BIOL492 Directed Research Writeup

Stephen Szwiec

May 11, 2021

Contents

Abstract

The creation of programmed and scripted tools for biological data extraction and analysis remains an ongoing effort in the field of bioinformatics. Delivering high-throughput analysis for plant breeding programs is a unique subset of of this, given agronomic improvement program usage of this data to make critical workflow decisions. To facilitate such a program, the curation and navigation of databases and related tools is a continual process, with heterogeneous information networks, including high-performance computing (HPC), used in developing new crop germplasm. In Summer 2020, I worked with Bandillo Lab of North Dakota State University as it launched an effort to document processes, to engineer systems, and to improve use of HPC and machine learning resources.

1 System Description

At NDSU Bandillo Lab, research is conducted using pea (*Pisum sativum*), chickpea (*Cicer arietinum*), and lentil (*Lens culinaris*) to develop these pulse crops into cultivars suited to the *terrior* of the northern plains region, and to improve the agronomically valuable phenotypes for these crop plants, including yield and protein content. The program is currently an 11 year-workflow, with each phase of testing performed in parallel across several test sites spanning the climate and soil basis of North Dakota [4]. In this process, the lab harnesses the genetic biodiversity of the germplasm collections of affiliated organizations as starter genetic material [25]. These accession lines, which consist of previously developed cultivars, divided by market class [28] [40], are used to create an initial parental P cross between these two recombinant inbred lines in a randomized block with repetition in a greenhouse environment. A subsequent cross of the *filial* F_1 generation repeats this randomized block breeding to generate variance in the population. Agronomically viable phenotype data is taken from both in the field and from quantitative chemical analysis of seeds yielded [29]. From generation $F_2 - F_5$, phenotypic selection of nursery plants funnels the top-performing blocks from the first trials into a single plant selection per block. From generation $F_6 - F_9$, observations on crop output yield factors and resistances to both biotic and abiotic stressors are taken, and phenotype data is analyzed using analysis of variance (ANOVA) [13] and best linear unbiased predictors (BLUPs) [45] for the phenotype data set taken. Finally, at the $F_{10} - F_{11}$ generation, statewide testing confirming the phenotype values of the germplasm and full sequencing complete the agronomic selection process.

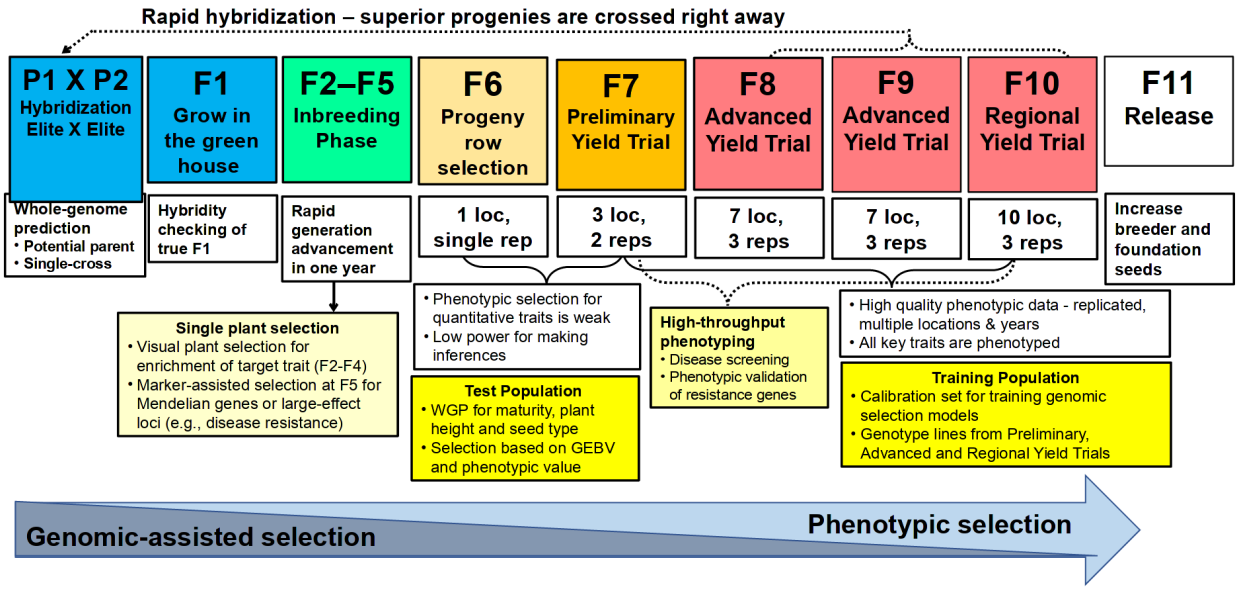

Figure 1: **Agronomic Selection Workflow** this diagram shows a simplified process workflow diagram for the lab, with agronomic selection occurring over 11 generations of crosses, and trials steps taken. Image used with permission, originally by Nanoy Bandillo (2020).

This program performs with a large data overhead, and consists of a heterogeneous network of information technologies with various deployment configurations. In May 2020, Bandillo Lab had phenotype data stored for the last 15 years, relating over 200 accession lines for pea, chickpea, and lentil, related to documenting the agronomic selection process. No networked or database system was in use for the management of genotype data, despite the core importance of this data to the project. The system was largely the legacy of a previous principal investigator at NDSU for pulse agronomics, and a cybersecurity attack in 2019 had encrypted notes from the previous team's process documentation.

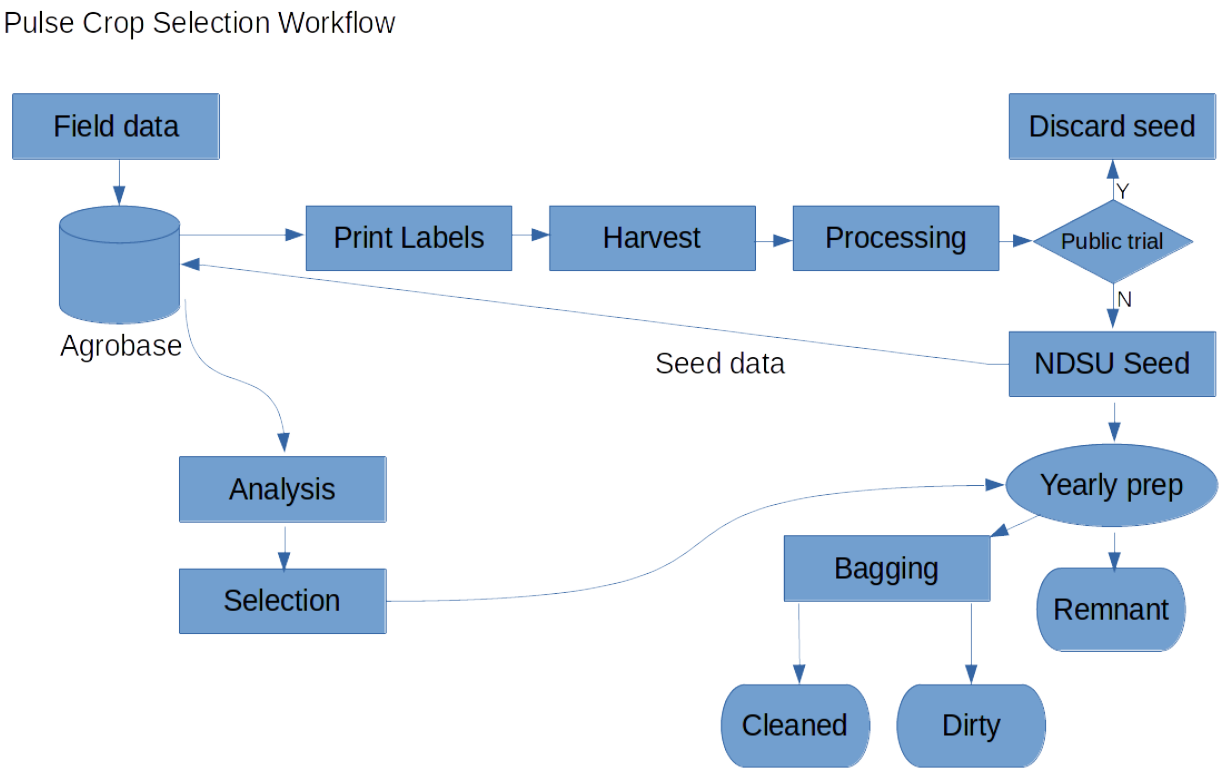

Figure 2: **Field Trial Processing Workflow** this diagram shows a simplified process workflow diagram for field work per experiment, in which field trial organisms are analyzed, their seeds inventoried and processed, and the internal usage of database programs to facilitate this.

2 Agrobase Audit

2.1 Introduction

In May 2020, the Bandillo Lab at NDSU performed an audit of the stored phenotype data. Agrobase®(Agronomix Inc., Winnipeg, Canada) is a closed-source phenotype database system offering experiment management and statistical analysis for agronomic data, and is used at NDSU across all of the agronomics and plant sciences programs. Primarily a GUI front-end to Microsoft SQL server, the Agrobase software provides a managed schema and templating system, and allows for data saved to have ANOVA performed in an automated fashion. This software has a prominent role at NDSU's plant science research, and falls under the purview of the Breeding Pipeline Database Management team; however, data of each breeding science team is individually managed by that same team.

2.2 Data Analysis

The database schema of Agrobase was studied to better understand the internal operations of the database system and to learn how to identify the outlying and anomalous data in Agrobase. For this task, a Windows 10 virtual machine was spun up as a test environment, into which both Agrobase software and Microsoft SQL Server were installed.

A full database dump of the Agrobase server was performed using a Microsoft SQL API call made to the Agrobase server, where a local copy of the Agrobase database was created using the same schema using Microsoft PowerShell. This reverse-engineered schema was then studied to understand the distribution of data within the database system, and a visual representation of this schema was generated to better study information flow through the system as a reverse engineering of the processes undertaken by the program.

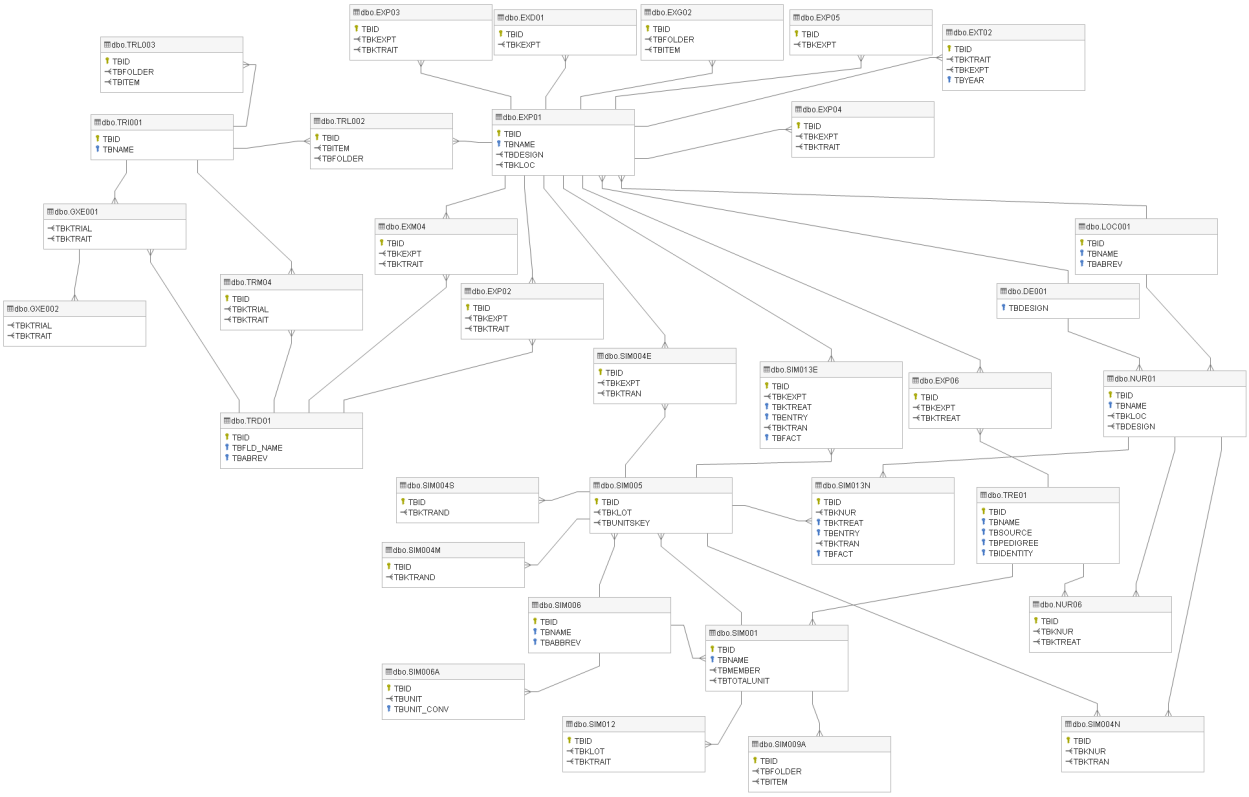

Figure 3: **Agrobase database schema** this diagram shows the highly linked nature of database tables within Agrobase, where treatments, experiments, experimental design, names, and raw data related to one agronomic line entry are spread across several tables.

After study of this database schema to understand how then information was being saved across the system, a series of scripts were written to identify abandoned experiments, erroneous data entry, and experimental outliers. These scripts were developed referencing internal NDSU Research Extension Center document "Data Collection and Reporting Pro-

tocols, Revision 9 (2019)" (authored by Hannah Worral) which identified agronomic traits for the experimental datasets and typical reported data per trait. These scripts were run per agronomic trait for both pea, chickpea, and lentil data sets. Example SQL code of the scripts created are included in **Appendix A**, and these tools were used to extract information from the database in an efficient manner using a connection to the server within the virtual machine.

2.3 Findings and Actions Taken

Study of the Agrobase database schema showed that raw experimental data (including numeric data) was saved into Agrobase as character strings, and then presented to the user as a particular datatype, the definitions for which were saved in another table. This allowed for erroneous data to accumulate, as no datatype checking was applied as an input validation, which prevented non-numeric data input when not using the GUI interface.

The audit of the Agrobase pulse crop database for abandoned experiments found that 58 chickpea experiments, 109 lentil experiments, and 244 pea experiments had no relevant agronomic data whatsoever, and could be removed from the database and archived for recording purposes to save system resources.

Outlier identification audit script found a number of outliers for agronomic trait information, but also found erroneous entries (for instance, pea test weight with negative numeric values.) All outliers were compiled into a report for review by Bandillo Lab team members, with a report of lines which had reported yield and protein outliers four standard deviations above the mean compiled into a separate report for rapid agronomic improvement of those lines.

Furthermore, steps were also taken to rectify erroneous data in Agrobase, either specifying that the particular data was instead lost or removing experiments for which no experimental data was found. All activities were logged and this log was presented to the Breeding Pipeline Database Management (BPDM) team [38]. These corrections allowed not only for improved database performance in the case of archiving abandoned experiments, but also allowed for new and subsequent experimental ANOVA analyses to better reflect their true experimental findings by removing erroneous data from the set.

3 Database Nomenclature Revision

3.1 Introduction

After conclusion of the Agrobase audit, audit findings concluded that further action would need to be taken, as experiment nomenclature stored in the database had shifted over time into a number of styles. To allow for easier development in the future, a new harmonized nomenclature for pea, chickpea, and lentil was created.

3.2 Methodology

Working within the Bandillo Lab team, and in coordination with the BPDM team, interviews were conducted with plant breeders to understand the schema of information hierarchy used within the program, and what information was relevant to the program. It was determined that the nomenclature for the Pulse program for use with Agrobase and other database tools would need to be a standardized schema for interoperability with a number of APIs. The schema would need to be nominally human-readable and machine-searchable by use of regular expressions, such that the code would not overlap and always uniquely identify data and research assets across the program. Furthermore, this system would need to cover the breadth of experimental trials performed, and be accepting of retained unknown experimental data from the legacy system. After revision, experiments in process were recorded using this system. Full text of nomenclature developed from these methods available as **Appendix B**.

3.3 Future Work

While the nomenclature has been revised moving forward, future work includes scripted renaming of experiments in the database, and creating an index of these changes to harmonize these changes across all data in the program.

4 Genome-Wide Agronomic Selection - parallelization for HPC clusters

4.1 Introduction

Genomic selection (GS) uses genome wide markers to predict breeding values [37] rather than considering the discrete molecular biology components individually, a more inferential approach to identifying the causes of variance between populations can be performed by taking the imputed matrix of only the single nucleotide polymorphism (SNP) matrix of the differences between genotypes. This method is underpinned by the understanding of genetic gain over a given time, known as the Breeder's Equation [33], given as:

$$R = \frac{ir\sigma_A}{y}$$

Where R is gain in a quantifiable trait over time, with i as selection intensity parameter, r as accuracy value of selection, with additive standard deviation of genetic variances σ_A, for a given y years. Having better prediction models, and performing earlier selection through genotype rather than by phenotype, allow agronomic researchers to maximize genetic gain on polyphenic traits.

By performing training using machine learning models with Bayesian inference on a genome-wide association study (GWAS) [50], qualitative prediction of a phenotype from any given genotype can be prepared given some smaller training population. Such a model would be system in the form:

$$Y_{ijk} = \mu + G_i + T_j + R_{k(j)} + e_{ijk}$$

Where the observed quantitative phenotype Y_{ijk} value is given by the system for the training data phenotype mean μ, genotypic effect matrix G_i, random trial effect T_j, for $R_{k(j)}$ replication effects nested within a trial, with some residual error e_{ijk}. This model would also produce a heritability H^2 estimate [50], in the form:

$$H^2 = \frac{\sigma_G^2}{\sigma_G^2 + \sigma}$$

Taken together, these predictions, heritability estimates, and residual error provide for actionable summary statistic data for agronomic improvement programs. Additional predictability of models can be achieved with the application of subpopulation structure as a weighted variable within the prediction models [3].

During the Summer of 2020, data analysis work on both genome-wide association study (GWAS) and on population substructure for a set of 482 accessions of field pea was performed in preparation of the recently-published article "Harnessing genetic diversity in the USDA pea (Pisum sativum L.) germplasm collection through genomic prediction." [5]. Data and findings from this publication have been republished with permission from its authors. The following provides more detail on the data science approaches used in preparation of this publication, and presents the same summary statistics as findings.

4.2 GWAS Data Analysis

Data analysis was performed as a scripted head-to-head training of machine learning models, with creation tasks on the NDSU CCAST Thunder2 super-computing cluster, using the R language [49] and running on Red Hat®Enterprise Linux®release 7.6 (Linux kernel 3.10.0), and utilized free and open source libraries, including The GNU coreutils [20], tassel5 [9], and OpenPBS [2].

Pea genotype data for accessions was received in variant call format [35], and a script **Appendix C.1** was used to generate a numeric genotype SNP matrix. This genotype matrix was then used with phenotype data from experiments of the same accession to train the rrBLUP [16], GAUSS [7], PLSR [8], ELNET [24], RF [32], and AVE [44] models. heritability and accuracy tests were also performed. The script **Appendix C.2** also uses parallelization of these model training loops across a pool of compute cluster nodes, made available through the PBS script **Appendix C.3**. This parallelization was performed with doParallel in R [53] and utilized 24 CPU cores (Intel ®Xeon ®Gold 6140 @ 3.7Ghz) and 5GB of memory on the CCAST Thunder2 cluster for computation, requiring 11 minutes to process through the data.

4.3 Pea Population Structure

Data analysis of pea population structure was performed using ADMIXTURE [14] using a 5-fold Quick Newton/Block validation for log likelihood of $k = 1 - 10$ subpopulations. Processing proceeded on NDSU CCAST Thunder2 cluster with 70 threads and 16GB of memory, and processing took 6 hours to complete.

4.4 Results

4.4.1 Phenotypic heritability and correlation

The heritability for plant height was 0.81, with an average height of 74cm. Pods per plant had a heritability estimate of 0.50 with a mean of 18 pods per plant and ranged from 15 to 23 pods per plant. Days to maturity had a mean of 104 days with an estimated heritability of 0.51. Seed yield per hectare ranged widely from 1734 to 4463 kg/ha with a mean yield of 2918 kg/ha and a heritability value of 0.67. The number of pods per plant was highly and positively correlated with seed yield. Correlation estimation also suggested seed yield was positively correlated with plant height (PH), days to maturity (DM), days to first flowering (DFF). For summary statistics, see **Appendix C.4**.

4.4.2 Comparison of predictive ability of different genomic selection models

No single model consistently performed best across all traits that we evaluated. Predictive abilities from different models performed similarly, although slight differences were observed, likely due to existing trait complexity. For DFF, the highest predictive ability was obtained from PLSR and ELNET (0.57). The genomic selection model using rrBLUP resulted in the highest predictive ability for the number of seeds per pod (0.30), days to maturity (0.46), and seed yield (0.35). Gaussian kernel exhibited the highest predictive ability for plant height (0.52). The highest predictive ability for pods per plant was obtained through ELNET (0.27). For full detail, see **Appendix C.5**.

4.4.3 Determining optimal marker density

To evaluate the increasing number of SNPs, we used the genomic prediction model with the highest predictive ability for a particular trait. For example, as PLSR provided the highest predictive ability for days to first flowering, we used this model to predict days to first flowering. In general, predictive ability increased with an increasing number of markers. The highest reported predictive ability was for the number of seeds per pod (0.30) at 30K markers. Pods per plant, plant height, and harvest date obtained the highest predictive ability when all 31K markers were utilized. We got prediction accuracy for seed yield at 20K markers (0.41) compared to the rest marker density.

4.4.4 Population structure and prediction

Population structure analysis with ADMIXTURE yielded a strong inflection point at $k = 7$, giving high likelihood of there being 7-8 ancestral subpopulations within the USDA pea

germplasm. Based on >60% ancestry, each accession was classified into eight subpopulations ($k = 8$). Using ADMIXTURE, we obtained eight subpopulations.

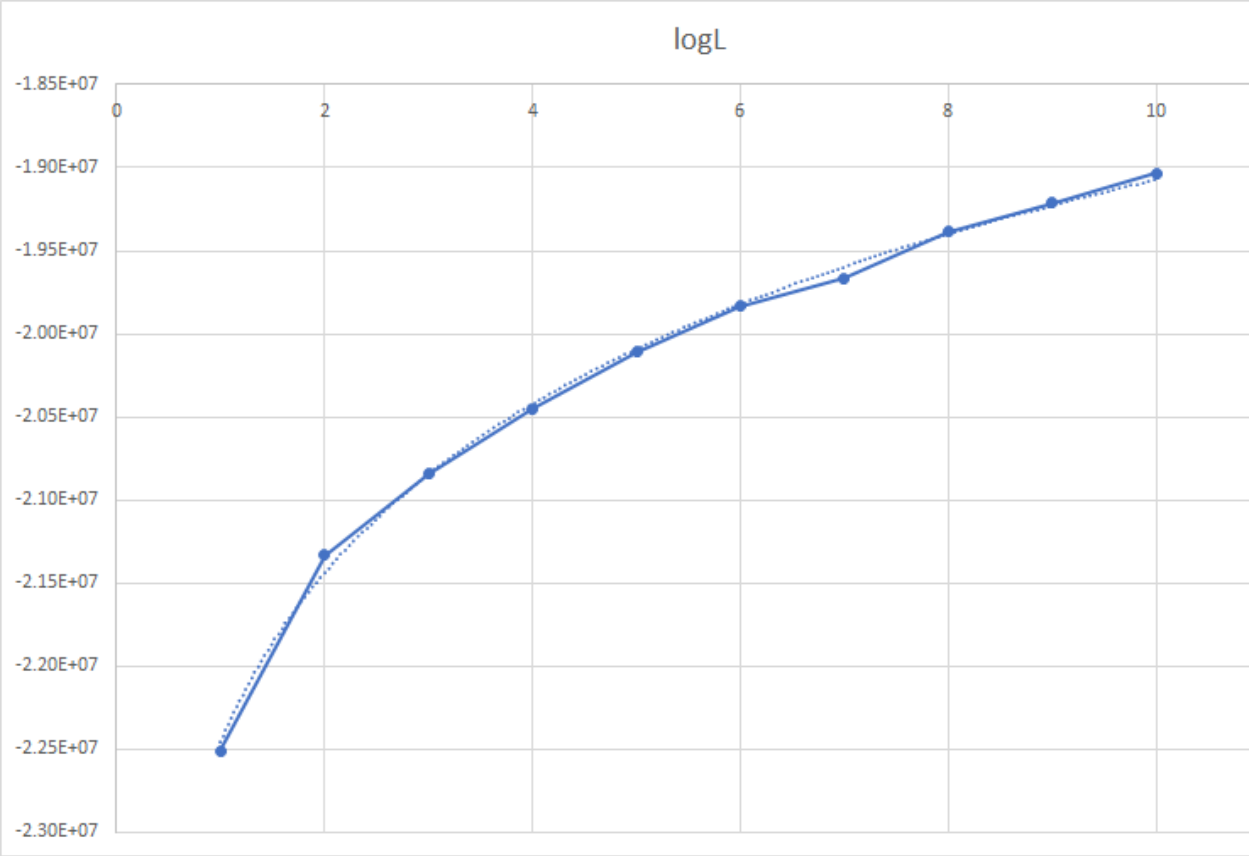

Figure 4: **ADMIXTURE log likelihood** Results from ADMIXTURE analysis of pea accession SNPs using 5-fold QN/Block verification, X-axis showing number of subpopulation clusters, and Y-axis showing log likelihood convergence point of the algorithm.

Population structure explained some portion of the phenotypic variance, ranging from 9-19%, with the highest percentages observed for plant height (19%) and seed yield (17%). Using either ADMIXTURE or PCA to account for the effect due to population structure, we improved the predictive ability. We observed a 6% improvement for days to first flowering and 32% for seed yield compared with models that did not account for population structure.

We also performed within-subpopulation predictions. Presented here are the predictive abilities for subpopulations 5, 7, and 8, as they had at least 40 entries. Subpopulation 8 had the highest predictive ability for days to first flowering (0.68), plant height (0.33), days to maturity (0.43), and seed yield (0.37). The highest predictive abilities for the number of seeds per pod (0.40) and pods per plant (0.12) were obtained from subpopulation 7. Notably, predictive ability was generally higher when all subpopulations were run in the model compared to when predictions were made within subpopulations.

5 Seed Inventory System Design

5.1 Introduction

The adoption of the Internet as a platform for digital transformation within organizations had led to a greater sophistication of organization processes, allowing for real-time data sharing for internal and external users. The creation of integrated data services, in the form of various database-enabled applications, has allowed for a greater speed of processes, for a decreased cost of services provided, and for a greater capacity for quantitative decision-making through data analytics. In this paper, a web application and related organizational processes are described for the modernization of a seed inventory system, providing for a lean supermarket system with double-entry accounting. The case studied to arrive at this system is the NDSU Pulse Crops Breeding Program at the North Research Extension Center. This document will describe the current state of the seed inventory system, detail the changes made, provide a description of the database schema to be used, and finally describe the technological tools used to achieve these changes.

August 2020, Bandillo Lab performed a transformative change to the processing of internal seed inventories. An ergonomic and usability study of the current system was performed and a kaizen process revision was performed, with a full proposal for a new seed inventory arising from this revision.

5.2 Description of System

At pilot site NDSU Minot North Central Research Extension Center (NCREC), seeds are divided by line and by experiment, and these seeds are kept in both envelopes and bags for the coming years. Bagged seeds are kept in a warehouse area: 300 inches in length and 196 inches in width (16.33ft x 25ft, or 4.97m x 7.62m), and these bags are kept in 42gallon (158L) plastic bins. Seeds in envelopes are otherwise stored in plastic trays in an adjacent room of the same warehouse. The database for this inventory system is kept on Agrobase, although the seed inventory module for Agrobase is not currently loaded. If a researcher would like to find a specific seed within the current system, that researcher would look at their computer, write down or print output from Agrobase, then find their seed based on this output.

5.3 Kaizen Audit Findings

The physical inventory system is both spatially wasteful, with seeds stacked into plastic bins, and is also a labor intensive and ergonomically brutal system to maintain, with the size of bins not allowing for quick repositioning and causing repetitive motion strain (stooping, lunging motions needed to find seed in bins). Furthermore, Agrobase tends toward being counter-intuitive for seed inventory: it is difficult to differentiate clean and dirty seed, years, experiment runs, etc. from seed inventory. This compounds to making tracking and auditing where specific seed comes from very difficult, and has no easy way to increment/decrement seeds from system. The system was overall found to be a blocking factor, wasting the time and effort of the people working with the system, and could easily be improved in this regard.

Finally, the current system is not conducive for informed decision making about the seeds in inventory. This system should be replaced to not only eliminate ergonomic and usability bottlenecks, but also to provide a system which is informative, economic, and auditable.

5.4 Changes Made

The smallest immediate change was to implement a supermarket system inspired by the work of Taiichi Ohno [42] for all seeds otherwise kept in bins in the primary storage area to reduce the ergonomic load of the site. A physical organizational schema was devised using a floor plan of the area, and implemented as follows.

Furthermore, a proposal for a future system was put into place. This includes a feasibility study of available open-source platforms for development.

5.5 Required Components

1. A wireless internet connected computer with monitor, such as a laptop, capable of running a modern browser and a free USB port (RasPi+monitor+battery, Chromebook, netbooks, etc.)
2. A wheeled trolley cart for moving inventory, with the top platform having space reserved for the computer system
3. USB connected plug-and-play bar-code scanner
4. Shelving units with bar-coded labels
5. Seed bags or envelopes with bar-coded labels

5.6 Physical Organization

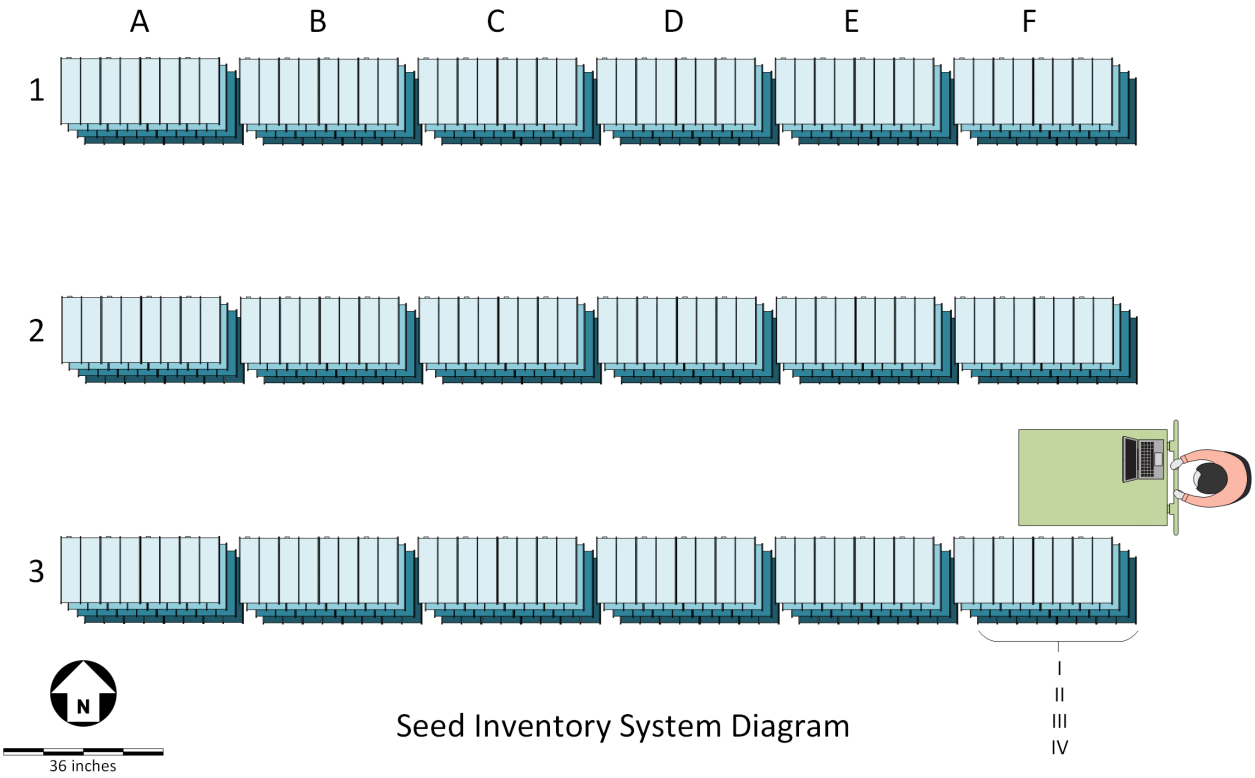

Figure 5: **Seed Inventory Layout** this diagram shows the implemented inventory layout at Minot NCREC pilot location. Figure shows 4 tiered shelving units with orientation reflecting the ordering schema of the overall system, and is built with the ergonomics of agricultural workers in mind. After successful launch with pilot, this system was re-implemented across other research sites.

The first component of the seed inventory system is to have a consistent physical organization structure, which is defined as the following:

1. All seeds are to be kept on shelving units, such that no bag or envelope-tray is kept higher than 60inches (152cm) to prevent ergonomic strain, workplace accidents, and wasted effort retrieving equipment.

2. All shelving units are to be physically organized in the following manner:

   (a) Shelving units are to be kept in rows and columns when possible, such that columns align East to West, and rows align North to South, and such that the aisles between shelving units also run East to West.

   (b) Shelving units are to be placed such that the aisle between shelving units has

adequate spacing for people to move through the aisle abreast of each other. At least 36inches (91cm) are to be allotted to each aisle.

(c) Shelf labeling schema

i. each column of shelving will receive a letter, **A-Z** onward to **AA-ZZ** if necessary.

ii. each row of shelving will receive a number, onward from **1**.

iii. each shelf, top to bottom, will receive a roman numeral, onward from **I**.

3. Shelving which cannot be arranged in rows and columns for safety or usability purposes should nonetheless use a similar grid arrangement to labeling schema: if shelves line the perimeter of a room, then each shelf can be considered one column, with each shelving unit a singleton member of its row: **A1, A2, A3**, etc.

4. Each shelf is to have affixed one bar-code per working side of the shelf, and this label also gives a human readable full name of that shelf in Column-Row-Shelf format. For example, the unit of the third column, second row, and forth shelf would be labeled **C2-IV**.

5. All seeds should be labeled, with primary defining characteristics printed on a label, including a machine readable bar-code with the seed name on both sides.

5.7 Proposed Future System

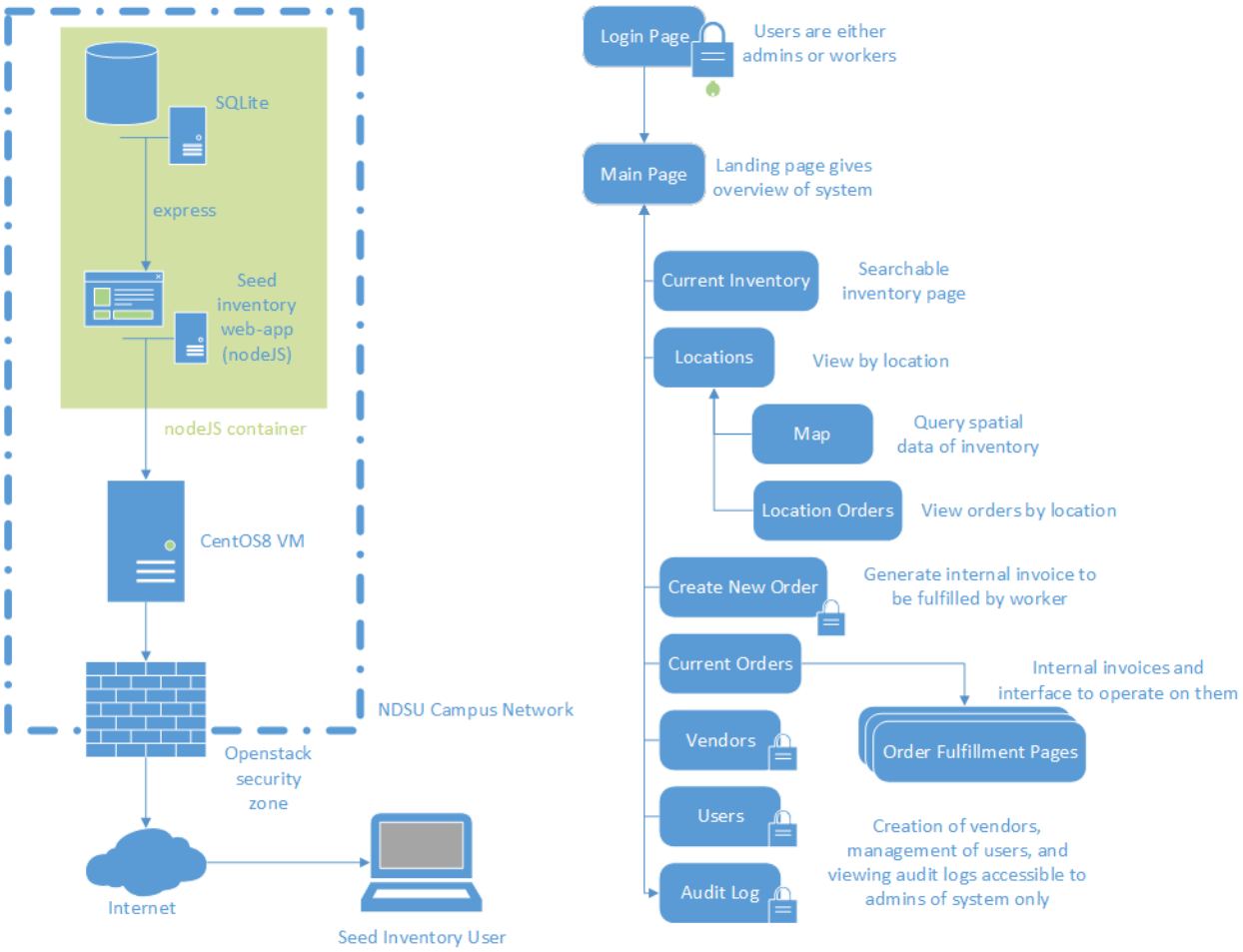

Figure 6: **Seed Inventory Layout** on left: network infrastructure diagram showing the seed inventory system alone within a larger NDSU-managed IT environment communicating to web interface user through the Internet. On right: seed inventory user presentation of web application pages, with locks showing administrative pages otherwise not shown to workers fulfilling orders.

In addition to the physical organization, the seed inventory is to be enabled through a client-server database program, which is hosted on a web-server and made available to users through a browser. This program will be constructed using available FOSS, and will itself be free and open source.
The following are a list of enabling FOSS technologies to be used to create the software component of this system:

5.7.1 GNU Affero General Public License version 3

Developed by the Free Software Foundation (FSF) and Dr Richard Stallman, the GNU General Public License (GPL) is a freely-applicable software license which allows the user of that software to run, study, share, and modify the software for any purpose. Rather than retaining strict copyright, it allows for authorship of the software creator to be retained while distributing in a free and open source manner, in a system known as 'copyleft'. More specifically, the GNU Affero General Public License version 3 (AGPL) provides a legal framework for the source code of a website to be offered over a network, allowing for the larger academic and programming communities to benefit from the inclusion of this software. The full text of the AGPLv3 is available at the GNU webpage [19].

5.7.2 CentOS 8 GNU/Linux

The CentOS Project is a community-driven, free software project based around the GNU operating system and Linux kernel and is available to anyone on the CentOS webpage [46]. This software distribution has several strengths, including receiving its upstream software updates from Red Hat Enterprise Linux, a wide number of available software packages, and community support.

5.7.3 Docker

Docker is a platform as a service (PaaS) system to deliver apps as containerized, lightweight, virtual systems. Utilizing the Linux container subsystem, Docker applications can be built and shipped as a container, such that the internal state of the app does not effect other running applications in the same environment, and the internal state of the app is fully described in code. This allows for a higher degree of security control than running services on bare-metal, allows for apps to scale for load balancing, and allows for predictable behavior of apps on all platforms. More information about Docker is available on the Docker webpage. [47]

5.7.4 SQLite

SQLite is a C-based SQL engine which provides for speed and high-reliability. This community-based project has created a system fully implementing all ACID properties and has provided its source code to the public domain. This system was chosen over similar systems such as PostgreSQL or MySQL due to its high speed, low overhead, and long-term support for the project. More information about SQLite is available on the project website. [30]

5.7.5 NodeJS

NodeJS, created by the OpenJS Foundation, is a server which listens for HTTP(S) requests and serves pages, much like any other web-server. Unlike Apache and nginx, NodeJS is fully built in JavaScript, and allows for client-side and server-side scripting to communicate seamlessly by using JS as a framework. This type of environment, called 'full stack JavaScript' allows for the creation of web-2.0 style web applications with low development overhead.

This is made possible through 'npm', the package manager of NodeJS, which is at this time the largest software package repository in the world [10]. These packages can be used in any configuration to develop web applications using these frameworks. More information about NodeJS and npm are available on the project webpage. [23]

5.7.6 Express

Express is a minimal and flexible web application framework, available in npm, which allows for the creation of Application Programming Interfaces (APIs). In other words, Express acts as a middleware between client-side events which happen in the user's browser, and the server-side handling of data, such as SQLite. Using Express, the pages of the site are saved as API calls, which cause different HTTP responses to be sent dynamically. As such, pages are built based on the information gathered from the database, and thus in sync without further human intervention, allowing the webpage to be a front-end to database operations. Full documentation and explanation of Express is available on their project webpage. [21]

5.7.7 Angular

Based on previous work by Google on the npm package AngularJS, Angular is a FOSS web framework which can be used to create user interfaces which adapt to both mobile and desktop browsers. This system is modular and allows for the creation of websites which are as dynamic and responsive as desktop-based applications. Furthermore, Angular has been redeveloped from the predecessor AngularJS to eliminate security flaws, and has been developed with security as a primary concern. Documentation is freely available on their project webpage. [27]

5.7.8 Bootstrap

Bootstrap is the most popular HTML, CSS, and JS library currently in use, originally developed by Twitter as a framework to standardize user interface design. As a JavaScript framework, it works with Angular to provide prebuilt design elements, themes, and icons out of the box, allowing coders to focus on implementation rather than visual design. The Bootstrap project and examples are made available on their project webpage. [52]

5.7.9 ngx-admin

A free and open source website administration panel, ngx-admin is another user interface web framework selected for this project. It implements a familiar panel design layout with features useful to database operations, including form validation, chart generation, the creation of data tables, and authentication. These features combined with the open and extensible nature of the framework, will save on development costs: rather than reinventing the wheel, only the functional features of the code will have to be implemented using prebuilt design elements. A demo of the various design elements provided is available as a web application. [1]

5.7.10 Mocha

Mocha is a test framework for NodeJS applications, which allows for serialization of testing during continued development for building automated unit tests of pages produced. Test driven development (TDD) allows for the creation of a test framework to be run at each change pushed to code, allowing for the automation of testing and for clear feedback to be given to the developer. In this way, bugs are catalogued during development with a lower overhead given to hiring UI/UX testers, or burdening the developer with manually checking each element of the program. Mocha and information about it is available on the project webpage. [22]

5.7.11 Github

GitHub is a website which allows for the hosting of git-based software repositories at no cost [11], and provides a number of features and frameworks which allows for easier management of development operations. Available both as a web interface and as an API, GitHub provides storage and version control for code and a number of related tools. The current repository for the NDSU Pulse Crops program is located at project page Bandillo-Lab on Github, and all code produced for this application will reside there.

5.8 Database Schema

The database schema for this project is a data schematic, showing the relationships between information sets which will be stored in the database as tables. Using the framework provided by SQLite, this information will be used to provide content for the functional model of the application.

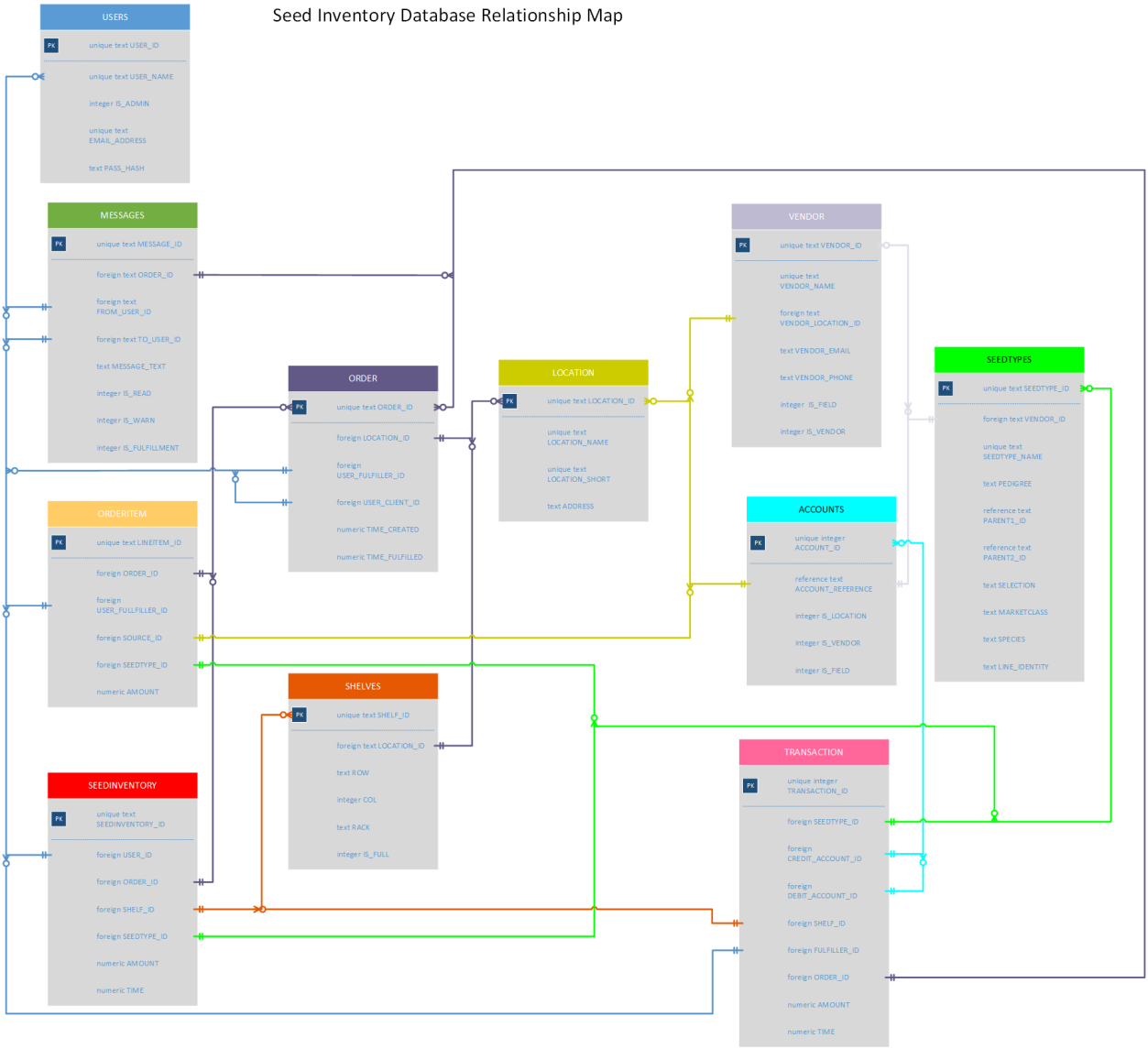

Figure 7: **Seed Inventory Database Schema** this diagram shows the database schema for the proposed system, with tables linked by inherited data flows via foreign keys to produce a stateful representation of the inventory system.

Primary IDs For both system security and for program flexibility, each row of each table will be given a system-side (rather than user-visible) unique primary key, which will be six randomized alphanumeric characters prefaced by a three character code based on its table membership, separated with an underscore. For instance, an ID for the USER table will be in the form **USR_XXXXXX**. This nomenclature prevents cracking of database structure via sequential ID calls, and allows for for 62^6, or over 56.8 billion database entries before the namespace is exhausted.

5.8.1 Users

The USERS table contains all information about the current system users. This information is as follows:

- A primary key unique alphanumeric text USER_ID, with the three character prefix USR
- A unique text USER_NAME
- An integer, 0 or 1, IS_ADMIN, which provides differing privileges for administrator and worker user accounts
- A unique text EMAIL_ADDRESS, for future system email integration
- A text PASS_HASH, which will be the hashed password of the user (no plain-text passwords will be saved in the system). This field will be supplied by the Argon2 hashing algorithm, which is recommended by OWASP for all new applications. [13]

5.8.2 Locations

The LOCATIONS table contains information about locations where the seeds are to be stored, or placeholder locations of seed sources. These locations have the following properties:

- A primary key unique alphanumeric text LOCATION_ID, with three character prefix LOC
- A unique text LOCATION_NAME
- A unique text LOCATION_SHORT similar to the location abbreviations in Agrobase
- A text ADDRESS

5.8.3 Vendors

Vendors are any possible source of seeds within the program, and include external vendors (such as private companies, other universities, or government bodies like the USDA) or internal sources (such as field activities, winter greenhouse, and other locations). The VENDOR table has the following properties:

- A primary key unique alphanumeric text VENDOR_ID, with three character prefix VND
- A unique text VENDOR_NAME
- A foreign text VENDOR_LOCATION_ID, which will either be NULL for external

vendors or a reference to another location in the program for internal vendors

- A text VENDOR_EMAIL which may be NULL
- A text VENDOR_PHONE which may be NULL
- An integer IS_FIELD, set to 1 if this vendor is a field, 0 otherwise
- An integer IS_VENDOR, set to 1 if this vendor is a literal external vendor of seed, 0 otherwise

5.8.4 SeedTypes

The SEEDTYPES table contains all information pertaining to stored types of seeds. This information is separate from the seed inventory itself, and as such each seed inventoried will have a seed type information set associated with it. SeedTypes have the following properties: seed

- A primary key unique alphanumeric text SEEDTYPE_ID, with three character prefix STP
- A unique text SEEDTYPE_NAME identifying the line
- A foreign text VENDOR_ID, associating the seed type with its source
- A text PEDIGREE
- A text PARENT1_ID which can either be NULL or a reference to another row in the SEEDTYPE table
- A text PARENT2_ID which can either be NULL or a reference to another row in the SEEDTYPE table
- A text SELECTION, detailing the selection phase of the seed
- A text MARKETCLASS, detailing the market-class of this seed
- A text SPECIES, detailing the species
- A text LINE_IDENTITY, for further internal association with line nomenclatures
- A numeric THOUSAND_SEED_WEIGHT, describing the weight of 1000 seeds of this type

5.8.5 Shelves

The SHELVES table contains all information of shelves in the program. Shelves have the following properties:

- A primary key unique alphanumeric SHELF_ID, with three character prefix SLF

- A foreign non-null text LOCATION_ID, placing the shelf at a physical location in the program
- A text COLUMN (A-Z,AA-ZZ)
- An integer ROW (1-N)
- A text STACK (I, II, III, etc.)
- An non-null integer IS_FULL, which is either 1 for a full shelf or 0 otherwise.

5.8.6 Orders

The internal logic of the system is based on the fulfillment of orders, which are internal invoices of planned changes to the inventory system. the ORDERS table can be thought of a list of orders for workers to fulfill on. The ORDERS table has the following properties:

- A primary key unique alphanumeric ORDER_ID, with three character prefix ODR
- A foreign, non-null LOCATION_ID referencing what site the order is for
- A foreign, non-null USER_CLIENT_ID referencing the administrator creator of the order
- A foreign USER_FULFILLER_ID, which is either NULL when an order is not fulfilled and otherwise a reference to the user who completed the order
- A numeric non-null TIME_CREATED
- A numeric TIME_FULFILLED which is initially null and timestamped at order completion

5.8.7 OrderItems

Rather than the converse, where Orders may have an unknown number of Order Items, all Order Items instead have a foreign key to its originator in Orders. The ORDERITEMS table contain these order items, which can each be thought of as a change requested for the seed inventory. The ORDERITEMS table has the following properties:

- A primary key unique alphanumeric ORDERITEM_ID, with three character prefix ITM
- A foreign ORDER_ID reference to its parent order
- A foreign SOURCE_ID which is a reference to a LOCATION of either a vendor or another internal seed inventory location
- A foreign SEEDTYPE_ID detailing what type of seed to be moved

- A numeric AMOUNT giving the count or weight of the order item, positive for additions to inventory and negative for seeds removed

- A text UNIT giving the unit of the amount

5.8.8 Messages

The message system is a user interface element allowing for communication between users on the platform. These messages may be automatically generated (on order fulfillment, order creation) or be the result of a user's interaction with the system (a worker flags an order as having a problem). The MESSAGES table has the following properties:

- A primary key unique alphanumeric MESSAGE_ID, with three character prefix MSG

- A foreign ORDER_ID reference to its parent order, which may be NULL if no order is related to the message

- A foreign FROM_USER_ID referencing the sender of the message

- A foreign TO_USER_ID referencing the receiver of the message

- A text MESSAGE_TEXT field

- An integer IS_READ which is either 1 if the receiver has seen it, or 0 otherwise

- An integer IS_WARN, which is ether 1 for priority messages, or 0 otherwise

- An integer IS_FULFILLMENT, which is either 1 for fulfillment confirmations, or 0 otherwise

5.8.9 SeedInventory

The seed inventory is the heart of the system and the primary goal of the system is to establish a granular command and control system over this information. The SEEDINVENTORY table is a list of all seeds in the inventory system, and has the following properties:

- A primary key unique alphanumeric SEEDINVENTORY_ID, with three character prefix SDI
- A foreign USER_ID referencing the user who placed the seed in its current spot
- A foreign SHELF_ID referencing the inventory item's current location
- A foreign ORDER_ID referencing the order which pulled the inventory item into the system
- A numeric AMOUNT, a seed weight in grams
- A text UNIT
- A numeric TIME of when it was last modified
- An integer IS_DIRTY clearly marking whether the seed has been cleaned

5.8.10 Accounts

An account is an abstract entity created for the accounting purposes of the seed inventory system. Every location has an associated account, and this account is used to populate the information of the TRANSACTIONS table. the ACCOUNTS table has the following properties:

- A primary key unique alphanumeric ACCOUNT_ID, with three character prefix ACC
- A foreign non-null LOCATION_ID
- A foreign VENDOR_ID which is NULL for internal locations
- An integer IS_LOCATION for internal locations, which is 1 if the account describes an inventory location and 0 otherwise

- An integer IS_VENDOR, which is 1 if the account describes a literal vendor and 0 otherwise

- An integer IS_FIELD, which is 1 if the account describes a field activity and 0 otherwise

5.8.11 Transactions

A transaction is any change made to the inventory database across its entire life, and this is the 'general ledger' of what could be described as a double-entry accounting system. As changes are made to the inventory, it is credited or debited, and a corresponding, mirror-image debit or credit is applied to another account reflecting the movement of seeds across the system. This TRANSACTIONS table is used to provide auditing for the system, and as such is an indelible record for any given accounting period. This table will only reset in length at the end of a seed inventory audit, on date the table will be compressed and saved outside of the database. The TRANSACTIONS table has the following properties:

- A primary key unique alphanumeric TRANSACTION_ID, with three character prefix TSN

- A foreign SEEDTYPE_ID reference

- A foreign CREDIT_ACCOUNT_ID reference

- A foreign DEBIT_ACCOUNT_ID reference

- A foreign SHELF_ID reference

- A foreign ORDER_ID reference

- A numeric AMOUNT

- A text UNIT

- An integer IS_DIRTY

- A numeric TIMESTAMP

5.9 Functional Model and User Experience

The overall user experience and functionality of the web application will be through the web browser and initially hosted at the Pulse Program webpage (https://pulse.ccast.ndsu.edu/). Upon navigation to this site, the user will be asked to login using their username and password.
Once the user has logged in, they will be taken to a landing: page, showing information including outstanding orders, recently fulfilled orders, and general system information at a glance. A top menu will display an icon, indicating any waiting messages for the user. On the left-hand side of the application, a collapsible menu will show the following options:

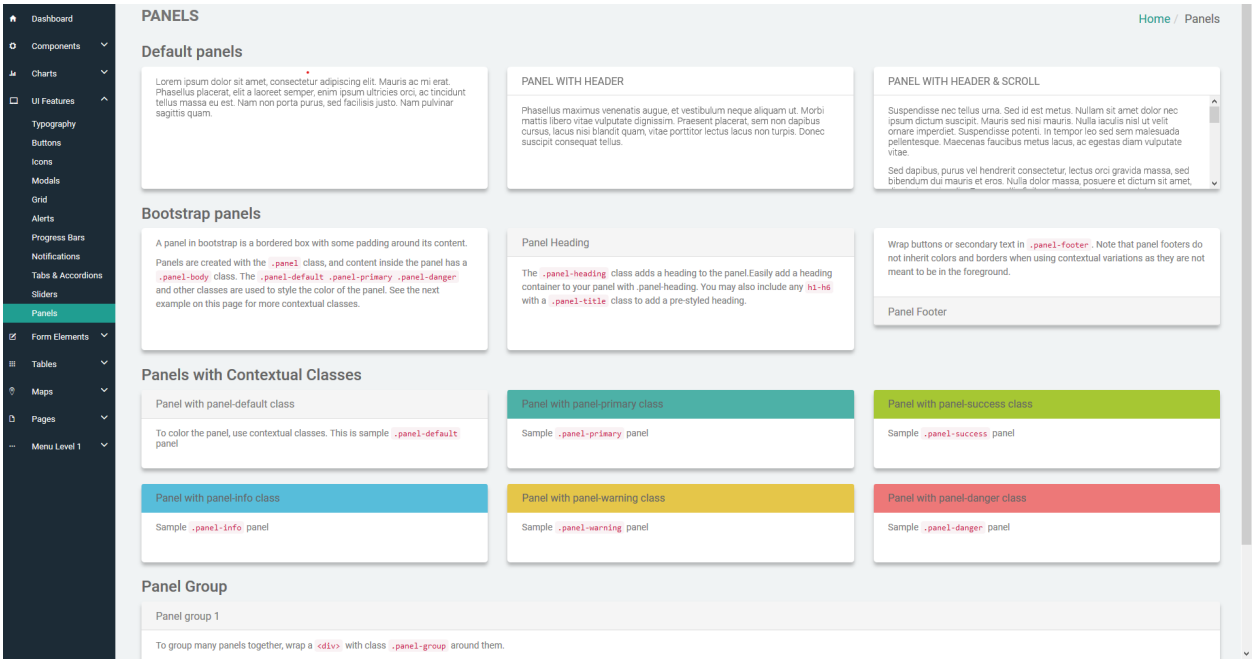

Figure 8: **UI/UX Proof of Concept Mockup** Generated from free and open source libraries previously mentioned, this proof of concept web interface was demoed for the lab members to demonstrate the layout, look, and feel of a fully implemented system.

- **Home** the current page

- **Current Inventory** a table of all inventory, with drop down menus and text input elements used to search this table and narrow down information. This will include a button to export current view as a CSV.

- **Locations** a table view of all locations in the database, allowing selection of location to view more specific information for each and three additional functions
  - **Map** generate a map of the current inventory of a selected location
  - **Location Orders** view outstanding and fulfilled orders by location
  - **Location Inventory** similar to 'Current Inventory' view, but with the current location already selected
- **Current Orders** a queryable view of all current orders in sequence
  - **Order Fulfillment Pages** generated by and accessible by each outstanding order, provides a workflow for checking inventory items into or out of the database. Workers can scan bar-codes or key in item IDs to pull changes across the inventory system here.
    The following additional pages will be available to administrator accounts:
- **Create New Order** generate an internal invoice to push changes to the inventory by selecting seed types, sources, and amounts using text boxes and drop down menus
- **Vendors** view, modify, create, and delete vendor information
- **Users** view, modify, create, and delete users
- **Audit Log** view and query the transaction-level log of the database, output as CSV, or generate an end-of-period audit to close the ledger and begin a new one
- **Import Data** import data into tables from external dataset: provides interfaces for import via CSV file or SQL connection, then shows changes to be made (allowing modifications).

6 Virtual Machine Environment Planning

6.1 Introduction

Since the first full DNA sequencing by Sanger in 1977, a series of breakthroughs in genomic sequencing, taken together as Next Generation Sequencing techniques, has applied computation toward genomics problem sets. With the decrease in costs of information technology associated with laboratory data management, the decrease in costs to Next Generation Sequencing (NGS) technologies (far exceeding the rate of Moore's Law [54]), and the wider availability of reference genomes available for use by plant breeders [31] [26] the implementation of a NGS platform 'in house' is more cost-effective and feasible than ever.

The rationale for the creation of a genomics database platform for the Pulse Breeding Program at NDSU extends from the present state of research. At the present time, NDSU plant breeding programs purchase sequencing services from outside laboratory partners. This data, generated at a cost, is a valuable part of the breeding programs using it. Therefore, as an asset, the genetics data received from outside partners requires a data management system, which not only stores this data in a durable and highly-available manner, but also allows for analytics performed on this data, and keeps knowledge about the data within a managed system. Such a system would not only safeguard the investment made on this data, but also allow for improved usage of data integration of genomics datasets into continued development. Such a system would allow for plant breeders to leverage the vast number of free-and-open-source genomics software tools now available without the creation of ad-hoc systems for each experiment or program.

In addition, by keeping genomics development within the platform, one can expect time-savings, as workflows become more routine and algorithms more optimized to the problem sets at hand. Moreover, such a platform would open up new possibilities for research, not only for the development of agronomic gains in researched species, but also for the development of new algorithms, tools, and methodologies for the same as new resources become available.

Given the highly heterogeneous information environment and high throughput of lab data workflows, the management of information technology, and systems administration tasks have become part of the data science work done within laboratories today. Moving genomic data between HPC clusters, personal researcher computers, organization web-servers, and rented services requires programming interfaces and API calls. At the present time, the NDSU agronomics program does not have an integrated data platform for genomics data. The current database platform, while robust, is not fully featured enough for genomics, nor does its structure provide for an open platform for further internal development by researchers. Therefore, a technological solution to incorporate genomics research and techniques into the plant breeding pipeline should be explored. The first component of this would be the implementation of a genomics database.

A feasibility study was conducted to provide recommendations and draft a proposal for a curated virtual environment which would initiate a genomics database platform at NDSU, these goals were expanded to incorporate the scope of the previously described seed inventory

system as well. The goals of the implementation plan included in the proposal are as follows:

- Describe the problems currently faced by the pulse plant science team and provide for solutions within the scope of an information system.
- Provide cost estimates, cost-benefits analysis, and alternative implementation plans for IT components of the system.
- Summarize and provide a logical outline for eventual project management of the system.
- Summarize and link to the various supporting documentation cited to arrive at these solutions.
- Detail the major architectural design aspects of the system in greater detail, comprising:
  - **Functional Design** detail regarding major functional aspects of the system and how the system will access the information it needs from the information environment.
  - **Technical Design** detail regarding how the major components of the system will be implemented in hardware, software, networking, database, and policy.
  - **Confidentiality, Integrity, and Availability** will be discussed as a functional component within the scope of both these designs.
- Detail the documentation to be generated during the systems design, development, testing, and deployment processes, including the anticipation for change management and revisions.
- Act as an open and living document, allowing for comment and feedback to be provided.

6.2 Requirements

6.2.1 Functional Overview

At the present time, NDSU agronomics breeding outsources genetic sequencing to outside laboratory partners, which perform sequencing for sampled specimens. For this data to be made more useful to researchers, a database platform is necessary to further explore and analyze the relationships between different genomes and the variation between plant genotype and phenotype data. The features which will allow for the highest return on investment for the research community at NDSU would include high data durability, high data availability, integration of a knowledge management system, and genomics annotation and indexing.

1. Data Durability The following features are defined as requirements for the new storage platform in regards to the durability of data housed such that:

- data integrity features in-file-system to ensure reliability for the data.
- high I/O performance to enable better response to end users.
- protection against data corruptions.
- file system snapshots for quick online backups, which don't load the servers.
- quota allocations per user.
- hardware level redundancy and hot-swappable hardware.
- off-site backup performed with both incremental and full backups.
- automated backup and scheduled testing of backups with notifications.

2. Data Availability The following features are defined as requirements for the new storage platform in regards to the availability of the data housed such that:

- continued availability during hardware or software failure built into the system via HA pair or cluster topology.
- real-time compression to optimize storage and also achieve higher performance.
- data should be agnostic to any front end or workstation OS, and be called via API for common tools already used by researchers (R, Python, etc.)
- data should be usable for common bioinformatics and statistical use cases, including next-generation modeling and Machine Learning.
- no single point of failure in systems hardware effects overall availability of data.
- usage of mature, commodity hardware to ensure replacement of components in case of failure and high uptime.
- proper access controls and strong encryption employed to ensure data is available within the organization and is not exposed to non-organization members, even when moved across third-party systems.
- cold spare platform in place for disaster recovery.

3. Knowledge Management In the context of informatics, empirical observations, as the data of an information system, drive the creation of information, with the analysis and processing of data yielding scientific conclusions. The data stored is not an end onto itself, but is instead made useful by revealing its underlying structure through a series of processes, procedures, and methodologies. The imperative and descriptive knowledge, taken together as organizational knowledge, is itself a core competency of laboratory science. This drive to retain knowledge within organizations is challenged by the diverse yet related datasets within biological sciences and by the communication overhead of research team members

performing work separated by time, space, and across disciplines. [15]

While extensive work has been done by NDSU for the creation of data management systems for phenotype data, there exists no knowledge management system for the retention of internal processes and methodologies related to processing this data. As genomic data management is included, the scale and complexity of data within the information system will increase, and with it, the value of a knowledge management system as a means to both facilitate collaboration and reporting between researchers, and to provide return on research investment as documented expert knowledge. With knowledge management, this knowledge also remains within the information system even when experts are not available. Therefore, in addition to the proposed data management system, it is a functional requirement of the new system to include a knowledge management system, which will allow researchers to produce documentation of processes, procedures, and methodologies and for this documentation to be managed with similar durability and availability requirements as the data stored.

4. Data Pre-processing Raw genomics data provided to NDSU breeding program is provided in FASTA/FASTQ format, which consists of both raw reads of sequences and comments provided per run. Even alone, this data presents a technical challenge to genetics-driven research [43]; however, the density of genetics sequence data is also an informatics challenge. For data to be useful to data-driven decision making, datasets must be both easily searchable and available for comparison. By default, both of these properties are difficult to obtain with the FASTA/FASTQ format, given both the density of data and the lack of context given to this data. For example, the whole-genome sequencing kept on reference by the NCBI for *Pisum sativum* is 4.6GB of data when uncompressed, or over 61 Million lines of 80 character width text with description lines included [39]. Mapping this raw genetic sequence into genes, regulatory mechanisms, introns, exons, phenotypes, etc. to determine the relationships which make the data useful both in systems biology and plant breeding is a primary challenge in bioinformatics as a practice. Two techniques to expose these relationships, and help to reduce the technical overhead necessary for researchers to interact with genetics datasets: genome annotation and genome indexing, taken together, serve to expose the most pertinent structures of genetics datasets for further study, and as such are valuable labor-saving tools and are functional requirements for the proposed system.

6.2.2 Proposed System

Having enumerated through the functional components necessary for genomics research, the following provides an overview of the software fulfilling the technical requirements which enable these functions. System design incorporated requested comments from NDSU BPDM and Bandillo Lab members.

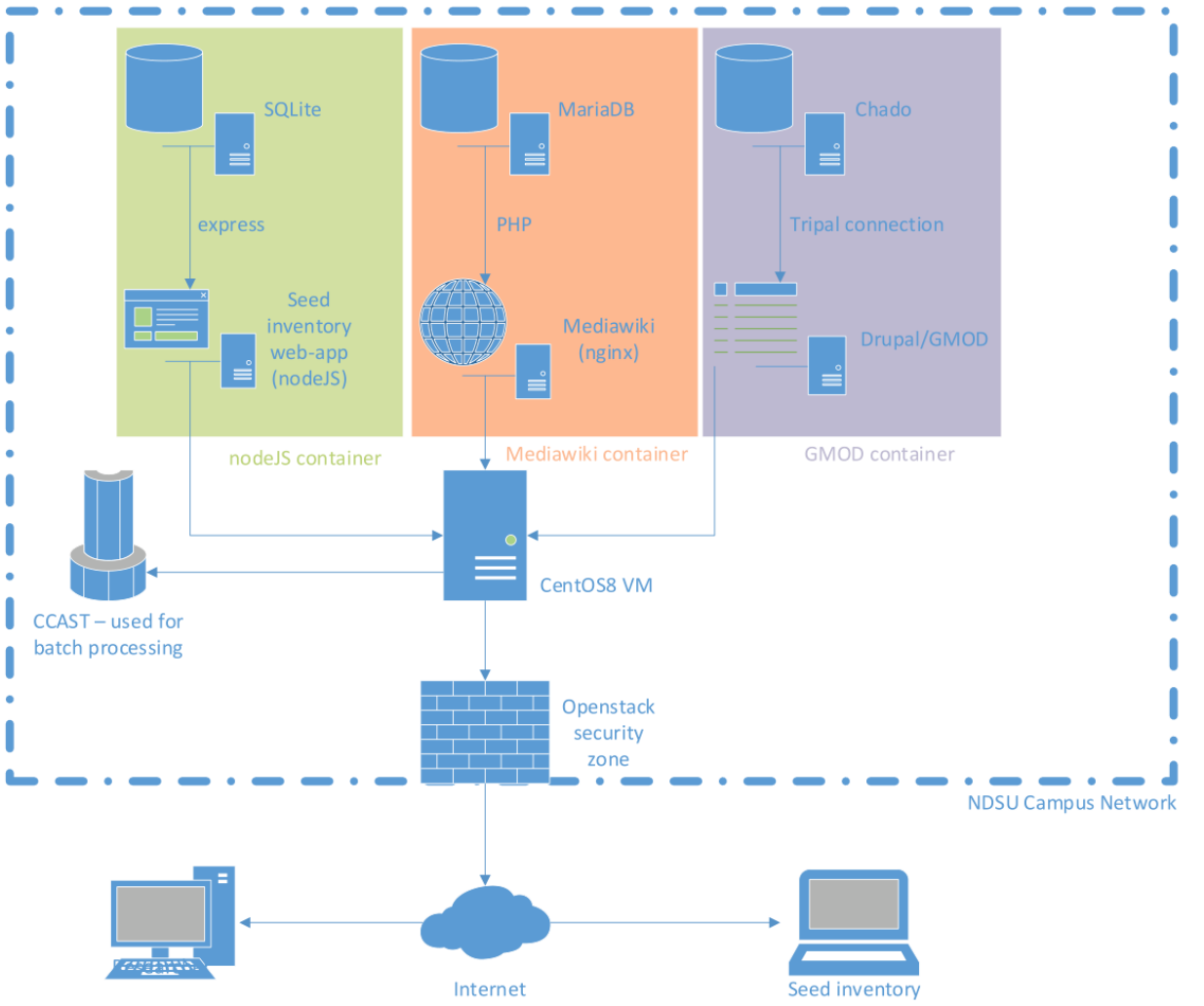

Figure 9: **Virtual Machine IT Environment** this diagram shows the three virtual machines identified as database applications useful to the breeding program's data analytics goals. At the present time, only the Centos host and the Mediawiki portion of this environment has been implemented, while the remaining have been approved for continued work.

The system consists of an application host server running within the NDSU CCAST security zone of the NDSU network. This virtual machine consists of a Centos 8 image running on an Openstack [48] virtual network, and is itself running Docker [47] to containerize application functionality. By leveraging the pre-existing network infrastructure of CCAST, this system can not only use the redundancy and fungibility of that system, but also allows for incredibly high throughput processing of openPBS scripted data science and machine learning tasks via the CCAST Thunder2 HPC cluster.

To fulfill the scope of the laboratory knowledge management system for genomic data pro-

cessing, Mediawiki [36] was chosen, with MariaDB [34] providing database back-end. Maintained by the Mediawiki Foundation, Mediawiki provides for a simple web application stack with builtin document revision history tracking and its own document formatting hypertext language, Wikitext, allowing for detailed instructions and knowledge to be retained in the organization by researchers. By providing a private wiki instance, a shared location for scientists to share procedures, templates, and information is created. Furthermore, this repository also allows documentation of all work done for genomics database, allowing the system to become largely self-documenting.

As for the genomics database itself, review of current university best-practice confirmed that an instance of the Generic Model Organism Database system (GMOD) [6] would provide the needed functionalities of scripted data pre-processing, gene annotation, and gene indexing. Utilizing the free and open source Drupal content management framework, GMOD allows for parallel genomic data work-tasks via Chado, its database back-end, by scripting in BioPerl [51].

As previously described, the final container running on the system is the seed inventory system, incorporating a nodeJS based web application providing researchers and agricultural workers access to a SQLite database system via express RESTful [17] middleware. As opposed to the other software, which will be taken from available open source packages, this inventory system will instead be built in-house to address the lack of free and open source inventory systems for plant breeders.

6.2.3 Work Performed

Cybersecurity Hardening A guideline was developed using guidelines from USDA and FDA Cybersecurity Infrastructure Plan [12], NIST SP 800-53 Rev 5 [41], and CIS benchmarks [18] to develop a hardening checklist, consisting of configuration changes from a base Centos 8 system running within NDSU CCAST. The entirety of these scripts and configuration modifications are available in **Appendix D**.

Mediawiki implementation In order to facilitate the change management overhead of continued systems improvements and implementations, a private wiki instance was used to document processes, policies, and proposal revisions. This system used Mediawiki running in a container inside of Docker, and a full documentation of the procedure for the creation of a lab wiki is in **Appendix E**

6.3 Future Work

At the present time, a virtual machine host and mediawiki implementation has been implemented in full. Future work on this system includes the creation of the seed inventory system web application as a free and open source Docker image, and the integration of GMOD for genomic data annotation and analysis.

7 Conclusions and Research Direction

The data science and statistical analysis tasks demonstrated in the course of research suggest that continued decreasing costs of high-throughput sequencing and phenotyping by sequencing technologies will, in turn, increase the network administration and systems administration tasks undertaken by bioinformaticians, statisticians, and data scientists. In addition, the size of genomic datasets and information density of genetic annotation data requires careful planning in the creation of robust and highly-available database systems to provide access to this data. The heterogeneous nature of the information network systems traversed during research also suggests that training in both management of information systems and cybersecurity training should also be pursued by data scientists and bioinformaticians as supplemental to and understanding of the field. Most tasks can be performed with free and open source software (FOSS) libraries. This body of FOSS code and its supporting documentation provides for custom analysis software pipelines and applications to be assembled from these libraries. However, with this comes increased overhead in time spent maintaining software and creating documentation for laboratory processes, workflows, and data output is of the utmost importance to data scientists to reduce costs from the creation of ad-hoc programming tools. Document management systems and knowledge management such as git and mediawiki are easy to implement and provide for a reduction in research time spent when used in a laboratory setting alongside more familiar database platforms, adding context and process awareness to data. Taken together, the frameworks and techniques presented here contribute to continued discovery of data analysis and management of information systems best practices, to be expanded upon.

Appendices

A Agrobase Audit

The following code were used to perform auditing of the Agrobase system.

A.1 Abandoned Experiment Identification

The following script was one of a series developed to identify abandoned experiments in the database, for which no experimental data was recorded. Three similar scripts were used, one each for the pea, chickpea, and lentil datasets. The script shown displays all abandoned experiments for the chickpea dataset. Note that metadata information of raw experiment data was saved across several tables.

```
use chickpea
--select * from dbo.tre01; --this is the table that contains treatments
    varieties, populations, parents

--select * from dbo.exp01; --this table contains information about experiments

--select * from dbo.EXD01; --information related to experimental design keys

--select * from dbo.EXD02; --contains raw data for all traits

--select * from dbo.TRD01; -- contains the names of all the traits of experiments

--select * from PEA.dbo.VW_TreatTraitsWN; -- secret decoder ring for trait
    numbers

--select * from PEA.INFORMATION_SCHEMA.COLUMNS where COLUMN_NAME = 'TBENT1'
    order by TABLE_NAME ASC; --dump every table column name in the DB schema

declare @t_exp table (
    c_id int,
    c_year int,
    c_name varchar(50)
);

declare @t_data table (
    c_id2 int,
    c_data varchar(500)
);

insert into @t_exp (c_id,c_year,c_name ) select tbid,tbyear,tbname from
    dbo.exp01;
```

```
insert into @t_data (c_id2, c_data) select
    dbo.exd02.TBKEXPT,
    TBDATAC
    from dbo.exd02i
    where TBDATAC != ('-9.0');
delete from @t_data where c_data = '';
delete from @t_data where c_data like ('%-9%');


delete @t_exp
from @t_exp
inner join @t_data on c_id = c_id2
where c_id = c_id2;
select * from @t_exp
select * from @t_data order by c_data asc
```

A.2 Outlier Identification

The following script was one of a series developed for outlier identification. This script identifies all non-null, non-missing entries four standard deviations from the mean for the 'test weight' (pounds per bushel) data for the lentil dataset. A different script was deployed for each of the agronomic traits described by Agrobase due to the peculiarities of string to numeric data conversion performed by Agrobase.

```
use lentil;

--get metadata for the trait from the db
declare @trait int = 25;
declare @datatype char = (select TBFLD_TYPE from TRD01 where TBID = @trait);
declare @data_len int = (select TBFLD_LEN from TRD01 where TBID = @trait);
declare @data_dec int = (select TBFLD_DEC from TRD01 where TBID = @trait);
print @data_len;
print @data_dec;

--make a table in memory for exporting information
declare @t_exp table (
       c_id int,
       c_year int,
       c_name varchar(50),
       c_mean decimal(20,1),
       c_sigma float(5),
       c_outliers varchar(500)
);

--make a table in memory for storing raw plot data
declare @t_data table (
```

```
        c_id2 int,
        c_data decimal(20,0)
);

--populate the export table with information for all experiments which are not
    multiyear
insert into @t_exp (c_id,c_year,c_name ) select tbid,tbyear,tbname from
    dbo.exp01 where TBYEAR>0;

--populate the data table with non-null, non-garbage raw data for the trait
    selected
declare @script nvarchar(max);
set @script = N'select dbo.exd02.TBKEXPT, cast(cast(TBDATAC as float)as
    decimal(' + cast(@data_len as nvarchar(2)) + N',' + cast(@data_dec as
    nvarchar(20)) + N')) from dbo.exd02 where TBKTRAIT =' + cast(@trait as
    nvarchar(20));
insert into @t_data (c_id2, c_data) exec(@script);

delete from @t_data where c_data = NULL;
delete from @t_data where c_data = 0;
delete from @t_data where c_data like ('%-9%');

--remove any experiments which have no raw data whatsoever from the export table
delete from @t_exp
    where not exists(select NULL
        from @t_data where c_id2 = c_id);

--perform calculation of means and sigma values
declare @currentID int;

declare idx cursor local static read_only forward_only
for select distinct c_id from @t_exp;

open idx;
fetch next from idx into @currentID;
while @@FETCH_STATUS = 0
begin
    update @t_exp set c_mean =
        (select avg(c_data)
        from @t_data where c_id2 = @currentID)
        where c_id = @currentID;
    update @t_exp set c_sigma =
        (select STDEV(c_data)
        from (
            select c_data
            from @t_data
            where c_id2 = @currentID)
```

```
        c_data)
        where c_id = @currentID;
    fetch next from idx into @currentID;
end;
close idx;
deallocate idx;

--calculate outliers
declare @i int = 1;
declare @index int = 0;
declare @rowCount int = 0;
select @rowCount = COUNT(0) from @t_exp;
declare @stdRange float = 4.0;
declare @lower decimal(20,1);
declare @upper decimal(20,1);
declare @mean decimal(20,1);
declare @std float;
declare @id int;
declare @buffer varchar(500);
while @i <= @rowCount
begin
    set @id =
    (select c_id from (
        select ROW_NUMBER() over (order by c_id asc) as
        RowNum,
        *
        from @t_exp
    ) as c_id
    where RowNum = @i);

    set @std = (select c_sigma from @t_exp where c_id = @id);
    set @mean = (select c_mean from @t_exp where c_id = @id);
    set @lower = @mean - (@stdRange * @std);
    set @upper = @mean + (@stdRange * @std);
    set @buffer = '';

    set @buffer = @buffer +
           (select string_agg(c_data, ';')
           from (select c_data
                  from ( select c_data
                     from @t_data
                     where c_id2 = @id
                  ) c_data
                  where c_data not between @lower and @upper
              )c_data
           )
```

```
    update @t_exp set c_outliers = @buffer where c_id = @id;
    print @id;
    print @buffer;
    set @i = @i + 1;
end;

update @t_exp set c_outliers = ('') where c_outliers is NULL;
select * from @t_exp;
```

B Nomenclature Revision

I. The Nomenclature for Pea, Chickpea, and Lentil will be a standardized schema which is both human-readable and machine-searchable.

II. All names will be rendered in upper case characters only.

III. The Nomenclature will be in the following order:

a. Year - two numeric digits **00**

b. Species - one character: **P**(ea), **C**(hickpea), or **L**(entil)

c. Experiment type - one character: **R**(egular), **S**(pecial), or **U**(nknown)

d. A seperating underscore _

e. Experiment code - two numeric digits + one following character, as follows:

(a) Regular Trials will have a three character alphabetic code, as follows:

- **MPB**: Mapping populations, bi-parental
- **MPM**: Mapping populations, multi-parental
- **CXB**: Crossing Block: greenhouse or field (F1)
- **WTN**: Winter nursery (WN) (F2-F4)
- **SGN**: Segregating Nursery (F2-F5)
- **SPS**: Single plant selection (F3-F5)
- **OYT**: Observational Yield Trial (F6)
- **PYT**: Preliminary Yield Trial (F7)
- **AYT**: Advanced Yield Trial (F8-F9)
- **RYT**: Western regional yield trial (F10)

- **CVT**: Core variety trial/Statewide testing (F10-F11)

(b) Special Trials will have a two character alphabetic code, as follows by Agrobase:

- **NG**: New germplasm: greenhouse or field
- **SI**: Breeding seed increase
- **BS**: Biotic stress test
- **AS**: Abiotic stress test
- **PT**: Fee test, independent experiment/private testing
- **GS**: Grad student work

(c) Unknown are reserved for historical data within Agrobase for which no originating documentation exists, these will have a 3 number code starting with 00, such that per year, each unknown is in the form **001**, **002**, **003**, etc.

f. An additional underscore __

g. Additional Information - if the biotic stress test, abiotic stress test, or grad student work is entered, then an additional field is required, followed by an underscore:

(a) biotic or abiotic: specify what stressor is being tested

(b) grad student work: specify grad student in form of first initial + last name

(c) preliminary yield trial: historical usage for designation of market class splits

h. Location - three character code for locations

IV. Usage Examples

- **20PR_AYT_MNT** = 2020 Pea AYT Minot
- **10PU_001_MNT** = 2010 Pea unknown (for coding PRIL found in Agrobase)
- **20PS_GS_JSTEFFES_FAR** = 2020 Pea grad student work Fargo
- **12PR_PYT_G_MNT** = 2012 PYT Green pea Minot
- **16LR_CXB_GHM** = 2016 Crossing Block for Lentil in Minot Greenhouse
- **20LR_CXB_GHF** = 2020 Crossing Block for Lentil in Fargo Greenhouse

C Genome-Wide Agronomic Selection - Scripting and Summary Statistics

C.1 Creation of SNP Matrix

The following scripting was used to create a numeric imputation of the SNPs from a VCF genotype file, and a following script performs a scripted renaming of this matrix's header information to match it with phenotypes in Agrobase.

```
    cat pea_vcf.vcf | awk '$1 ~ /^#/ {print $0;next} {print $0 | "sort -k1,1
        -k2,2n"}' > pea_sorted.vcf
/tassel-5-standalone/run_pipeline.pl -Xms512m -Xmx10g -vcf ./pea_sorted.vcf
    -NumericalGenotypePlugin -endPlugin -export pea.prob -exportType
    ReferenceProbability
```

```
    $ sed 's/,/ /' ~/documents/t.GBS.H.pea.bd.csv | awk '{print $1}' | cut -d_
        -f2,3 | sed 's/_PSP//' | sed
's/PS//' | sed 's/PS_//' | sed 's/_$//' | sed 's/^_//' > tempfile.csv
$ paste -d "," tempfile.csv ~/documents/t.GBS.H.pea.bd.csv > ~/cleannames.csv
$ cut -d',' -f2 --complement ~/cleannames.csv | sort > ~/output.csv
```

C.2 Parallel Genome Wide Agronomic Selection

The following code performed GWAS analysis of pea data using the CCAST Thunder2 HPC cluster.

```
# Code for parallel Genome Wide Agronomic Selection
# R package dependencies list: rJava, devtools, BGLR, GSwGBS, rrBLUP, listenv,
    doParallel, pls, glmnet, randomForest
# Original code by Dr. Md Abdullah Al Bari and Stephen Szwiec, August 2020

#!/usr/bin/Rscript

library("BGLR")
library("GSwGBS")
library("rrBLUP")
library("parallel")
library("listenv")
library("doParallel")
#library(dplyr)
#data(wheat) #Load example data set
#########################################################
##                                    Begin for Editing
#########################################################
cores <- detectCores()
cl <- makeCluster(cores[1] - 1)
registerDoParallel(cl)
```

```
today=Sys.Date()

#write.csv(Pheno, 'Pheno_Q_GS.csv', row.names = FALSE)
#pheno preparation
pheno <- read.csv('pheno.csv',head = T, stringsAsFactors = F, na.string="NA")
geno <- read.csv('geno.csv', head = T, stringsAsFactors = F)

dir.create(paste(".","/GS_CV_rrBLUP/",sep=""))
setwd('./GS_CV_rrBLUP')
#setwd("..")
a=2 # why a is 2 ? pheno traits starts with 2nd col
#a=11# when I need just one traits, located on 11th col
acc<-matrix(ncol=7, nrow=10)
colnames(acc)<-c("Fold", "Trait", "P_value", "r", "lower95_CI", "upper95_CI",
    "h2")
pred_valid<-matrix(ncol=2)

while(a<=ncol(pheno))
{
  #working on cv loops
  name<-colnames(pheno)[a]
  print(paste("Working on trait", name))
  b<-pheno[,c(1,a)] # all rows and col 1, and a col[a is variable col]
  d<-na.omit(b)# remove missing values
  d<-data.frame(merge(d, geno, by="Genotypes")) #dataset with only trait, no NA
      and genos 3:ncol(d)
  r=nrow(d)
  tr=(nrow(d)*0.8)
  pred_valid<-matrix(ncol=7)# creating 7 col NA
  colnames(pred_valid)<-c(name, "RRBLUP", "GAUSS", "PLSR", "ELNET", "RF", "AVE")
  #colnames(pred_valid)<-c(name, "RRBLUP")

  i=1
  while(i<=10)
e Variant Call {
    print(paste("Working on trait ", name, " fold ", i))
    ##Randomize training and testing pops
    train<-as.matrix(sample(1:r,tr))
    test<-as.matrix(setdiff(1:r,train))
    TR<-data.frame(d[train,]) # dim(d)=238 41343, TR is 0.8 of that
    TE<-data.frame(d[test,]) # row of test and all col of d

    ##put data into training and testing pops
    pheno_trn=as.matrix(TR[,2]) # all row of TR with col 2
    geno_trn=as.matrix(TR[,3:ncol(TR)])# all row TR and 3 to onward cols
    pheno_valid=as.matrix(TE[,2], row.names=TE$Genotypes)
```

```
    colnames(pheno_valid)<-name # different name in different cycle
    geno_valid=as.matrix(TE[,3:ncol(TE)])

    #GS
    tmp=GS.model(phenoTrain=as.numeric(pheno_trn), genoTrain=geno_trn,
        genoPredict=geno_valid)
    tmp<-data.frame(pheno_valid,tmp)
    tmp <- tmp[,1:2]
    pred_valid<-c(pred_valid,tmp)
    write.csv(pred_valid, file=paste("Trait_CV_",name,"_values_",today,".csv",
        sep=""), append=TRUE, row.names=T)

    #h2
    model<-mixed.solve(as.numeric(pheno_trn), Z=geno_trn)
    h2=round((var(pheno_trn)-model$Ve)/var(pheno_trn),2)

    #Test accuracy of Avg model
    c<-cor.test(tmp[,1], tmp[,2])
    cat(paste("correlation trait", name, "round", i))
    cat(cor.test(tmp[,1], tmp[,2])$estimate)
    acc[i,]=cbind(c(i, name, c$p.value, c$estimate,c$conf.int[1],c$conf.int[2],
        h2))
    write.csv(acc, file=paste("Trait_CV_", name,"_GSAccuracy_",today,".csv",
        sep=""), append=TRUE, row.names=F)
    i=i+1
    remove(train, test, TR, TE, pheno_trn, geno_trn, pheno_valid, geno_valid, tmp)
  }
  a=a+1
  remove(b,d,name, r, tr, pred_valid)
}

stopCluster(cl)

paste("finished with loop")
```

C.3 PBS Script for compute node pooling

The following code allows previous R script to be run in a parallel manner, with the slowest thread's wait time as the determining factor of overall speed, as per Amdahl's Law.

```
#!/bin/bash
#
#PBS -q default
#PBS -n CV_GS_2020_mdbari
#PBS -l nodes=12:ppn=2
#PBS -l walltime=15:00:00
#PBS -l mem=5gb
```

```
#PBS -o CC_GS_2020_mdbari.log
#PBS -m abe
#PBS -M md.bari@ndsu.edu
#PBS -W group_list=x-ccast-prj-bandillo
preliminary(){
    echo Beginning Work
    cd /gpfs1/projects/nonoy.bandillo/bari/R_analyses/GS_CV_Pulse_DB/
    module load R
    chmod a+x ./CCAST_CV_GS_2020.R
    timestamp=$(date +%s)
}
runprogram(){
    Rscript ./CCAST_CV_GS_2020.R
}
early(){
    timestamp2=$(date +%s)
    difference=$($timestamp - $timestamp2 / 60 / 1000 )
    echo Job was terminated early after $difference seconds!
}
cleanup(){
    timestamp2=$(date +%s)
    difference=$($timestamp - $timestamp2 / 60 / 1000)
    module unload R:
    echo Job completed after $difference seconds!
}
trap 'early; cleanup' 2 9 15
preliminary
runprogram
cleanup
exit
```

C.4 Phenotypic Heritability and correlation

Table 1. Heritability and summary statistics for seed yield and other agronomic traits

Trait	Mean	Range	SD	CV(%)	H^2
DFF (days)	71	60-84	4.8	6.7	0.90
NoSeedsPod (Nos.)	5.7	4.4-6.9	0.5	8.5	0.84
PH (cm)	74	37.6-108.3	11.5	15.5	0.81
PodsPlant (Nos.)	18	15-23	1.5	8.3	0.50
DM (days)	104	99-112	2.4	2.3	0.51
SeedYield (Kg ha^{-1})	2918	1734-4463	451	15.4	0.67

DFF is days to first flowering; NoSeedsPod is the number of seeds per pod, PH is plant height, PodsPlant is the number of pods per plant, DM is days to physiological maturity, SeedYield is seed yield per hectare, SD is the standard deviation, CV is coefficient of variance, H^2 is heritability in the broad sense.

C.5 Comparison of predictive ability

Table 2. Predictive ability of genomic selection models for seed yield and agronomic traits

Traits	**RR-BLUP**	**GAUSS**	**PLSR**	**ELNET**	**RF**	**BayesCpi**	**RKHS**
DFF (days)	0.60 (0.57-0.63)	0.60 (0.58-0.63)	0.57 (0.53-0.61)	0.57 (0.52-0.61)	0.55 (0.52-0.58)	0.59 (0.55-0.63)	0.54 (0.5-0.58)
NoSeedPod	0.42 (0.37-0.48)	0.41 (0.37-0.47)	0.41 (0.36-0.46)	0.41 (0.35-0.48)	0.40 (0.35-0.45)	0.42 (0.38-0.46)	0.40 (0.34-0.48)
PH (cm)	0.39 (0.33-0.44)	0.38 (0.33-0.44)	0.42 (0.38-0.48)	0.37 (0.31-0.42)	0.45 (0.4-0.5)	0.45 (0.41-0.48)	0.43 (0.39-0.48)
PodsPlant	0.28 (0.22-0.33)	0.26 (02-0.32)	0.25 (0.2-0.31)	0.23 (0.17-0.29)	0.28 (0.22-0.34)	0.23 (0.17-0.29)	0.28 (0.23-0.34)
DM (days)	0.42 (0.36-0.47)	0.41 (0.36-0.47)	0.44 (0.39-0.5)	0.40 (0.34-0.46)	0.41 (0.35-0.46)	0.47 (0.43-0.5)	0.45 (0.4-0.48)
SeedYield (kg ha-1)	0.38 (0.34-0.42)	0.38 (0.34-0.42)	0.31 (0.27-0.36)	0.38 (0.33-0.48)	0.39 (0.35-0.44)	0.35 (0.31-0.39)	0.42 (0.37-0.48)

DFF is days to first flowering, PH is Plant height in cm, DM is days to physiological maturity.

D VM Hardening - Security Implementation

The following security checklist was provided to NDSU BPDM for internal improvement of virtual machines and was tested using Centos 8.

1. Filesystem
    - `/etc/fstab`
        - `/tmp` mounted with 'nodev,nosuid,noexec'
        - `/var`, `/var/log`, and `/var/log/audit`, and `/home` are on seperate partitions
        - bind mount `/var/tmp` to `/tmp`
        - set 'nodev' on `/home`
        - set 'nodev,nosuid,noexec' for `/dev/shm`
    - Set sticky bit on all world-writable directories
2. System Updates
    - set system to update automatically
    - set proper repositories for development libraries
    - reserve cron jobs for nessessary system downtime
3. Secure Boot
    - for the following directories, root ownership via `chown` and `chmod 600`
        - `/boot/grub/grub.cfg`
        - `/etc/fstab`
        - `/etc/grub.conf`
    - `sed -i "/SINGLE/s/sushell/sulogin/" /etc/sysconfig/init`
    - `sed -i "/PROMPT/s/yes/no/" /etc/sysconfig/init`
    - set a bootloader password for UEFI subsystem
4. Process Hardening
    - Restrict Core Dumps
        - edit `/etc/systemd/coredump.conf` and set `Storage` to `none` and `ProcessSizeMax` to `0`
        - run `Systemctl daemon-reload`
        - `echo "fs.suid_dumpable=0" >> /etc/sysctl.conf`
        - `sysctl -p`

    - Add line `hard core 0` to `/etc/security/limits.conf`
    - Add line `fs.suid_dumpable = 0` to `/etc/sysctl.conf`
    - Add line `kernel.exec-shield = 1` to `/etc/sysctl.conf`
  - Randomize Virtual Memory
    - Add line `kernel.randomize_va_space = 2` to `/etc/sysctl.conf`

5. OS Hardening
  - remove unused legacy services (rsh, rlogin, rcp, ypserv, ypbind, tftp, talk, etc.) if present
  - disable any services and applications started by xinitd or inetd not utilized
  - remove xinetd if not needed
  - disable or remove services which would not be utilized (DNS, SNMP, LDAP, etc.)
  - set daemon umask

6. Network Security
  - `iptables` configuration to deny ssh attempts from outside whitelist, drop all input not related to functional tasks, drop all forwarded packets, conntrack established connections
  - install `fail2ban` and configure to create local ssh jail with `maxretry` set to 3 for root login
  - `/etc/ssh/sshd_config`
    - `PermitRootLogin no` set
    - `AllowUsers` whitelist
    - change the default ssh port
    - `PermitEmptyPasswords no`
    - `UseRoaming no`
    - `IgnoreRhosts yes`
    - `HostbasedAuthentication no`
    - `X11Forwarding no`
    - `Ciphers aes192-ctr,aes256-ctr`

    - `ClientAliveInterval 900`
    - `ClientAliveCountMax 0`
    - `UsePAM yes`
    - `loglevel INFO`
  - `chown root:root /etc/ssh/sshd_config`and `chown 600 /etc/ssh/sshd_config`

7. Prevent evil-maid attacks
  - edit `/etc/modprobe.d/blacklist.conf`
    - add line `blacklist usb_storage`
  - edit `/etc/rc.local`
    - add lines: `modprobe -r usb_storage` and `exit 0`

8. Daemon auditing
  - The following are legacy services that should not be running or installed. Check for these with `apt`, `yum`, and/or `systemctl`
    - Telnet
    - RSH
    - NIS
    - TFTP
    - TALK
  - The following should only be running and installed if nessessary to allow for work performed
  - SFTP
  - DNS
  - LDAP
  - SMB
  - DHCP
  - NFS
  - SNMP

- HTTPD
- `grep umask /etc/init.d/functions` and ensure this is `027` or `022`
- `sudo service xinetd stop; sudo chkconfig xinetd off` if xinetd is not needed

9. Network Params
   - In `/etc/sysctl.conf` set the following
     - `net.ipv4.ip_forward = 0`
     - `net.ipv4.conf.default.send_redirects = 0`
     - `net.ipv4.conf.all.accept_redirects = 0`
     - `net.ipv4.conf.default.accept_redirects = 0`
     - `net.ipv4.icmp_ignore_bogus_error_responses = 1`
     - `net.ipv4.tcp_syncookies = 1`
     - `sysctl -p`
10. Config hardening
    - `chmod 644 /etc/passwd`
    - `chown root:root /etc/passwd`
    - `chmod 644 /etc/group`
    - Drupal+Tripal+Chado stack research `chmod root:root /etc/group`
    - `chmod 600 /etc/shadow`
    - `chown root:root /etc/shadow`
    - `chmod 600 /etc/gshadow`
    - `chown root:root /etc/gshadow`
11. Cron Hardening
    - `chown root:root /etc/anacrontab`
    - `chmod og-rwx /etc/anacrontab`
    - `chown root:root /etc/crontab`
    - `chmod og-rwx /etc/crontab`
    - `chown root:root /etc/cron.hourly`

- `chmod og-rwx /etc/cron.hourly`
- `chown root:root /etc/cron.daily`
- `chmod og-rwx /etc/cron.daily`
- `chown root:root /etc/cron.weekly`
- `chmod og-rwx /etc/cron.weekly`
- `chown root:root /etc/cron.monthly`
- `chmod og-rwx /etc/cron.monthly`
- `chown root:root /etc/cron.d`
- `chmod og-rwx /etc/cron.d`

12. ACL and Password Policy Enforcement
    - Enable SELinux
      - edit `/etc/selinux/config` and set line `SELINUX=enforcing`
    - Enable PAM and config:
      - `/etc/pam.d/other`
      - `/etc/pam.d/common-password`
      - `/etc/pam.d/password-auth`
      - `/etc/pam.d/system-auth`
      - `/etc/pam.d/su`
      - `/etc/security/pwquality.conf`
13. Utilities
    - Install and run `lynis`
      - `cd /opt/`
      - `git clone https://github.com/CISOfy/lynis`
      - `cd lynis; ./lynis audit system`
    - Install and run `rkhunter`
      - `sudo apt install rkhunter` or `yum install rkhunter`
      - `rkhunter -c`

    - `crontab -e` followed by `0 3 * * * /usr/sbin/rkhunter -c 2>&1`
  - install and run `clamav`
    - `sudo apt-get install clamav` or `yum install clamav`
    - `freshclam`
    - `clamscan -r -i /`
  - install `ntp` server
    - `sudo apt-get install ntp`
    - edit `/etc/ntp.conf`such that server is set to
    - `systemctl enable ntp && systemctl restart ntp`
    - `timedatectl set-ntp off`
  - install `syslog` daemon
14. Remove uncommon protocols and filesystems from kernel
15. adding hidepid to /proc
  - `mount -o remount,rw,hidepid=2 /proc`
  - `echo 1 > /proc/sys/kernel.sysrq`
16. banners `/etc/issue.net`

NOTICE TO USERS

Attached here should be included a banner decided as proper forewarning by NDSU ITC as a warning for unauthorized users to log off immediately if they have not recieved prior authorization

---

17. Make su unavailable to normal users
  - `sudo dpkg-statoverride --update --add root sudo 4750 /bin/su`

E Mediawiki Docker Container Setup

To begin setup, install docker. On Centos8, this is: `dnf -y install docker-ce --nobest`

configure firewall so docker containers can forward packets to each other without issues:

```
firewall-cmd --permanant --zone=public --add-rich-rule='rule family=ipv4 source
    address=172.27.0.0/16 accept'
firewall-cmd --reload
sysctl net.bridge.bridge-nf-call-iptables=0
sysctl net.bridge.bridge-nf-call-iptables=0
sysctl net.bridge.bridge-nf-call-arptables=0
sysctl net.bridge.bridge-nf-call-ip6tables=0
```

Pull the needed docker images and run them:

```
docker run --name=mariadb -p 3306:3306 -v wiki-volume:/var/lib/mysql \
-e MARIADB_ROOT_PASSWORD=CHANGETHISPASS123 -d mariadb
docker run --name=wikiname -p 80:80 -p 443:443 --link maria-db \
-d simplyintricate/mediawiki
```

copy over config files, including the system .key file you generate with a certificate signing authority - otherwise just use letsencrypt

```
docker cp ./bundle.cer wikiname:/etc/nginx/conf.d/
docker cp ./yoursite.key wikiname:/etc/nginx/conf.d/
docker cp ./default.conf wikiname:/etc/nginx/conf.d/
docker restart wikiname
```

Finally, install mediawiki as per normal using the browser, but use the name of the mariadb server container as your SQL server host during installation.

F Acknowledgements

This work used resources of the Center for Computationally Assisted Science and Technology (CCAST) at North Dakota State University.

I would like to extend my sincerest gratitude and acknowledge the contributions of Dr Nonoy Bandillo, Dr Ana Morales-Heilman, Dr Thomas Walk, Dr Md Abdullah Al Bari, Hannah M. Worral, Nick Dusek, and the other members of the research community at NDSU: their guidance, knowledge, and human warmth have made this undergraduate research experience a wonderful one.

This summer and the previous semester, I was fortunate enough to work for North Dakota State University (NDSU) with the Bandillo Lab research group, the Minot North Central Research Extension Center, and with the Breeding Pipeline Database Management team. As an undergraduate researcher, I worked closely with these teams throughout the summer and fall to improve and audit information technology systems used for agronomic phenotype data, and also to research and implement technological improvements for both the pulse crop program specifically and the plant sciences breeding pipeline system in general.

References

[1] Akveo, *ngx-admin - documentation*, 2019.

[2] James Patton Jones Casimir Lesiak Bhroam Mann Bill Nitzberg Tom Proett Judith Utley Albeaus Bayucan Robert L. Henderson, *Portable batch system administrator guide*, Veridian Systems Inc., 2000.

[3] Nonoy Bandillo, Diego Jarquin, Qijian Song, Randall Nelson, Perry Cregan, Jim Specht, and Aaron Lorenz, *A population structure and genome-wide association analysis on the usda soybean germplasm collection*, The Plant Genome **doi: 10.3835/plantgenome2015.04.0024** (201507).

[4] Worral H et al. Bandillo N Stefaniak T, *Registration of 'nd dawn' - large yellow pea.*, J. Plant Regist. **15** (2021), 53–61.

[5] Md Abdullah Al Bari, Ping Zheng, Indalecio Viera, Hannah Worral, Stephen Szwiec, Yu Ma, Dorrie Main, Clarice J. Coyne, Rebecca McGee, and Nonoy Bandillo, *Harnessing genetic diversity in the usda pea (pisum sativum l.) germplasm collection through genomic prediction*, bioRxiv (2021), available at https://www.biorxiv.org/content/early/2021/05/08/2021.05.07.443173.full.pdf.

[6] S. Cain et. al. B.D O'Connor A. Day, *Gmodweb: a web framework for the generic model organism database*, Genome Biology **R102** (2008).

[7] Rex Bernardo, *Prediction of maize single-cross performance using rflps and information from related hybrids*, Crop Science **34** (1994), no. 1, cropsci1994.0011183X003400010003x, available at https://acsess.onlinelibrary.wiley.com/doi/pdf/10.2135/cropsci1994.0011183X003400010003x.

[8] Ron Wehrens Bjørn-Helge Mevik, *Introduction to the pls package*, 2020.

[9] Peter J. Bradbury, Zhiwu Zhang, Dallas E. Kroon, Terry M. Casstevens, Yogesh Ramdoss, and Edward S. Buckler, *TASSEL: software for association mapping of complex traits in diverse samples*, Bioinformatics **23** (200706), no. 19, 2633–2635, available at https://academic.oup.com/bioinformatics/article-pdf/23/19/2633/451862/btm308.pdf.

[10] Paul Brown, *State of the union: npm* (2017).

[11] Software Freedom Conservancy, *git documentation*, 2021.

[12] Cybersecurity and Infrastructure Security Agency, *Food and agriculture sector specific plan*.

[13] Pierre Dagnelie, *Analyse statistique à plusieurs variables*, Gembloux, 1975.

[14] Kenneth Lange David H Alexander John Novembre, *Fast model-based estimation of ancestry in unrelated individuals*, Genome Research **19** (2009), 1655–64.

[15] Peter J. Embi, Courtney Hebert, Gayle Gordillo, Kelly Kelleher, and Philip R. Payne, *Knowledge management and informatics considerations for comparative effectiveness research*, Medical Care **51** (2013), no. Supplement 8Suppl 3.

[16] J. B. Endelman, *Ridge regression and other kernels for genomic selection with r package rrblup*, Plant Genome **4** (2011), 250–255.

[17] Roy Thomas Fielding, *Architectural styles and the design of network-based software architectures*, Publication, 2000.

[18] Center for Internet Security, *Cis centos linux 8 benchmark* (2020).

[19] Free Software Foundation, *Gnu affero general public license*, Free Software Foundation, 2007.

[20] ———, *Gnu coreutils*, Free Software Foundation, 2021.

[21] OpenJS Foundation, *expressjs docs*, 2021.

[22] ———, *mochajs documentation*, 2021.

[23] ———, *nodejs docs*, 2021.

[24] Jerome Friedman, Trevor Hastie, and Robert Tibshirani, *Regularization paths for generalized linear models via coordinate descent*, Journal of Statistical Software **33** (2010), no. 1, 1–22.

[25] Claire Coyne Gayle Volk, *The critical role of international collaborations to improve conservationand utilization of crop collections*, USDA, Fort Collins, Colorado, 2021.

[26] David M. Goodstein, Shengqiang Shu, Russell Howson, Rochak Neupane, Richard D. Hayes, Joni Fazo, Therese Mitros, William Dirks, Uffe Hellsten, Nicholas Putnam, and Daniel S. Rokhsar, *Phytozome: a comparative platform for green plant genomics*, Nucleic Acids Research **40** (201111), no. D1, D1178–D1186, available at https://academic.oup.com/nar/article-pdf/40/D1/D1178/16957607/gkr944.pdf.

[27] Google, *angularjs docs*, 2021.

[28] Hans Kandel Julie Pasche Michael Wunsch Janet Knodel Kenneth Hellevang Gregory Endres Shana Forster, *Field pea production*, NDSU, 2016.

[29] Teketel A. Haile, Taryn Heidecker, Derek Wright, Sandesh Neupane, Larissa Ramsay, Albert Vandenberg, and Kirstin E. Bett, *Genomic selection for lentil breeding: Empirical evidence*, The Plant Genome **13** (2020), no. 1, e20002, available at https://acsess.onlinelibrary.wiley.com/doi/pdf/10.1002/tpg2.20002.

[30] Richard D Hipp, *SQLite*, 2020.

[31] Kevin L Howe, Premanand Achuthan, James Allen, Jamie Allen, Jorge Alvarez-Jarreta, M Ridwan Amode, Irina M Armean, Andrey G Azov, Ruth Bennett, Jyothish Bhai, Konstantinos Billis, Sanjay Boddu, Mehrnaz Charkhchi, Carla Cummins, Luca Da Rin Fioretto, Claire Davidson, Kamalkumar Dodiya, Bilal El Houdaigui, Reham Fatima, Astrid Gall, Carlos Garcia Giron, Tiago Grego, Cristina Guijarro-Clarke, Leanne Haggerty, Anmol Hemrom, Thibaut Hourlier, Osagie G Izuogu, Thomas Juettemann, Vinay Kaikala, Mike Kay, Ilias Lavidas, Tuan Le, Diana Lemos, Jose Gonzalez Martinez, José Carlos Marugán, Thomas Maurel, Aoife C McMahon, Shamika Mohanan, Benjamin Moore, Matthieu Muffato, Denye N Oheh, Dimitrios Paraschas, Anne Parker, Andrew Parton, Irina Prosovetskaia, Manoj P Sakthivel, Ahamed I Abdul Salam, Bianca M Schmitt, Helen Schuilenburg, Dan Sheppard, Emily Steed, Michal Szpak, Marek Szuba, Kieron Taylor, Anja Thormann, Glen Threadgold, Brandon Walts, Andrea Winterbottom, Marc Chakiachvili, Ameya Chaubal, Nishadi De Silva, Bethany Flint, Adam Frankish, Sarah E Hunt, Garth R IIsley, Nick Langridge, Jane E Loveland, Fergal J Martin, Jonathan M Mudge, Joanella Morales, Emily Perry, Magali Ruffier, John Tate, David Thybert, Stephen J Trevanion, Fiona Cunningham, Andrew D Yates, Daniel R Zerbino, and Paul Flicek, *Ensembl 2021*, Nucleic Acids Research **49** (202011), no. D1, D884–D891, available at https://academic.oup.com/nar/article-pdf/49/D1/D884/35364073/gkaa942.pdf.

[32] Andy Liaw and Matthew Wiener, *Classification and regression by randomforest*, R News **2** (2002), no. 3, 18–22.

[33] J.L. Lush, *Animal breeding plans*, Read Books Limited, 2013.

[34] MariaDB, *Mariadb knowledge base*, 2021.

[35] John Marshall, *The variant call format specification*, Glasgow, Scotland, 2021.

[36] MediaWiki, *Mediawiki — mediawiki,* 2020. [Online; accessed 9-May-2021].

[37] Goddard ME Meuwissen TH Hayes BJ, *Prediction of total genetic value using genome-wide dense marker maps*, Genetics **Apr;157(4)** (2001), 1819–29.

[38] Ana Morales and Thomas Walk, *Ndsu breeding database pipeline management*, NDSU, 2021.

[39] NCBI, *pisum sativum (pea) full genome* (2018Mar).

[40] Sandesh Neupane, Rajeev Dhakal, Derek M. Wright, Deny Kumar Shrestha, Bishnu Dhakal, and Kirstin E. Bett, *Strategic identification of new genetic diversity to expand lentil lens culinaris medik. production using nepal as an example*, bioRxiv (2020), available at https://www.biorxiv.org/content/early/2020/08/19/2020.08.18.256420.full.pdf.

[41] National Institute of Standards and Technology, *security and privacy controls for information systems and organizations 2020*, Special Publication **800-53** (2020).

[42] T. Ohno, *Toyota production system: Beyond large-scale production*, Taylor & Francis, 1988.

[43] Louis Papageorgiou, Picasi Eleni, Sofia Raftopoulou, Meropi Mantaiou, Vasileios Megalooikonomou, and Dimitrios Vlachakis, *Genomic big data hitting the storage bottleneck*, EMBnet.journal **24** (2018), no. 0, 910.

[44] Paulino Perez and Gustavo de los Campos, *Genome-wide regression and prediction with the bglr statistical package*, Genetics **198** (2014), no. 2, 483–495.

[45] Hans-Peter Piepho, J. Möhring, AE Melchinger, and A. Büchse, *Blup for phenotypic selection in plant breeding and variety testing*, Euphytica **161** (200805), 209–228.

[46] CentOS Project, *Centos 8 documentation*, 2020.

[47] Docker Project, *Docker docs*, 2021.

[48] Openstack Project, *Openstack docs*, 2021.

[49] R Core Team, *R: A language and environment for statistical computing*, R Foundation for Statistical Computing, Vienna, Austria, 2020.

[50] J. Rutkoski, R.P. Singh, J. Huerta-Espino, S. Bhavani, J. Poland, J.L. Jannink, and M.E. Sorrells, *Genetic gain from phenotypic and genomic selection for quantitative resistance to stem rust of wheat*, The Plant Genome **8** (2015), no. 2, plantgenome2014.10.0074, available at https://acsess.onlinelibrary.wiley.com/doi/pdf/10.3835/plantgenome2014.10.0074.

[51] J. E. Stajich, *The bioperl toolkit: Perl modules for the life sciences*, Genome Research **12** (2002), no. 10, 1611–1618.

[52] Bootstrap Team, *bootstrap documentation*, 2021.

[53] Michelle Wallig, *doparallel manual*, 2020.

[54] Kris A. Wetterstrand, *Dna sequencing costs: Data from the nhgri genome sequencing program (gsp)* (2020).