DataGrate Solutions

by

Sean Batt and Nayanchandra Patel

Submitted to
the Faculty of the School of Information Technology
in Partial Fulfillment of the Requirements for
the Degree of Bachelor of Science
in Information Technology

	Sean Batt	April 13th 2020
Signature	Print	Date
	Nayanchandra Patel	April 13th 2020
Signature	Print	Date
Professor James Scott		April 13th 2020
Faculty Adviser		Date

University of Cincinnati
College of
Education, Criminal Justice, and Human Services
April 2020

Prepared by
Sean Batt and Nayanchandra Patel

Students of
University of Cincinnati
College of Education, Criminal Justice, and Human Services
School of Information Technology

April 2020

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

TABLES

FIGURES

ABSTRACT

Enterprise data migration was a common practice amongst businesses in corporate America. Organizations chose to migrate databases for a variety of reasons including trying to reduce costs by moving to cloud-based databases, seeking specific database features and functionality, or their existing systems were simply outdated and unable to keep up with the demands of the business (Alooma 2019). As companies continued to expand and evolve, features such as agile data migrations and integration capabilities became more integral. Company acquisitions and departmental mergers were common catalysts for data migrations. Upon these occurrences, the responsibility had been given to the IT department to migrate and integrate the data into the new environment to allow end users to complete daily functions. DataGrate was conceived to provide data solutions to both the technology professional and the end user. Data functions included on the front end allowed users to easily pull, clean, and organize data for the end user. At the end of the project lifecycle, DataGrate solutions had been developed into a fully functional data migration software with dynamic capabilities and user functionalities, making the product a success for both the DataGrate Solutions team and the end user.

1. INTRODUCTION

1.1. PROBLEM

Enterprise data migration and integration has often resulted in substantial data loss, data corruption, and data incompatibility (Xplenty 2019). To help mitigate data migration problems, many businesses have attempted to perform in house data migration, resulting in poorly executed migration projects. Businesses have failed to properly migrate data for many reasons; some of which include lack of formal data governance, methodology, and tools to support the migration.

1.2. SOLUTION

DataGrate Solutions has provided affordable and effective technology to service business data stores of all kinds. User functionality and features provide compatibility, organization, and data management. Part of this user functionality are simple Structured Query Language (SQL) functions that can be used to help organize the data in a form specific to what the user wants, and an ability to export the data for further analysis. The migration feature of this product allows for data to be transferred with a click of a button. Data can move from MySQL to MySQL or MySQL to PostgreSQL. While out of the box solutions exist, the team found that they became costly to operate in terms of time and money. Such out of the box solutions include, Carbonite, and the service company Talend. What these solutions lacked is the user functionality DataGrate allows for. Easy to use software coupled with end user capabilities makes DataGrate Solutions different from other products of this nature.

1.3. GOALS & METHODOLOGY

The following section, goals and methodology, will simply list what DataGrate Solutions sought to accomplish through project implementation. DataGrate Solutions will utilize the ETL (Extract, Transfer, Load) framework for this project, which will allow for the construction of a data pipeline to perform data migrations from one database to another.

- Data Migration
- Data Integration
- Data Management

Further research and exploration revealed that integrating tableau to visualize the data was not feasible as the cost was too high, and the number of work hours taken to establish the connection between the databases and the visualizer was not corresponding properly to the output. Instead, a reduced scope on the visualization, allowed for a frontend capability built into the interface when the data migration had been initiated.

1.4. OVERVIEW

The following sections of this report provide further detail regarding DataGrate Solutions. Descriptions of Problem, Solutions, User Profiles and technical diagrams followed by a series of illustrations will provide insight to DataGrate Solutions' planning, scheduling, and website.

2. DISCUSSION

2.1. PROJECT CONCEPT

DataGrate Solutions was developed for enterprise level data transfers and migrations, with unique front-end functionalities. This product was conceived to provide data solutions to both the technology professional and the end user. DataGrate Solutions includes a uniquely coded transfer pipeline, which allows for capabilities not currently found in any other products of similar purpose. The DataGrate Solutions team had taken particular interest in data, data analytics, and migration capabilities which provoked the idea to create a data migration solution product.

2.2. DESIGN OBJECTIVES

The following section further elaborates on DataGrate Solutions goals and objectives. The goals and objectives provide a brief analysis of what DataGrate Solutions sought to accomplish through project implementation.

Data Migration

- Transfer of data from one location to another
- Dynamic database capabilities
- *Please Refer to Figure 6: Data Migration Process*

Data Integration

- The combination of data sets from multiple different sources into a single, cohesive unit of data.
- This will be accomplished once the data transfer has been initiated. The pipeline will have to load the data into the second database with new schemas and data infrastructure.

Data Management

- SQL functions to pull and organize data
- Accomplished by creating a front-end webpage which have actionable buttons, which are mapped to the backend that execute upon uniquely written SQL queries to query the database and call for the data that was asked for on the front end.
- These buttons were developed to deliver a simple method for the end user to pull data. This functionality is not meant for complex analytics. Along with the data that is generated when clicking on an SQL button, the user will also be provided an option to export the data on the UI, which the user can then perform further analysis, and manipulate as necessary.

Initial conception of the DataGrate Solutions product included a major component that was not able to be developed as a part of the final solution. Initially, DataGrate Solutions included a compression software component that would compress data as it was being transferred to the endpoint. The DataGrate Solutions team deemed it unnecessary and problematic to move forward with this component of the project. The compression software programming would have been to complex to incorporate for our product and would have hindered the ability for the team to meet necessary goals and deadlines.

2.3. USER PROFILE

The User Profile provides detail to various projected related areas. User Profile provides information regarding targeted user groups, software, interface and experience. A breakdown of user functionality and software navigation is provided as well.

Potential Users

While there are many potential users of our software service, DataGrate Solutions aimed to target two groups of people. The first group was IT Professionals; this could include people ranging from database admins to networking professionals. The second group is end users. This may include anyone in the enterprise that would consume or use the data such as data and business analysts.

Software, Interface and Related Experience

Depending on what the user is using the product for, users need a varying range of capabilities to use this software. Starting with IT professionals, if the goal is to perform the migration, then the user has to to have knowledge pertaining to network infrastructure, database architecture, storage components, as well as SQL knowledge. That is the technical knowledge needed; additional knowledge would include business requirements, and an understanding on the use of the data being migrated. This will not only ensure a smooth migration process, but also make it easy for the professional who will be handling the data being migrated.

The end user applications of this data will have to have basic SQL knowledge, and an understanding on how table schemas operate. This will be imperative for any requirement of using this data, and DataGrate Solutions team is working under the assumption that the end user will have adequate data/business analytical skills to make use of the frontend capabilities.

Task Experience

Initial opening of the software brings users the homepage, which contains a simple informative page on what DataGrate solutions is, and what it has to offer. Upon further exploration into the program, users have two options: navigation to the database, or navigation to the migration portal. If navigated to the database portion of the program, users have the ability to insert the table name, and pull data accordingly. Several pre-made SQL function buttons are on the user interface (UI) for quick pulls. Once the user hits one of the buttons, then the data will appear in an organized fashion along with the SQL that was used to retrieve that data.

If the user navigates to the migration portal, the user will then need to input the necessary information, such as what data from which database will they be migrating and where it will go. Once the necessary information is input, then the user can start with the data transfer process.

Frequency of Use

This software in prototypical phase was intended to be used when a migration needs to happen from one database to another database of the same configuration. (e.g MySQL to MySQL). In the later stages it can be used when different databases are involved (e.g MySQL to PostgreSQL). In its prototype phase it can also be used when trying to pull/analyze the data stored in the receiving database, which is fully dependent on the user of the data.

Key Interface Design Requirements that the Profile Suggests

- Intuitive UI for both front and backend use cases
- Responsive design components for the PC environment
- Design layout is clean and simple
- Functionality represents a high level of user friendliness

2.4. USE CASE DIAGRAM

Figure 1: Use Case Diagram, visually describes specific groups or individuals and how they interacted with DataGrate Solutions.

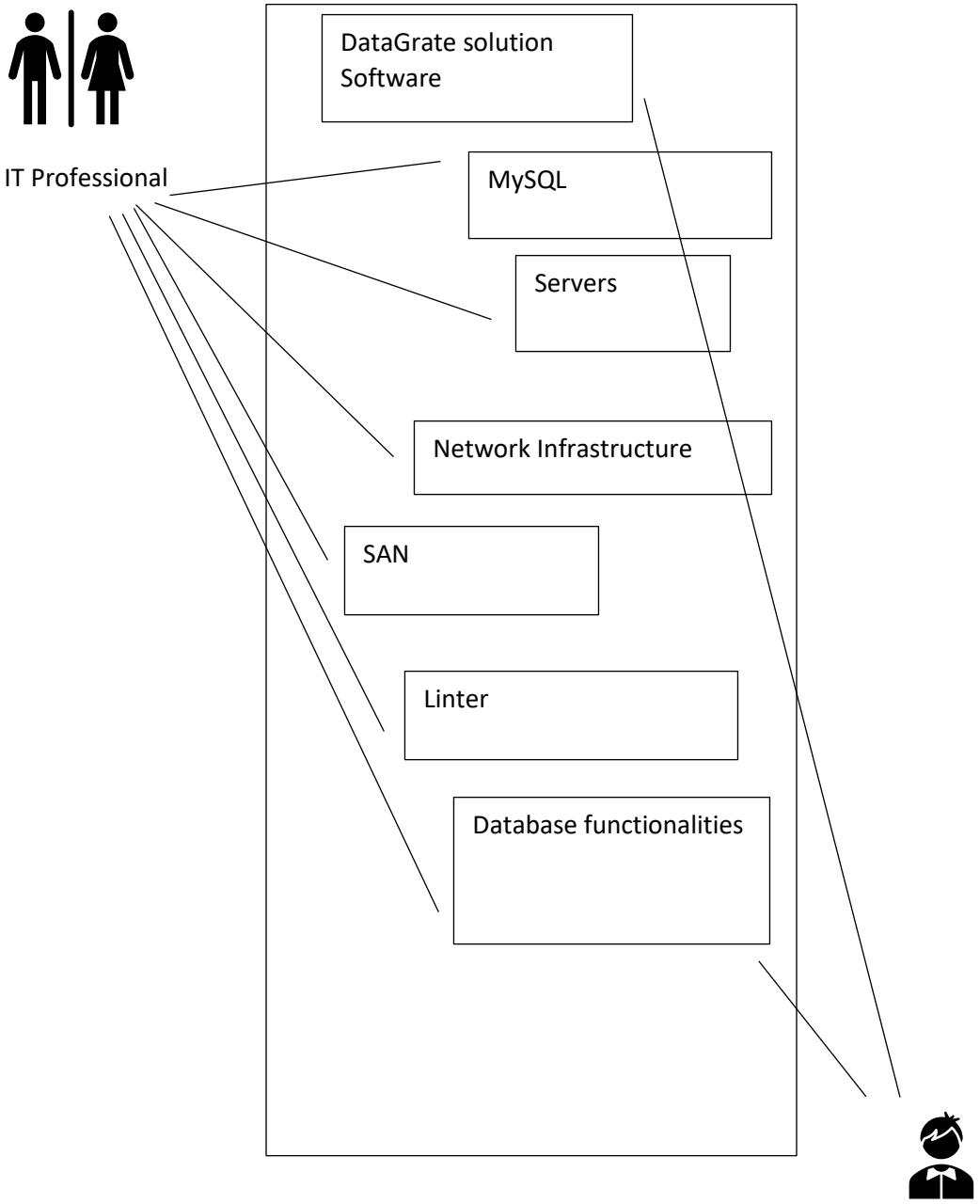

Figure 1: Use Case Diagram

2.5. TECHNICAL APPROACH

The following section describes different programs and software used to establish the technical architecture and the process in which the web application functions in its current state. The frontend provides the capability to pull data from a database that has been configured with MySQL. The following diagrams show the process on how the front end would communicate with the backend, along with the technology used for each end.

Figure 2: Technology Stack, visually represents the current technology stack leveraged for DataGrate Solutions web application.

Area of DataGrate	Tools and Technology Used
Hosted On	Docker Containers
FrontEnd	VueJS Vuetify Tailwind CSS Images designed with Adobe Illustrator
BackEnd	Express
Databases	MySQL PostgreSQL

Technical Stack Overview

- Using styling frameworks such as Vuetify and Tailwind, we have built a sleek UI
- Leveraging a JS framework (VueJS) we have built a functional and responsive frontend
- An operational backend was created using a server-side language (Express.JS) which calls from our database
- Two functional databases

Figure 2: Technology Stack

Figure 3: Flow Diagram, shows the process in which data is called from a database back to the frontend after a user initiates a data call using a simple button on the interface.

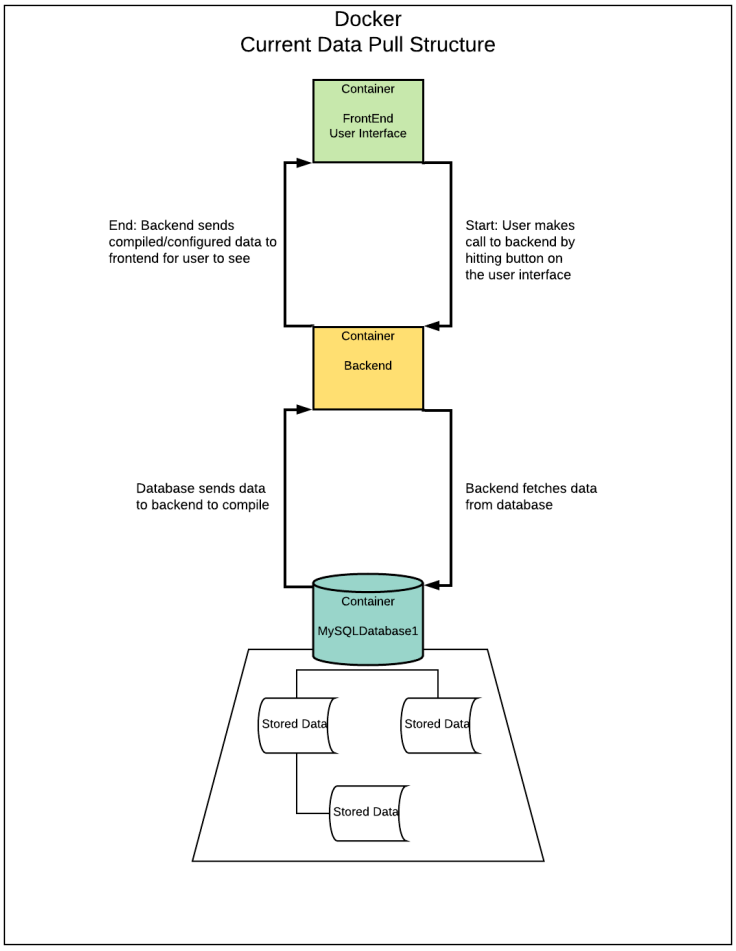

Figure 3: Flow Diagram

The visual below, *Figure 4: Database Interface Visual*, represents the interface a user would see after selecting the “database” tab. When the user has reached the “database” page, the user would then click on one of several buttons to retrieve a specific set or piece of data.

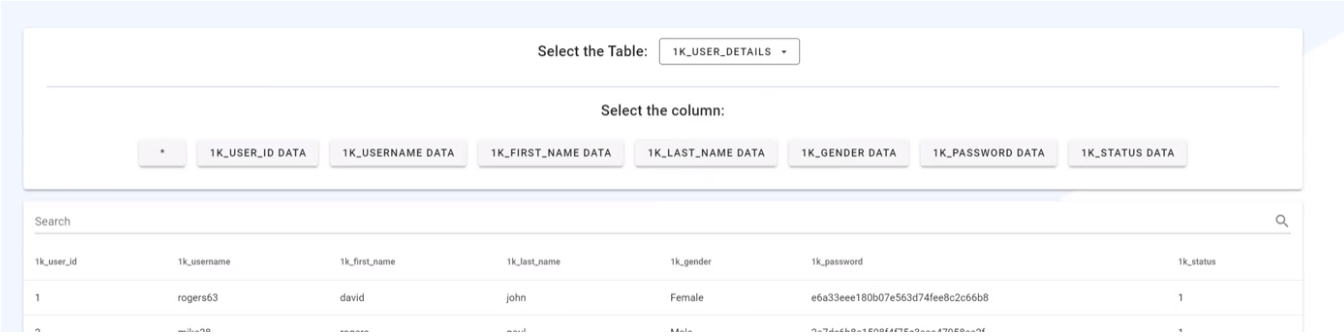

Figure 4: Database Interface Visual

Figure 5: Frontend Data Organization, descibes how data is organized on the frontend after the user has selected the desired button on the database page.

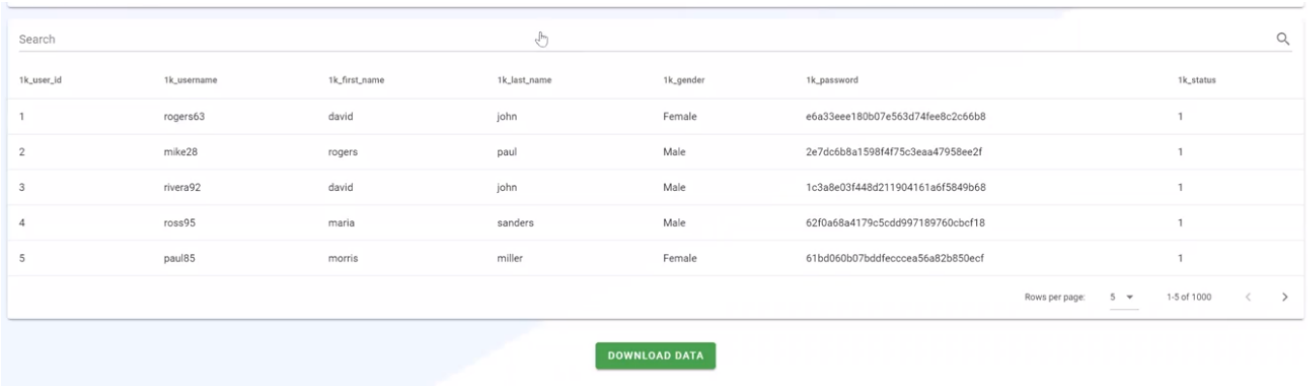

Figure 5: Frontend Data Organization

Figure 6: Communications Code, represents the code that allows for the front end and backend to communicate with each other, as well as how the backend communicates with the database

Figure 6: Communication Code

Figure 7: Data Migration Process, is a visual representation of what the data migration process looks like in completed form. Notice, the front-end shows data from the second database, since this is where the data will be moved.

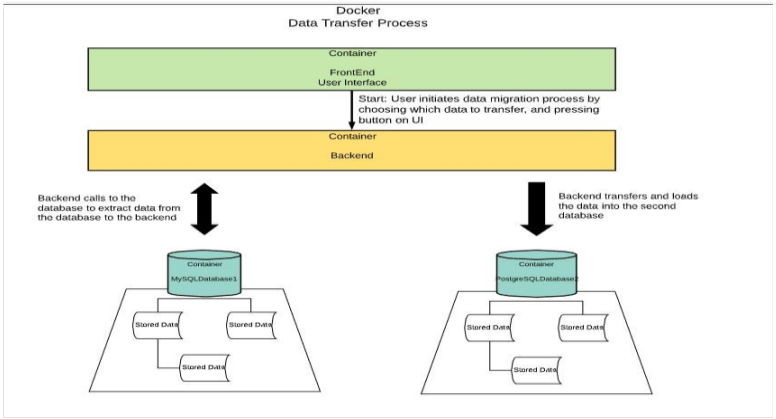

Figure 7: Data Migration Process

2.6. TESTING OVERVIEW

DataGrate used a dynamic methodology for testing. Our testing included a manual quality assurance style approach; which entails members of the team ensuring each portion of the project functioned as designed by using the web page interface, data migration pipeline, and extracting data from the databases as selected by the user with test data. Using this same approach, a third party, unfamiliar with the code and design of the project, also attempted to use each function of the project in the same manner to ensure user friendliness. DataGrate's project solutions was analyzed on performance; ensuring access, speed and capability are up to par. Performance testing included stress and load testing, as well as scalability testing. This ensured the software would be able to sustain various sizes of databases and data loads during the migration and extraction processes.

METHODOLOGY

DataGrate's methodology of testing the program included a three-part series of tests to ensure that the maximum number of issues were detected and fixed before the final version of the web application. The parts are outlined below:

Manual Quality Assurance Testing

The developer ran functionality and quality assurance tests as each individual feature of the web application was built and pushed out. This ensured that each feature was operational and working without failure before the final version of the web application was developed.

Test Case Testing

Both members of the team ran the test cases and recorded the results on prepared test case documentation, to ensure the objectives outlined above are met.

Third Party Testing

Finally, a third person tested the web application. An individual unfamiliar with the code explored our web application. This is often identified as adhoc testing, which simulates real world use of the application. DataGrate Solutions team observed and took notes as necessary to document any failures that occurred to remediate during development stages of the web application.

TESTING SCOPE

The testing of DataGrate Solutions ensured the functionality, performance, and usability of different pieces of the software.

Testing Scope Breakdown

Three main points of testing were conducted to ensure functionality, performance and usability as noted above. The migration of data from point A to point B, extracting data from desired database, and the testing of front-end interface.

Data Migration Testing

In testing the migration of data, Team Fifty-One selected an IP address destination, an IP address source, database type and a set of test data and initiated the migration of desired data using the front-end UI, or webpage. The webpage interface contains a series of drop-down menus and entry points to input desired values and parameters per user.

Data Extraction

After testing the migration of data from point A to be point B, Team Fifty-One focused on the capability to extract data from a database. The user had the option to extract and input data from same type databases, as well extract and input data into different types of databases. For example, the user had the option to select the extraction of data from a SQL database, and input the data into an additional SQL

database. Similarly, the user had the ability to extract data from an SQL database and input the same data into a different database such as an Oracle database. This tested the dynamic capability of the software.

Performance and Usability Testing

Lastly, Team Fifty-One tested the front-end UI to ensure functionality, speed and performance, and user friendliness. The testing included the visibility and overall presentation of our webpage, the ability to navigate the web site efficiently, the ability to locate functions efficiently, and the functionality of each part of the site including but not limited to SQL functions, database selection menu, database destination and source IP address entry bar, and user creation menu which included username and password entry.

OBJECTIVES

Our testing plan was created and executed in relation to the following objectives:

1. The individual web application features must function at the state at which there are no failures on the feature for the end user.
2. Each feature should be functional for each unique group of end users. IT professionals, and database users.
3. Any bugs that surface during testing are documented and later revisited to remediate during the development of this program.
4. None of the bugs that were surfaced in the objective above are still remaining before the IT Expo or are labeled as such in the documentation.
5. Each section of the testing plan must be agreed to a “passing” grade by each member of DataGrate Solutions before the IT expo.

The objectives outlined above test the performance and overall capability of DateGrate Solutions software. Through this process, Team Fifty-One was able to identify defects and errors in the software code, ultimately affecting overall performance. Team Fifty-One was also able to identify new methods

to creating a more efficient migration pipeline, as well as increase user friendliness through side by side database viewing, dynamic database pulling, and a cleaner interface.

While in the process of testing performance and identifying defects and errors, Team Fifty-One recognized portions of the project and software that were working exceptionally well. Creating SQL functions for the purpose of pulling pieces of data is just one an example of Team Fifty-One accomplishments.

TESTING PROCEDURE

To conduct the tests that are outlined above the following needs to be established:

- Have a set of pass / fail conditions
- Create a table to visualize the pass / fail conditions and if the we application meets the criteria to pass

Test Cases

A. Open DataGrate Solutions Web Application
    a. Steps:
        - Enter DataGrate Solutions URL
    b. Expected Outcome:
        - DataGrate Solutions loads and brings up homepage
        - Page fits the screen its loaded on
B. Homepage test
    a. Steps:
        - Scroll up and down page
        - Click on all links and buttons

- Use Navbar

b. Expected Outcome:

- Homepage loads correctly on user device
- All links / buttons redirect and work as intended to do so

C. Database Page Test

a. Steps:

- Use dropdown menus to select databases
- Use dropdown menus to select columns to pull data from
- Click on execute button

b. Expected Outcome:

- Correct database is filled in
- Correct columns are showing for options
- Data successfully pull on execute

D. Migrate Page Test

a. Steps:

- Use dropdown menus to select source database
- Fill in form for destination database
- Test execute (will come later once migration pipeline is developed)

b. Expected Outcome:

- Database chosen persists along with other actions
- Data migrates successfully upon execution on UI

Pass/Fail Conditions

It is expected of DataGrate Solutions to pass all of these tests in order to meet the expectations of the team members and end users in order to push the final version of the solution before the Tech Expo. Everything on the frontend UI functions intended. All the buttons, links, databases, informational pages should redirect and display the correct information for the tests to pass. If any of the outlined conditions are not executed as intended, that case will be documented and analyzed for a further detailed investigation.

Test Log

Table 1: Test Log, provides detail into which tests DataGrate Solutions ran during the development phase of the product

Condition	Pass	Fail
Home Page	☑	
1. Home page launches upon URL execution	☑	
2. Scroll works as inteded	☑	
3. Links redirect to correct pages	☑	
4. All Buttons are functional	☑	
Database Page	☑	
1. Dropdown menus are functional	☑	
2. Dropdown menus are showing correct info	☑	
3. Execute functions as intended	☑	
4. Dropdown menus showing correct column names	☑	
5. There is persistence after db selections are made	☑	
Migrate Page	☑	
1. Dropdown menus are functional	☑	
2. Dropdown menus are showing correct info	☑	
3. Migrate button is functional	☑	
4. Correct data is copied over to the destination	☑	

TESTING LEARNING PROCESS

Throughout the development process Team Fifty-One learned there were more issues than we initially anticipated. We learned early on to feature tests as we were developing in order to match our timeline as necessary. With so many aspects for the user to explore on the UI it was imperative that each function worked as intended because some functions are reliant upon another function to work properly. As Team-Fifty-One continued to develop the project, testing for bugs was conducted to prevent any issues related to functionality when the project lifecycle concluded. Furthermore, Team Fifty-One learned that having a third-party test what was being created to be very important. If the solution did not meet user needs, it would have been necessary to make changes in order to meet those needs. The third-party testing process helped us with error and bug detection as well as enhance UI and UX development.

2.7. PROJECT COST AND BUDGET ANALYSIS

Table 2: Cost and Budget Analysis Table, provides detail to what costs DataGrate Solutions accrued throughout the project lifecycle.

Item	Cost	Purpose	Description
Hardware	$0	Host to data and software, communication	Servers, laptop computers, printers, mice, necessary accessories
Software Operating Systems Programs	$0	Data transfer and organization, coding and scripting, documentation, marketing	SQL, Visio, Google Docs, Java, Microsoft Office, Brackets, Tableau, Python, PHP, Microsoft Server 2016, Sandbox
Wages	$0	Fulfill employee living standard	N/A graduating requirement

Table 2: Project Cost and Budget Analysis Table

2.8. PROJECT SCHEDULE (GANTT)

Figure 8: Gantt Chart, includes both Fall and Spring Semesters' deliverables.

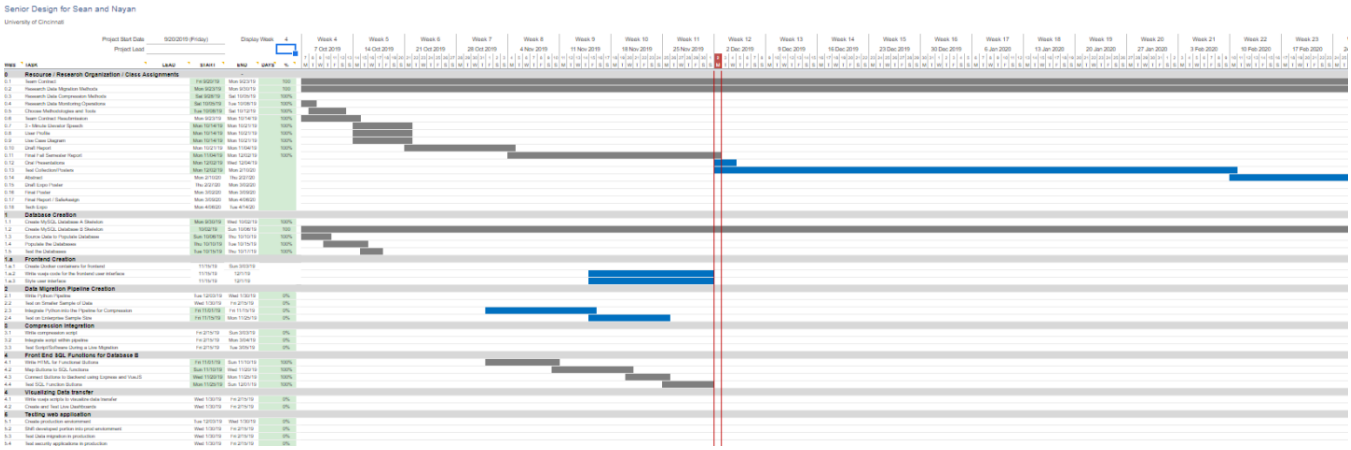

Figure 8: The Gantt Chart

2.9. PROBLEMS ANALYSIS

DataGrate had run into several problems while developing this product. The first of which was scope management. During the start of this project, the scope was quite large, with many features to be developed. This problem was mitigated by discussing which features were the most important vs features that were just "nice to have". Once the team was able to identify that, the scope was cut down to a much more manageable size. Some features that were cut out were the compression during migration feature, and the ability to visualize the transfer on Tableau. While these features would've been great, it was out of the team's ability to execute on those ideas.

2.10. DISCUSSION FOR THE FUTURE

If DataGrate had to start all over again, the team would've conducted the research much quicker and began development in a timely manner, as well as not start with as large of a scope. While that problem was mitigated, it did feel as if the team was on a time crunch the whole way through. If the team had

more time, we would've leveraged it by adding in more database capabilities rather than just the two DataGrate had used by completion.

Some suggestions the team received from others was to chunk the data during the migration process. While this feature would be great to have for large transfers, the team ran out of time and expertise to execute on the idea.

As of now, DataGrate Solutions is planned to be used as an internal tool, but if interest is peaked outside of the team, the team is more than happy to discuss that with additional stakeholders.

3. CONCLUSION

3.1. CONCLUDING DISCUSSION

Fall 2019

DataGrate Solutions accomplished several tasks for the duration of the fall semester. On the administrative side the following items were completed:

- Team Contract
- Team Abstract
- User Profiles and Use Cases
- Our Elevator Speech

During the development phase of DataGrate Solutions, the following tasks were accomplished:

- Research best practices on working with data and data infrastructure
- Compile methods to create the frontend and backends of the program
- Github for commits and versioning
- Create Vuejs frontend and map unique functionalities to frontend
- Create Express backend
- Create 2 MySQL databases
- Host all technical components on Docker
- Create and design informational pages on what DataGrate Solutions is

While developing and completing the tasks outlined above, DataGrate ran into some blockers. As a two person company, everyone was responsible for all parts of the project, this put us into a time crunch, as not all parties are experts in the development fields. When creating the front and backend, much research was done, which consumed significant time. Many things were learned along the way. The company understands how to setup containers and host different portions of the web application on

those containers. How those containers communicated with each other was also a new concept that was learned. Without that, the entire project would have been greatly impacted. The scope of the project was another blocker, it started out to be a very large scope, but as the program was developed the scope was managed and brought down to a manageable size.

Spring 2020

During the Spring Semester, the DataGrate Solutions team completed several tasks and made several improvements to the product. On the administrative side, the following items were completed:

- Final Abstract
- Tech Expo Poster
- Testing Report
- Final Presentation
- Final Report

The development side of the project has seen the following updates:

- Dynamic Database capabilities (MySQL -> PostgreSQL)
- Dynamic data pulling functionalities on the front end
- Increased migration capabilities (1 million rows)

DataGrate worked thoroughly on the development of the migration portion of the project. The foundation for the company was well established during the Fall Semester, allowing for only a few adjustments and updates that needed to be made to the final product during the Spring Semester. A sleek, and responsive frontend has been iterated upon, along with a fast-acting backend. This allowed for a better user experience that ultimately helped with the end goal of providing a simple migration and analytical tool.

Team members were able to enhance coding and development skills, further knowledge regarding software platforms and programs such as Docker and Vue, and fully understand the scope of a data migration tool. Since the completion of the Fall Semester, DataGrate Solutions has developed the migration solution to include transfer of data between different databases, making the product dynamic. The interface has been cleaned and now includes several components to adhere to user friendliness. The team was able to enhance the data migration capabilities by allowing for the migration of one million rows of data.

3.2. REFERENCES

Bibliography

Alley, G. (n.d.). Database Migration Challenges. Retrieved from https://www.alooma.com /blog/database-migration-challenges

Smallcombe, M. (2019, July 25). Data Migration Simplified: Understanding the Use Cases, Challenges, and Processes of Data Migration. Retrieved from https://www.xplenty.com/blog/data-migration-simplified-understanding-the-use-cases-challenges-and-processes-of-data-migration/

4. APPENDIXES

4.1. APPENDIX A– TECHNICAL TERMS AND ACRONYM LIST

1. Structured Query Language (SQL) - programming language used for storing, manipulating and retrieving data in relation to databases
2. PostgreSQL – database management system emphasizing extensibility and SQL compliance
3. Docker – Set of platform as service products that uses OS-level virtualization
4. User interface (UI) – means by which user and computer interact
5. Data Migration – The act of moving data from one area to another
6. Data Integration – The act of combining different data from different areas into one place of storage
7. Extract, Transfer, Loading (ETL) - extract, transfer and loading data and what this means (python coded data pipeline)

4.2. APPENDIX B – WBS AND WBS (TABULAR)

WBS

Figure 9: WBS, details DataGrate Solutions work components and related necessary tasks.

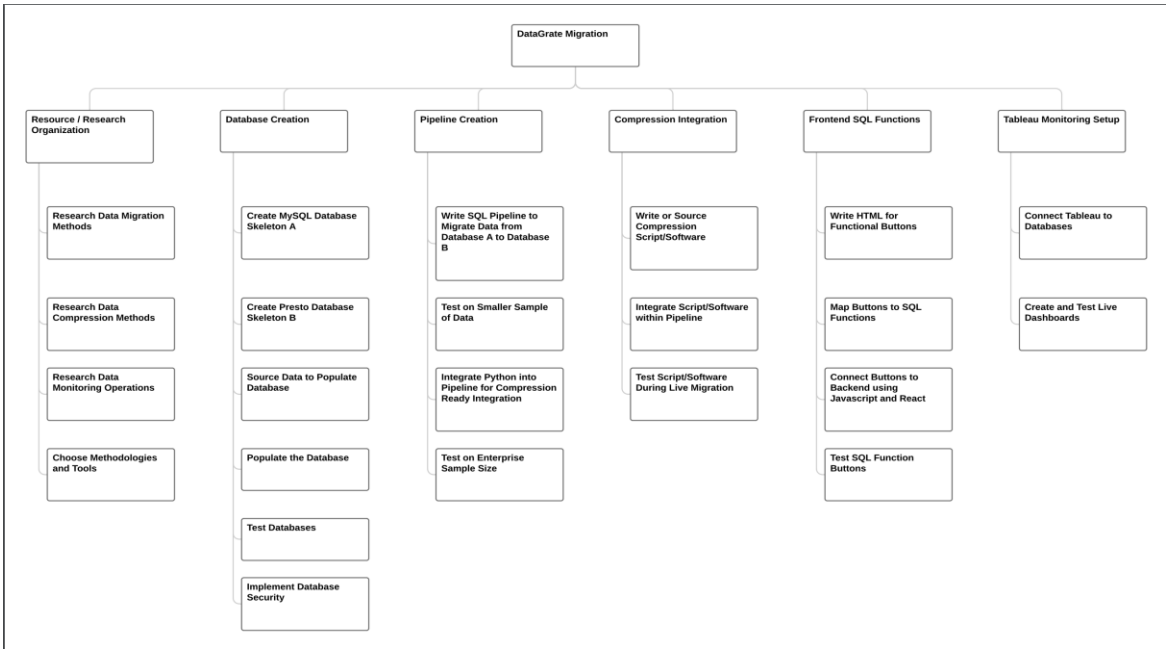

Figure 9: WBS

WBS (TABULAR)

Table 3: *WBS (Tabular)*, provides detail and description to project related work and tasks in a tabular version of the work breakdown structure.

Project	Tasks	Sub Tasks
DataGrate Migration Project	1. Resource / Research Organization	Research data migration methods Research monitoring operations methods Choose methods / tools for project
	2. Database Creation	Create database skeletons (2) Source data to populate databases Populate the databases Test the databases Implement security
	3. Pipeline Creation	Write SQL pipeline Test on smaller sample of data Test on Enterprise Sample Size Visualize data transfer on frontend
	4. Frontend SQL Functions	Write HTML for functional buttons Map buttons to HTML functions Connect buttons to backend through JS or React Test SQL buttons

Table 3: WBS (Tabular)

4.3. APPENDIX C – DATAGRATE SOLUTIONS WEBSITE

Figure 10: Homepage, explains the company and has an email sign up page for those who would like more information on how to use our product.

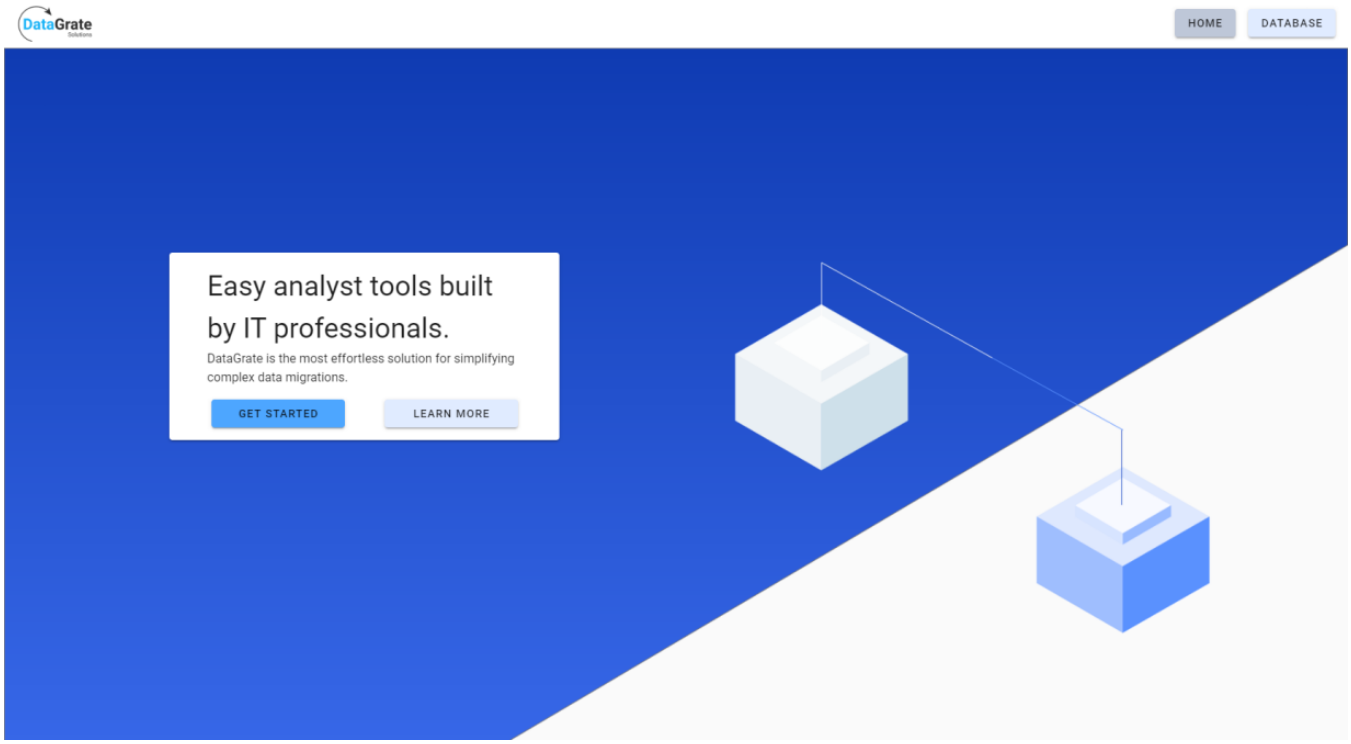

Figure 10: Homepage

Figure 11: About Us Page, is a screenshot of the DataGrate Solutions About us Page – The About us Page is a representation of Team Fifty-One and some descriptions about the founders of the company. Also included is how this idea was formed and provides detail behind the originality of this project.

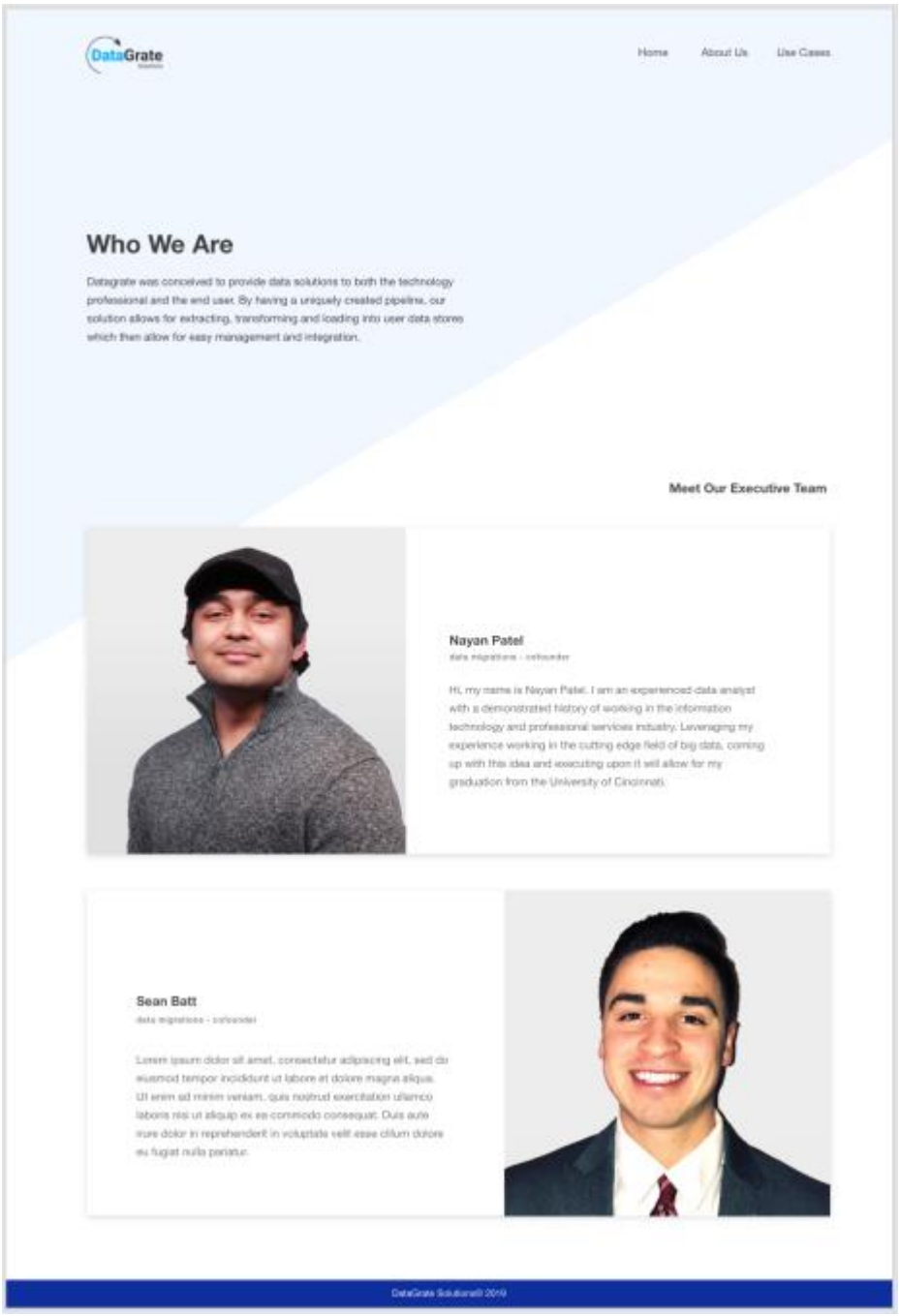

Figure 11: About us Page

4.4 APPENDIX D – TECH EXPO POSTER

Figure 12: Tech Expo Poster, is a combined textual and visual representation of the project. This includes relevant technology, key product features and description of what the solution accomplished.

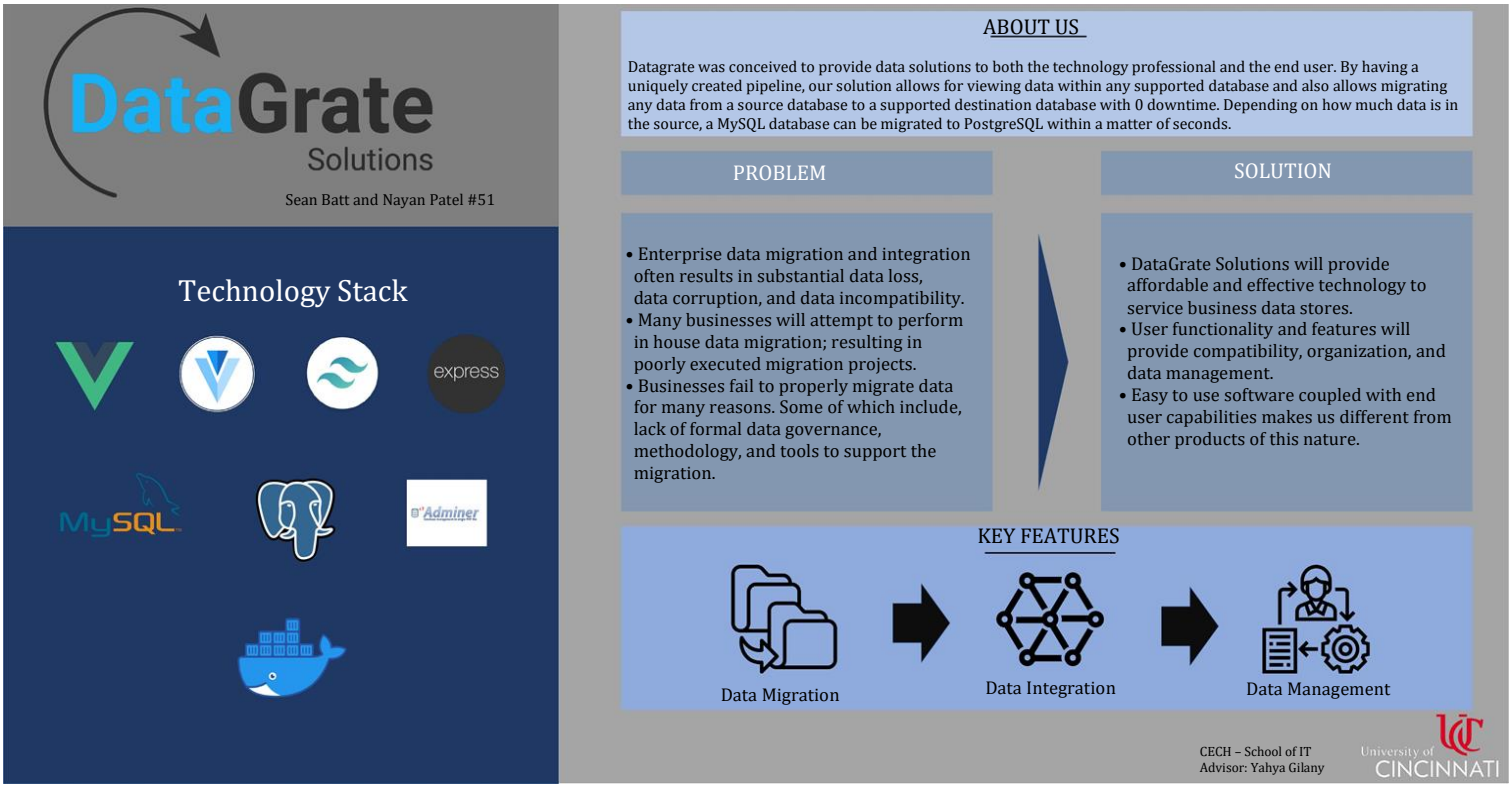

Figure 12: Tech Expo Poster