collection once took two months [17]

Knowledge Environment Enhancement

- Artifacts are specific to each course
- ODU instructors use a variety of platforms (e.g., Bb, PLE, CoWeM) to host artifacts [17]
- Special needs and distance learning lack proper support [3]
- Shared artifacts can benefit organizations on a fundamental level [16]

SOLUTION STATEMENT

A^3 framework is a framework for aggregating and archiving artifacts for educators, researchers, and students. A^3 framework seeks to overcome the challenges of individualization, location, and formatting in academic knowledge management by keeping information available, normalized, and centralized while being enhanced by a robust user interface.

SOLUTION CHARACTERISTICS:

Creating Formal Knowledge Repositories

- Create a robust infrastructure for artifacts
- Support version history of artifacts for organization of knowledge assets
- Centralize information concisely and effectively

Improve Knowledge Accessibility

- Create knowledge artifacts that are widely applicable
- Create cross course accessibility
- Normalize artifacts from varied platforms (e.g., Bb, PLE, CoWeM) into translatable formats*

*Not implemented in Prototype

SOLUTION CHARACTERISTICS:

Knowledge Asset Management

- Unify formatting among instructors through normalization
- Remove necessity of individual knowledge asset request
- Create systematic storage of vital course information
- Automate collection of standard reference materials*

Knowledge Environment Enhancement

- Translate artifacts to a universally applicable format
- Core organizational improvements through a cooperative environment
- Normalization to allow functionality across artifact formats
- Special needs and distance learning accessibility*

*Not implemented in Prototype

CUSTOMERS AND END USERS:

Old Dominion University Computer Science Department:

Real World Product:

- Guest
- Student
- Faculty
- Administrator
- Tester

Prototype:

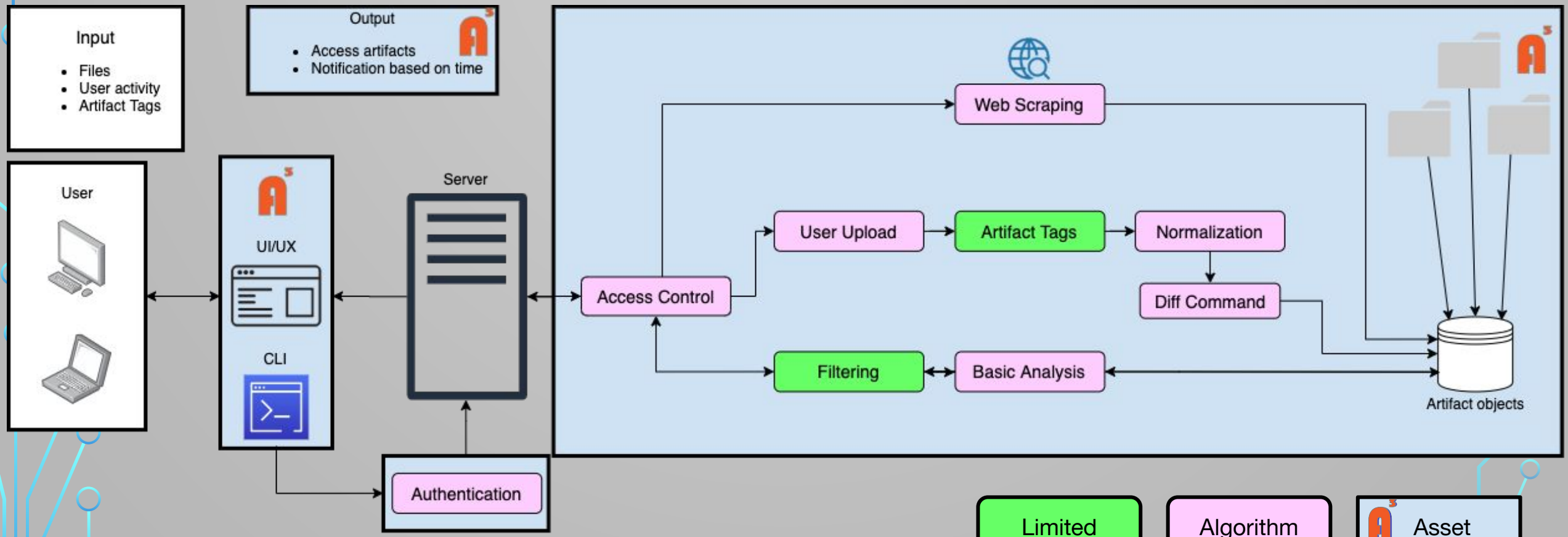
- Guest
- Faculty
- Tester

Image Credit: (Freepik.com 2020)

RWP vs. Prototype

Feature/Capabilities Comparison Chart			
Feature/Capability	**Real World**	**A³ Prototype**	**A³ Prototype Actual**
Database Storage	X	X	X
Graphical User Interface	X	Limited	Mock-up
Command Line Interface	X	X	Limited
User Authentication	X	Limited	Limited
Access Control	X	X	X
Artifact Upload	X	X	X
Repository Creation	X	X	X
Artifact Normalization	X	X	Limited
Artifact Comparison	X	X	X
Artifact Update	X	X	X
Artifact/Repo Deletion	X		
Web Scraping	X	Limited	Limited
Artifact Change Record	X	X	X
Artifact Exporting	X	X	X
Artifact/Repo Tags	X	Limited	Limited
Artifact/Repo Searching	X	Limited	Limited
Artifact Contributor List	X		
Artifact/Repo Sharing	X		

MAJOR FUNCTIONAL COMPONENTS



DEVELOPMENT TOOLS

Software Requirements:

- Language: Python 3.8 or newer
- Python framework: Flask, mysql.connector, and pypandoc
- GUI language: HTML, CSS, and JS
- JS frameworks: React and Node (with npm)
- Documentation: pydoc and Sphinx
- Configuration management: tox or Conda
- Analysis: pycodestyle (formerly PEP 8) and Pylint
- APIs: BeautifulSoup 4, requests, pandoc

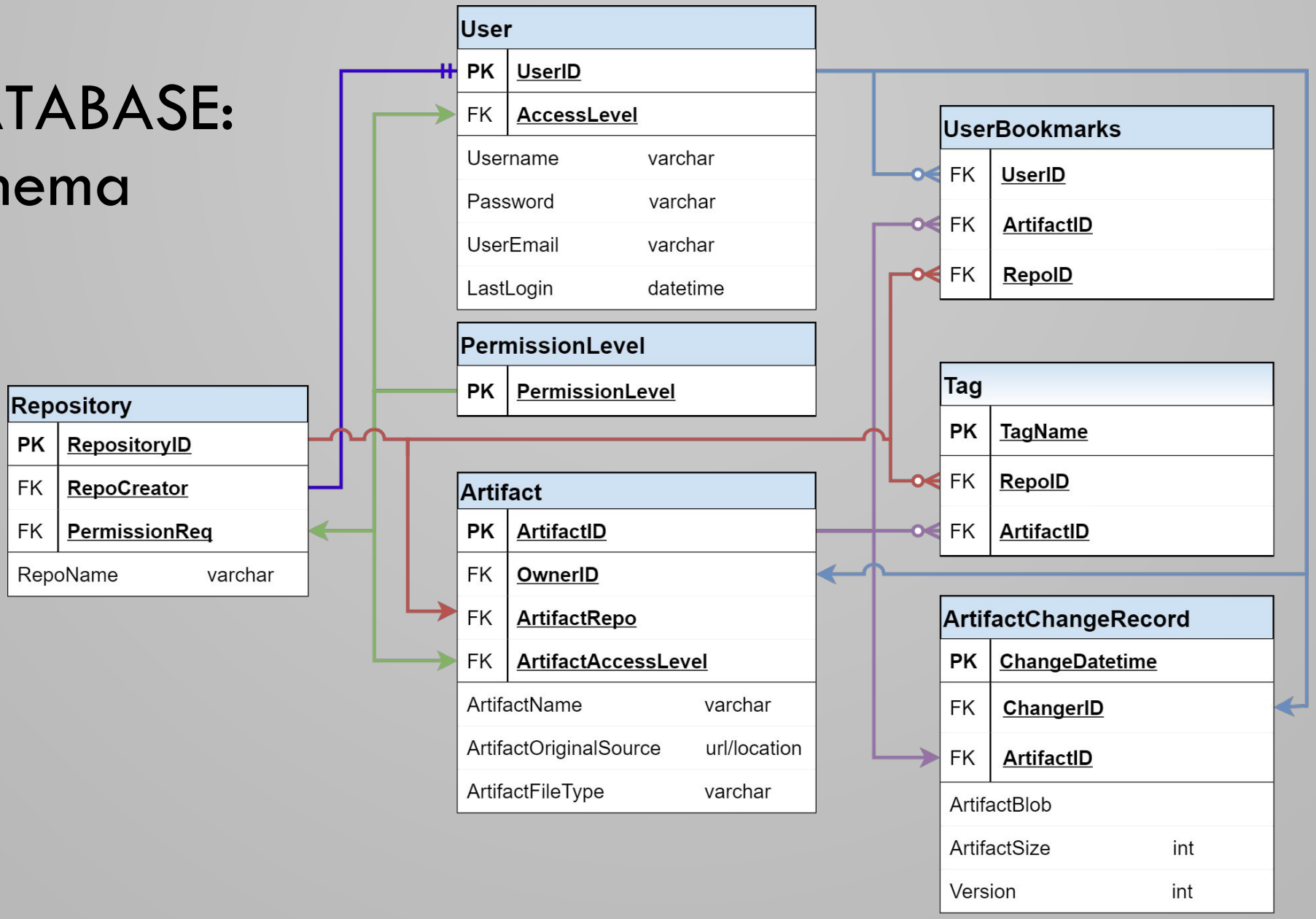
DEVELOPMENT TOOLS

Technology Requirements:

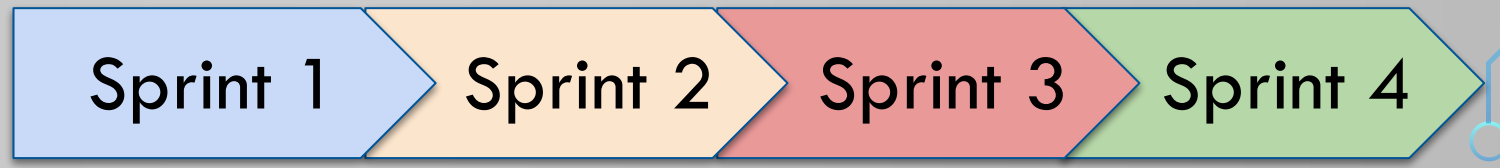
- Code Repository with Version Control: GitLab
- Issue Tracking and Development Scheduling: GitLab
- Containerization: Docker and Docker Compose
- Database: MySQL
- IDE: Visual Studio Code (VS Code)

Hardware Requirements: Single VM instance running an Ubuntu distribution on the ODU CS server

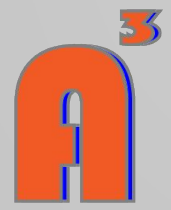
DATABASE: Schema



AGILE SPRINTS



- Sprint 1 (July 5, 2020)
  - Database framework
  - CLI interface
  - Simple comparison function
  - Authentication/Role-based access
  - Testing
- Sprint 2 (July 12, 2020)
  - Full database implementation
  - Normalization function(s)
  - Analysis function(s)
  - GUI framework
  - Testing
- Sprint 3 (July 20, 2020)
  - Diff function (line-by-line)
  - Web scraping implementation
  - Final GUI implementation
  - Limited artifact tags and filtering
  - Testing
- Sprint 4 - if time permits (July 27, 2020)
  - Full implementation of artifact tags and filtering
  - Notifications
  - Testing



framework

A repository of reliable resources.
Our goal is simplicity.

REFERENCES

1. *Blackboard Archive Extractor*. (2016, December 15) cs.odu.edu. Retrieved March 10, 2020, from https://www.cs.odu.edu/~cpi/old/411/crystals17/.
2. Carroll, J., Choo, C. W., Dunlap, D., Isenhour, P., Kerr, S., MacLean, A., & Rosson, M. (2003). Knowledge Management Support for Teachers. *Educational Technology Research and Development*, *51*(4), 42-64. www.jstor.org/stable/30221184
3. Davenport, T., Long, M. & Beers, M.. (1997). *Building Successful Knowledge Management Projects* [Working Paper]. Retrieved March 8, 2020, from https://www.researchgate.net/publication/200045855_Building_Successful_Knowledge_Management_Projects.
4. *Document Management Software | eFileCabinet*. (2020). eFileCabinet. Retrieved February 20, 2020, from https://www.efilecabinet.com.
5. Domes, S. (2017). *Progressive Web Apps with React: Create lightning fast web apps with native power using React and Firebase*. Packt Publishing Ltd.

REFERENCES cont.

7. *File Sharing and Sync For Education, Schools and Universities - FileCloud*. (2020). FileCloud. Retrieved February 20, 2020, from https://www.getfilecloud.com/file-sharing-and-sync-for-education/.
8. *GitHub Features: The right tools for the job*. (2020). GitHub. Retrieved March 10,il 2020, from https://github.com/features#team-management.
9. Kennedy, T. (2020, January 21). *Home · Wiki · Thomas J. Kennedy / cs-roars-proposal*. GitLab. Retrieved 26 April 2020, from https://git-community.cs.odu.edu/tkennedy/cs-roars-proposal/-/wikis/home.
10. Nvlpubs.nist.gov. (n.d.). *Glossary of Key Information Security Terms*. From https://nvlpubs.nist.gov/nistpubs/ir/2013/NIST.IR.7298r2.pdf.
11. MacFarlane, J. (2006). *Pandoc - About pandoc*. Pandoc.org. From https://pandoc.org/index.html.
12. Tsapps.nist.gov. (2020). *Data Loss Prevention*. From https://tsapps.nist.gov/publication/get_pdf.cfm?pub_id=904672.
13. Xie, I., & Matusiak, K. K. (2016, July 29). Digital preservation. *Science Direct* (255-279). Retrieved March 10, 2020, from https://www.sciencedirect.com/science/article/pii/B9780124171121000090

REFERENCES cont.

14. Zeil, S. (2019, December 26). *Building the Website*. cs.odu.edu. Retrieved 26 April 2020, from https://www.cs.odu.edu/~zeil/cowem/Public/buildingTheWebsite/index.html.
15. Zeil, S. (2020, January 21). *zeil / CoWeM - Course Websites from Markdown*. GitLab. From https://git-community.cs.odu.edu/zeil/Course_Website_Management.
16. Brunelle J., personal communication, March 2, 2020.
17. Kennedy T. J., personal communication, February 12, 2020.

APPENDIX

USER STORIES: Guest

I **must** be able to :

- choose to login with credentials
- export a public artifact
- view a list of public artifacts
- access the database with both a CLI and limited GUI.
- access the database outside of the network

I **wish** to be able to :

- be notified of what type artifacts I can access (i.e. public/private)
- filter and sort public artifacts that belong to the database

I **must not** be able to :

- access private artifacts
- edit or update artifacts
- access user accounts
- modify or update public artifacts
- change a user's access level
- upload an artifact

USER STORIES: Faculty

I **must** be able to :

- do anything a guest can do
- login with credentials
- upload artifacts
- edit and update artifacts I own
- set access level requirement for artifacts I own
- upload an artifact via web scraper
- normalize files on upload
- view a Diff report on command
- view a list of all private artifacts

I **wish** to be able to :

- access the source of any artifact
- see usage reports about artifacts I own
- tag artifacts with keywords describing their content
- create notifications for myself regarding regular updates to artifacts

I **must not** be able to :

- access, edit, or update artifacts I don't own
- delete user accounts
- change a user's access level

USER STORIES: Tester

I **must** be able to :

- access artifacts within my testing parameters
- view reports about users and artifacts
- have administrative capabilities within my testing parameters
- analyze results
- report test problems and anomalies

- have all the capabilities of any other account type
- manage the database including:
  - change attributes of any artifact, such as access level requirement
  - remove artifacts
  - create and change user accounts

I **wish** to be able to :

- improve software quality
- capture user requirements

I **must not** be able to :

- test their own products
- make all the decisions and changes to assure a better quality
- edit database schema