Designing a Replicable Data Infrastructure for Education Research

Jordan Machita
School of Data Science
University of Virginia
Charlottesville, USA
jm8ux@virginia.edu

Taylor Rohrich
School of Data Science
University of Virginia
Charlottesville, USA
trr2as@virginia.edu

Yusheng Jiang
School of Data Science
University of Virginia
Charlottesville, USA
yj3tp@virginia.edu

Yiran Zheng
School of Data Science
University of Virginia
Charlottesville, USA
yz3qw@virginia.edu

***Abstract*—The field of education research suffers from a lack of replication of existing research studies. SERA (The Special Education Research Accelerator) is a proposed crowdsourcing platform being developed by a research team at the University of Virginia's School of Education that intends to help provide a solution by enabling large-scale replication of research studies in special education. In this paper, we present our design and implementation of a cloud-based data pipeline for a research study that could serve as a model for SERA. Cloud-based design considerations include: financial cost, technical feasibility, security concerns, automation capabilities, reproducibility, and scalability [1] [17]. We have designed an architectural framework that practitioners in education research can use to host their studies in the cloud and take advantage of automation, reproducibility, transparency, and accessibility. Implementation of our platform design includes automating the data extraction and cleaning, populating the database, and performing analytics and tracking. Additionally, the project includes the development of a web-facing API for researchers to query the database with no SQL knowledge necessary as well as a web-facing dashboard to present select information and metrics to the applied research team. Our data pipeline is hosted on Amazon Web Services (AWS), which provides functionality for automation, storage, database hosting, and APIs. We present this architecture to demonstrate how data could flow through the pipeline of SERA to achieve the goals of large-scale replication research.**

***Index Terms*—data architecture, design, cloud, data pipeline, education, AWS, replicability**

I. Introduction

Replication is the repetition of a research experiment to verify or disconfirm the results of the original study [9]. Replication is needed to substantiate existing research by demonstrating that the original findings were stable enough to be discovered more than once [5]. Despite the significance of replication in research, the field of education, among other social sciences, suffers from a dearth of replicated studies [9]. This lack of systematic replication means that few studies have substantial enough evidence to definitively determine effectiveness for different groups, across different contexts and settings and different research designs. [18]. This is a particularly prominent problem in special education research due to small sample sizes that result from special education students making up only a small proportion of the overall student population in any given school or district [18]. The need for replication studies and larger sample sizes in special education research has prompted a research team out of the University of Virginia and led by Vivian Wong to begin the development of the Special Education Research Accelerator (SERA). SERA will be a platform that will leverage crowdsourcing to conduct large-scale replication studies in special education [18]. In this paper, we propose a design and implementation of the data pipeline that represents just one component of SERA. The purpose of the architecture we present, and more largely SERA, is to provide a platform that can be used to conduct replications of research studies in order to combat the replication crisis in education, and the social sciences in general. We will compare different platforms to use to host such an infrastructure by considering financial cost, technical feasibility, security concerns, automation capabilities, reproducibility, and scalability [1] [17]. The detailed comparison will be introduced in the methodology section.

II. Background

A research study is replicable if independent studies are able to produce the same causal effect demonstrated in the original study within the limits of sampling error [19]. Replication is needed to provide evidence that the findings of a study are not the result of error, bias, or chance [3]. Moreover, replications help to more precisely determine the size of the effects found in studies and the extent to which they generalize across contexts [9]. However, there are stringent replication assumptions that must be met according to the causal replication framework presented by Wong, Steiner, and Anglin [19]. Given the requirements of a replication study, one aspect that can aid in the ease of conducting replication is ensuring that the processes used when carrying out original studies, including data collection and research procedures, are transferable and replicable. Given the importance of replication in research, Mackel and Plucker conducted a study to determine the prominence of replication studies in the top 100 education journals and found that out of 164,589 published studies, only 221 were replications [9]. This is an overall replication rate of 0.13%. Two similar studies were conducted to determine the replication rate specifically for special education and found it to be 0.5% [10] and 0.4% [8]. This scarcity of replication extends to other domains in the social sciences, such as psychology, which has a replication

rate of 1.07% [11]. One reason for the scarcity of replication in the social sciences is that researchers may view conducting replications as lacking prestige, originality, or excitement [9]. This is in part caused by social science fields and publishers valuing novelty over replication [6]. Although journals tend to publish only a small amount of replication research, this proportion has been increasing in recent years [11]. Another reason for the scarcity of replication is that in order to replicate a study, researchers need information regarding the research design and the decisions related to the collection, analysis, and reporting of data and procedures. Therefore, issues of transparency and accessibility are barriers for efficiency in the replication of research.

While the fields of both education and psychology have similarly suffered from this replication crisis, an initiative within psychology has been developed in response to the lack of replication. Namely, this is the Physiological Science Accelerator (Psych Accelerator), a large-scale crowdsourcing platform used to conduct research studies in psychology [13]. Crowdsourced research refers to large-scale collaboration on a research project across multiple sites or laboratories [13]. The distributed nature of crowdsourced research requires large-scale data collection and processing, which makes the efficiency and standardization of procedures, such as with an automated data pipeline as we propose, all the more critical. The Psych Accelerator has been successful in developing the infrastructure and processes needed to conduct high-quality crowdsourced replication studies in psychology [2]. The Special Education Research Accelerator is modeled after the Psych Accelerator and is being designed to implement a similar large-scale platform for crowdsourcing research in special education [18].

A core component of SERA will be a data infrastructure adept to handle the inflow of data from partner sites and to store this data in an accessible and secure manner [18]. In this paper, we present the design and implementation of such an architecture for an existing study that could be used as a model to construct the architecture of the data pipeline for SERA. We evaluate our choice of infrastructure according to several key factors: financial cost, technical feasibility, security, automation capabilities, reproducibility, and scalability [17]. Given that education researchers may not be formally trained to assess platforms according to these factors, we walk through a more detailed breakdown of the framework we use to evaluate the platform alternatives in the methodology section and we provide guidance on how to frame such considerations. The results section presents our final platform decision. In the discussion, we detail the design and implementation of our architecture based on the chosen platform.

III. Methodology

A. Data Domain

In order to construct a data pipeline that can be tailored to various projects to aid in research replication, we used data from a separate research study to pilot the design and implement the framework. This study, coming out of the TeachSIM team at UVA, evaluates the effectiveness of providing pre-service teacher candidates with standardized simulated practice opportunities along with targeted coaching on their instructional skills. The data collected in the study includes numerous surveys taken by the participants before and after simulations and coaching sessions, as well as the transcripts obtained from the actual simulations. Figure 1 shows the generalized workflow for how participants and researchers interact with the data in the study. The participants complete the surveys and simulation sessions, then the research team cleans the survey data and transcribes the simulations before uploading the data to the database, where the whole team can then access this processed data. Additionally, the research team manually updates a participant tracker to record that the participant completed the corresponding surveys and simulator sessions. This process is repeated for numerous surveys and simulator sessions for each participant throughout the life of the study.

While the survey questions are certainly project specific, the use of surveys, often through Qualtrics, is a common practice in studies. This presented us with an opportunity to design a pipeline that could be largely functionally independent of the specific fields in surveys, meaning that with minimal modifications, it could be deployed for an entirely different research study. The other type of data generated in this research study is the transcription of simulator sessions. Throughout this paper and in our project design, we primarily focus on the survey data; however, we will also address the transcriptions.

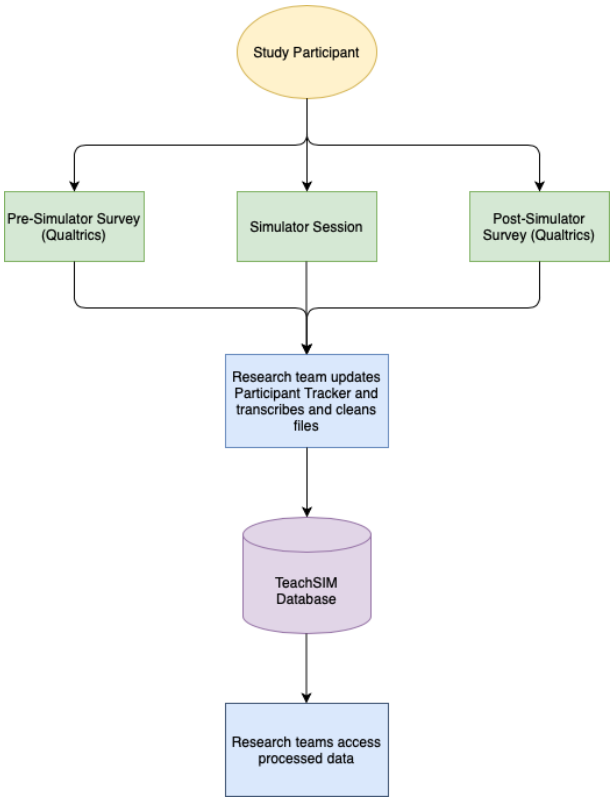

Fig. 1. TeachSIM study workflow.

B. Hosting Alternatives

Our project is centered around the design and implementation of a data pipeline that is able to support the full life cycle of an education research project. There are numerous factors that influence the decision of what platform to use to host such an infrastructure. These considerations include: financial cost, technical feasibility, security concerns, automation capabilities, reproducibility, and scalability [1] [17]. We also consider the ease of use by education researchers who may not be familiar with advanced data storage or processing platforms. The financial cost includes any upfront expenditures, the estimated monthly operating costs, and any anticipated allowances for required maintenance. Technical feasibility relates to the solution's ability to host each piece of the pipeline and whether it must be complemented with other tools to be effective [16]. Security concerns include how personally identifiable information is handled and how the solution adheres to university compliance policies [17]. The automation capabilities incorporate to what degree the pipeline will be self-sufficient, any limitations that require manual input or opportunities for pipeline connectedness, and the ability to automate processes that are otherwise performed or run by hand [17]. Reproducibility refers to the level of ease in altering this pilot pipeline or replicating it for other studies and projects [15]. Finally, scalability pertains to how the platform is able to handle increased quantities of data and increased demand on access to the database [17]. We will consider each of these factors when evaluating the potential platforms.

1) Rivanna: The University of Virginia has its own High-Performance Computing system called Rivanna that is a centralized resource for the university and is available for computational research [14]. Rivanna initially seems to be an attractive in-house option as the project could receive a resource allocation at little or no cost. Moreover, compliance to university data security policies is achievable due to the platform being entirely owned and operated by UVA. However, Rivanna is only a computational space and would be unable to host and sustain a database [14]. Therefore, the only component of the project that would be feasible with Rivanna is running any cleaning and analytics scripts. Rivanna has no automation capabilities as it is not intended to host an entire pipeline. Additionally, Rivanna is a UVA specific resource and thus would result in a project that is only reproducible within the university community.

2) Purchasing and Operating a Server: Operating a server poses several advantages including physical control of the data and keeping the data in-house. However, the upfront costs of purchasing a server can be prohibitive, especially if one does not know what their computational needs will be. For example, if the project pipeline is being underutilized by the server, we would have overspent on the server. Likewise, if the project scales with hundreds of researchers utilizing it, the purchased server may not be enough and another one would have to be added, costing both money and time as additional server space is brought up and utilized. Additionally, physical servers require maintenance and patching by IT personnel that add additional costs – when utilizing a cloud provider the cost and act of upkeep is abstracted away. Overall, this could require an upfront investment of $3,000-$5,000 with possible maintenance costs of an additional $100 per month [7]. The majority of the cost being upfront reduces the flexibility of this solution. Practical security concerns do not differ between the cloud and an in-house server; however, in terms of university compliance, owning a server may be more advantageous. As previously mentioned scalability is also a concern for purchasing and operating a server. The technical feasibility and automation capabilities remain with owning a server, which was not a feature of Rivanna.

3) Amazon Web Services (AWS): AWS is one of the potential cloud-based solutions that could host the entirety of the data pipeline. Based on our team's infrastructure design, AWS would cost approximately $266.26 per month, which amounts to $7,987.80 over the 30-month project for which our partners at the School of Education would be implementing this pipeline [4]. AWS offers extensive technical feasibility and would be able to host an end-to-end solution. Moreover, this is a highly utilized platform, which benefits from substantial documentation, example use cases, and available support. While technical feasibility is of no question, there are security compliance concerns related to AWS. Currently, the University of Virginia compliance protocol requires that no personally identifiable information is hosted in a cloud-based environment such as AWS. This requires that all data uploaded to, processed on, stored on, or accessible from AWS must be de-identified. However, AWS offers significant automation capabilities. Once raw data is uploaded to the cloud, processing and cleaning the data, writing it and storing it in the database, providing analytics and metrics, and creating an API to query the database can all be done in the cloud with no necessary manual intervention. Furthermore, connection with Qualtrics could further reduce the need to manually upload raw data as de-identification may be done in Qualtrics prior to the data reaching the cloud. AWS also provides automatic scalability to resource needs and uses a framework that would allow the pipeline to be reproducible with minimal adjustments.

4) Microsoft Azure and Google Cloud: The three dominant cloud-based platforms: AWS, Microsoft Azure, and Google Cloud are quite similar in their capabilities and cost. For the focus of this project, each would be able to accomplish the needed technical capabilities and automation. For example, AWS Lambda is what allows for events such as data upload to trigger action in the pipeline, and Microsoft offers Azure Automation and Google has Cloud Functions, each of which performs similar functionality. Moreover, security concerns over the cloud are not affected by vendor as each allows for significant security protocols, but the issue lies with university compliance rather than the actual security risk of cloud platforms. Despite the similarities, AWS standouts from its competitors mostly in its domination of market share that has led to extensive documentation, tutorials, and other widely available assistance that makes it more attractive to those who

are less familiar with cloud-based architecture [12]. For these reasons, we will use AWS as our cloud-based platform when comparing with the other hosting alternatives.

IV. Results

We have identified and evaluated the three main alternatives for the implementation of the data pipeline as part of this project: Rivanna, owning and operating a server, and a cloud-based solution. Specifically, we evaluated them according to financial cost, technical feasibility, security concerns, automation capabilities, reproducibility, and scalability. Given the technical limitations of Rivanna as a strictly computational space, it would be unable to host a substantial portion of the data pipeline and thus would not be an effective platform despite its low financial cost and high security. This leaves the alternatives of a local server or using a cloud-based platform, specifically AWS. There are significant benefits to hosting the pipeline in the cloud that make AWS more attractive than a local server. Among these are financial costs, automation capabilities, reproducibility, and scalability. Cloud-based solutions do not require the upfront expenditure of purchasing a server and have more flexible pricing options. AWS makes automation easier by providing services that are easily integrated across different steps of a pipeline. Having an architecture designed according to AWS specifications makes it more easily reproducible in AWS for other research projects. Lastly, AWS enables automatic scalability to meet the demands on the pipeline while owning and operating a server in-house would require additional expenditure to scale to higher demands. However, the main drawback stems from security and compliance concerns due to the university's current policies related to the requirements of hosting data in the cloud. For these reasons, we use a cloud-based platform, namely AWS, to design and implement our architecture.

V. Discussion

A. Designing in AWS

There are some key terms specific to AWS that needs to be addressed before being able to fully digest our architecture design. Amazon S3 (Simple Storage Service) is the storage infrastructure service provided through AWS. Essentially, S3 buckets are where users can upload and store data, such as csv files. AWS Lambda functions are essentially serverless snippets of Python code that run in the Cloud in response to events, such as data being uploaded to an S3 bucket. Lambda functions, and serverless computing in general, have the added benefit of abstracting away the management of servers and simply focusing on writing code. The pricing model for serverless computing is also beneficial and scalable, only paying for the time that the code is running. Amazon RDS is a distributed relational database service offered by AWS to host a database in the cloud. Amazon API Gateway is a service that enables the development and management of APIs through AWS. An API is an application programming interface that allows a user or program to communicate with another program or database.

In order to design the data architecture for AWS, we started with the workflow for the study that was presented in Figure 1 and created the pipeline seen in Figure 2. The 'Data Input User' represents Qualtrics or an actual researcher uploading data to an S3 bucket. Obtaining the raw data is thus the first step in the data pipeline, which is followed by data cleaning. Translating this to AWS, a member of the research team uploads the de-identified raw data to an S3 bucket, which triggers a Lambda function to clean the data and then store the output in a second S3 bucket. The next step in the workflow is to add the data to the database. Traditionally, this is simply a storage system such as SharePoint or Box. However, in the pipeline, the cleaned data arriving in the second S3 bucket will trigger another Lambda function to write this data to the database hosted on RDS. Lastly in the workflow, research teams have access to the data. Since the data is in a relational database rather than on SharePoint, we use an API gateway that allows the research team to communicate with and query the database.

The main components of implementing the data pipeline described previously and illustrated in Figure 2 were designing the database, writing the Lambda functions, and building the API.

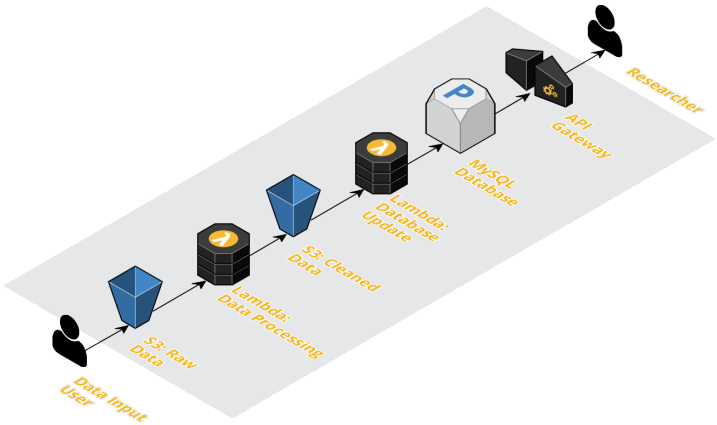

Fig. 2. Data Pipeline Design.

B. Relational Database

One of the major decisions in implementing the data pipeline was the choice of the database. A relational database, MySQL, was chosen as opposed to NoSQL solutions for several reasons. From a reproducibility perspective, this pipeline will be utilized by researchers – SQL querying knowledge is both more common and standard than that of NoSQL querying languages, which often vary depending on which database engine is utilized. Second, the nature of survey data naturally represents itself in a relational format – the data

consists of rows of de-identified participant ids with columns that represent measures or aggregated metrics for that specific participant. Additionally, the use of tables presents an opportunity for organization. For example, in the TeachSIM database, we have five tables. 'Identifiers' is used to store information about the participants and study, such as site location, study year, control groups, etc. Three of the tables store all of the participant responses on the surveys because each of the 10+ surveys falls into one of the three categories, which enables a clear opportunity for aggregation and organization into only three tables. The last table is 'Participant_Tracker', which consists of the names of the surveys as columns, the participants as rows, and the values as indicating whether or not the participant has completed that particular survey.

C. Lambda Functions

Lambda functions are a useful tool for automating otherwise manual work at low cost when the work to be done is triggered by a specific event, such as a participant in a study filling out a survey and thus generating new raw data. While the generation of data in a manual pipeline may require a researcher to clean the file locally and upload it to the data store, this can all be done automatically with prebuilt Lambda functions as a method of increasing efficiency and reducing the redundancy of manual processes.

We utilized AWS Chalice to write and deploy our Lambda functions; however, we could have also built them solely using the AWS Management Console. The benefit of Chalice is that the back-end integration with S3 and API Gateway is wrapped in a decorator that simplifies the construction of Lambda functions. It can be thought of as Infrastructure as Code (IaC), where a developer specifies the resources that need to be deployed in a programmatic fashion. Chalice also allows the python code to be written locally and deployed from the command line. Whether the Lambda functions are built through Chalice or the Management Console, they function the same.

The first Lambda function in our pipeline is triggered by raw data being uploaded to an S3 bucket. The Lambda function fetches the csv data, performs data cleaning, and stores the result in an intermediary S3 bucket. The main component of this Lambda function is the code to clean and preprocess the data, which is largely specific to the study. For the purposes of the pilot study, TeachSIM, the preprocessing code was translated to Python from Stata, where researchers were previously cleaning the data. An additional component of this Lambda function is the code used to generate the data for the Participant Tracker that identifies which participants were included in each of the raw survey csv files.

The second Lambda function exists to fetch the cleaned csv that was output by the previous Lambda function and insert the data into the database. The function establishes a connection with the database and matches the columns and participants to update the required tables and fields in the database. An important design decision here is how to handle participants that have duplicate entries. Essentially, the Lambda takes each individual entry and updates only the non-null fields from the data, which means that existing data will not be overwritten with null data to prevent information loss from duplicated participant entries that may have uneven fields. Unlike the first Lambda function, this code was not translated from the research team but is instead largely independent of the actual data being processed.

D. API

The API in our design serves as the secure gateway between the database and authorized personnel. Utilizing the industry-standard REST architecture, the API hosted on Amazon API Gateway consists of GET endpoints to query de-identified participant data from the database. There are five levels of use cases for querying the participant data in the database via the API : Time, Groups, Measure Categories, Specific Measures, and Field types. The function will first connect to the relational database and then take five parameters each representing the requirement for each level of use cases and automatically generate the SQL queries according to the parameters. The function will return the result of SQL queries in a tabular format (either JSON or csv).

E. Security

Because our pipeline consists of de-identified information about people, security is of the utmost importance. We employ a 'defense-in-depth' approach to security with several layers to ensure secure access to data. First, all of our cloud resources live within a VPC: a Virtual Private Cloud. This ensures that our resources are not even externally routable on the public internet. Additionally each resource belongs to a Security Group: essentially a set of configurations for a firewall to ensure that traffic can enter and leave to specified IP address ranges. The final step in the pipeline is to allow access to the data to authorized users, this is done via an API Gateway. This API serves as the 'middle-man' between our RDS database and researcher users. To ensure security at the API level, API keys are utilized to ensure that only authorized individuals can access the API endpoints.

F. Next Steps & Project Extensions

1) Transcription Data: Throughout this paper, we have limited the discussion of our data pipeline to the csv files produced by the survey data; however, another important component is the transcriptions of the simulation sessions. These transcriptions are in the form of text files and are thus handled much differently than the survey data. Incorporating this text data into the pipeline is still in infancy due to the added complexity. However, we have proposed a design for such an all-inclusive architecture that can be seen in Figure 3. Essentially, the process for csv files will be mimicked in the overall design but vastly different in specifications and it will be executed in parallel. The process starts the same with a member of the research team uploading the raw data to an S3 bucket, which will be specifically designated for transcript files. Instead of the data simply being cleaned, the upload

will trigger a Lambda function that executes natural language processing (NLP) analytics such as cosine similarity scores for documents. The results of which will be stored in a csv file that ties specific fields to the transcripts produced from a participant's simulator sessions. This output will then trigger a Lambda function that updates a table in the database specifically designed to store the results of such analysis. While the section of the pipeline has not yet been implemented, there is a team working on the NLP analysis for the TeachSIM study and we are working with them to map out the implementation of this design.

2) Qualtrics Automation: The pipeline as we have described it relies on a member of the research team uploading de-identified raw data to the first S3 bucket. However, in our original design, this data was pulled directly from Qualtrics, the software through which the participants fill out the surveys. Qualtrics is a widely used survey platform and plays a major role in research projects across education and the social sciences. Thus, designing a pipeline that integrates with Qualtrics provides extensive benefit to the field and makes the pipeline all the more attractive for adaptations to other studies and research projects. In addition to an API that enables automated data fetching, Qualtrics also has the capability to host an identification cross-walk whereby the research team could automate de-identification through Qualtrics so that all data being pulled into the pipeline is de-identified. In this case, a Lambda function could then be used to pull the de-identified survey data directly from Qualtrics and deposit it in the S3 bucket, thus triggering the pipeline as is currently implemented. The main roadblock to implementing this aspect of the pipeline was receiving access from the university Qualtrics team to the Qualtrics API that is needed to fetch the data. However, this obstacle is dependent on the institution that controls the Qualtrics accounts and therefore the API access. There is substantial potential for the development of this component of the pipeline to further increase efficiencies and reduce manual work, but it was not able to be achieved within the scope and time frame of this project. Thus, this remains a manual process in our current data pipeline but presents an opportunity to extend the reach of this project.

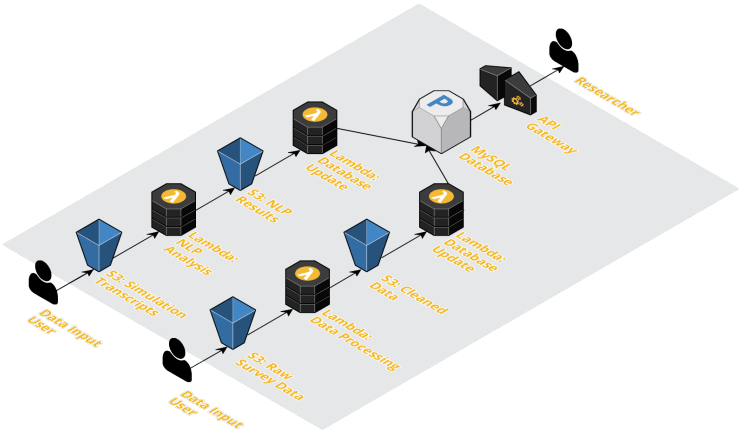

Fig. 3. Extended Data Pipeline Design.

REFERENCES

[1] "A cloud services comparison of the top three iaas providers," 22-Dec-2020. [Online]. Available: https://www.cloudhealthtech.com/blog/cloud-services-comparison. [Accessed: 31-Mar-2021].

[2] Chartier, C. (2019, March). News from the Accelerator – March 2019. Retrieved from https://psysciacc.org/2019/03/29/news-from-the-accelerator-march-2019/

[3] Coyne, M. D., Cook, B. G., & Therrien, W. J. (2016). Recommendations for replication research in special education: A framework of systematic, conceptual replications. Remedial and Special Education, 37, 244-253. doi: 10.1177/0741932516648463

[4] "Draw AWS diagrams," Cloudcraft. [Online]. Available: https://www.cloudcraft.co/. [Accessed: 31-Mar-2021].

[5] Gomez, O., Juristo, N., & Vegas, S. (2010). Replication, Reproduction, and Re-analysis: Three ways for verifying experimental findings. 1st International Workshop on Replication in Empirical Software Engineering Research.

[6] Hüffmeier, J., Mazei, J., & Schultze, T. (2016). Reconceptualizing replication as a sequence of different studies: A replication typology. Journal of Experimental Social Psychology, 66, 81–92. https://doi.org/10.1016/j.jesp.2015.09.009

[7] Lahn, M. (2021). How much does a server cost for a small business in 2021? [Online] Available: https://www.servermania.com/kb/articles/how-much-does-a-server-cost-for-a-small-business/. [Accessed: 24-Mar-2021].

[8] Lemons, C. J., King, S. A., Davidson, K. A., Berryessa, T. L., Gajjar, S. A., & Sacks, L. H. (2016). An inadvertent concurrent replication: Same roadmap, different journey. Remedial and Special Education, 37, 213-222. doi: 10.1177/0741932516631116

[9] Makel, M. C., & Plucker, J. A. (2014). Facts are more important than novelty: Replication in the education sciences. Educational Researcher, 43, 304–316. doi:10.3102/0013189X14545513

[10] Makel, M. C., Plucker, J. A., Freeman, J., Lombardi, A., Simonsen, B., & Coyne, M. (2016). Replication of special education research: Necessary but far too rare. Remedial and Special Education, 37, 205–212. doi: 10.1177/0741932516646083

[11] Makel, M. C., Plucker, J. A., & Hegarty, B. (2012). Replications in Psychology Research: How Often Do They Really Occur? Perspectives on Psychological Science, 7(6), 537–542. https://doi.org/10.1177/1745691612460688

[12] Miller, R. "The cloud infrastructure market hit 129B in 2020," TechCrunch, 04-Feb-2021. [Online]. Available: https://techcrunch.com/2021/02/04/the-cloud-infrastructure-market-hit-129b-in-2020/. [Accessed: 31-Mar-2021].

[13] Moshontz, H., Campbell, L., Ebersole, C. R., IJzerman, H., Urry, H. L., Forscher, P. S., ... Chartier, C. R. (2018, April 3). The Psychological Science Accelerator: Advancing psychology through a distributed collaborative network. https://doi.org/10.1177/2515245918797607

[14] "Research Computing," UVA Research Computing. [Online]. Available: https://www.rc.virginia.edu/service/high-performance-computing/. [Accessed: 31-Mar-2021].

[15] Rupprecht, L., Davis, J., Arnold, C., Gur, Y., & Bhagwat, D. (2020). Improving Reproducibility of Data Science Pipelines through Transparent Provenance Capture. PVLDB, 13(12): 3354-3368. https://doi.org/10.14778/3415478.3415556

[16] Simplilearn. (2021). Understanding Types of Feasibility Study, and Its Importance. [Online] Available: https://www.simplilearn.com/feasibility-study-article. [Accessed: 31-Mar-2021].

[17] Sodt, R. & Maravic, I. (2018). Data Processing Pipelines. O'Reilly Media, Inc.

[18] Wong, V. C. (2020). Developing Infrastructure and Procedures for the Special Education Research Accelerator.

[19] Wong, V. C., Steiner, P. M. (2018). Replication designs for causal inference. EdPolicyWorks Working Paper Series. Retrieved from http://curry.virginia.edu/uploads/epw/62_Replication_Designs.pdf