© 2020 Aaron Green

LIVEDATALAB: A CLOUD-BASED PLATFORM FOR DATA SCIENCE EDUCATION

BY

AARON GREEN

THESIS

Submitted in partial fulfillment of the requirements
for the degree of Master of Science in Computer Science
in the Graduate College of the
University of Illinois at Urbana-Champaign, 2020

Urbana, Illinois

Adviser:

Professor ChengXiang Zhai

ABSTRACT

The growth of the "big data" industry has created an urgent need for educating a large number of data scientists and engineers. However, learning practical data science skills requires hands-on experience with large, real-world datasets, which is difficult to offer at scale due to hardware and cost limitations. In this thesis, we present LiveDataLab, a novel cloud-based solution enabling deployment of hands-on assignments with large, real-world datasets and integration of data science education, research, and applications together in one ecosystem. LiveDataLab provides a novel project-based learning platform, alongside open leaderboard competitions, course assignment hosting, and auto-grading abilities. All of these applications are powered by a novel auto-scaling cloud backbone enabling these capabilities at relatively low cost. Additionally, LiveDataLab simultaneously serves as a platform supporting data science research via its integration with large, real-world datasets. LiveDataLab provides the ability to handle the growing demand and necessity for data science education, support data science research, and enable data science applications all in one singular platform. Ultimately, LiveDataLab brings learners, educators, researchers, and application developers together on a single unified platform to collaborate in a highly efficient big data ecosystem.

To my parents, for their love and support.

ACKNOWLEDGMENTS

I want to thank Professor ChengXiang Zhai for his support and guidance throughout the time working on this project. Additionally, I would like to thank Chase Geigle for his efforts developing the original CLaDS system, which served as the original motivation for this project. I would also like to thank Assma Boughoula for her collaboration in helping apply LiveDataLab to support CS 410: Text Information Systems. Finally, I would like to thank the several others who contributed in some way over the past couple of years to this project, as well as the University of Illinois for the resources and peers that made all of this possible.

TABLE OF CONTENTS

CHAPTER 1 INTRODUCTION 1
1.1 Motivation 1
1.2 Solution 2

CHAPTER 2 RELATED WORK 4

CHAPTER 3 SYSTEM OVERVIEW 6
3.1 Goal of LiveDataLab 6
3.2 System Design 6
3.3 Summary 13

CHAPTER 4 PROJECT-BASED LEARNING PLATFORM 14
4.1 Objective 14
4.2 Project Hosting 14
4.3 Project Submission Environment 18
4.4 Project Leaderboards 20
4.5 Summary 22

CHAPTER 5 HOSTING DATA SCIENCE COURSES 23
5.1 Objective 23
5.2 Preliminary Gitlab Hosting System 23
5.3 Project-Based Learning Platform for Course Hosting 25
5.4 Dataset Annotation System 27
5.5 Summary 28

CHAPTER 6 LIVEDATALAB AS A RESEARCH PLATFORM 29
6.1 Objective 29
6.2 Research Experiments 29
6.3 Summary 30

CHAPTER 7 DEPLOYMENT OF LIVEDATALAB 31
7.1 CS 410: Text Information Systems 31
7.2 ECE 418: Image and Video Processing 33
7.3 CS 510: Advanced Information Retrieval 33
7.4 CS 410: Online MCS Version 34
7.5 Summary 34

CHAPTER 8 CONCLUSION AND FUTURE WORK 35
8.1 Conclusion 35
8.2 Future Work 35

REFERENCES 37

CHAPTER 1: INTRODUCTION

1.1 MOTIVATION

As we progress through the information age, there is a growing need for skilled data scientists across many industries. Breakthroughs in medical diagnoses, information synthesis and recommendation to medical staff, financial modeling, business analytics, and many other sectors all demonstrate the growing value that data science provides. This is, in no small part, due to the vast quantities of data produced and accessible across almost every industry, sector, field, and business. With the advent of cloud computing, storing this immense amount of information is cheap, and virtual machines make it so anyone with sufficient funding and education can perform extensive analysis. However, the bottleneck many companies face isn't access to data, but rather skilled individuals to extract insights from it. Data scientist is one of the jobs with the most rapidly growing demand[1]. Finding educated and skilled individuals to fill the roles is the main challenge.

Alongside this need, demand for data science education is growing rapidly, as people clamor for these new, coveted positions. Learners from a variety of socioeconomic, geographic, and technical backgrounds are all looking for ways to gain these desired skills. As a result, online learning services such as Coursera[2] are growing rapidly to meet this demand, enabling users to access lecture videos from esteemed institutions and field experts. Traditional universities are also beginning to offer online degree programs to meet this demand. The University of Illinois now offers an entirely online Masters of Computer Science[3], and the U of I is far from the only university to explore these novel avenues to expand its reach.

Industry and learner demand are both growing rapidly; however, the main challenge lies within actually providing learners a practical, useful data science education that will allow them to succeed and create value once they leave the program. Hands-on experience with large, real-world datasets is essential for gaining practical data science skills. However, with learners coming from widely varying socioeconomic and technical backgrounds, many of them do not have the computational resources to set up an environment sufficient for working with real-world scale datasets. In addition to needing to download a massive dataset, for which their computer may not have enough storage, and set up the proper tooling, for which they may not have the technical background, their computer also likely isn't powerful enough to

[1] https://business.linkedin.com/content/dam/me/business/en-us/talent-solutions/emerging-jobs-report/Emerging_Jobs_Report_U.S._FINAL.pdf

[2] https://www.coursera.org/

[3] https://cs.illinois.edu/academics/graduate/professional-mcs-program/online-master-computer-science

run experiments in a timely manner. This is especially true in the machine learning and deep learning fields which require powerful and expensive graphics processing units (GPUs). Cloud computing could solve the problem of providing powerful hardware; however, it does so at a mighty cost while also requiring a fairly complicated tooling installation and setup process. GPUs on most cloud services can cost near $500 per month. This is far too expensive for a vast majority of learners; therefore, assignments are restricted to using small, downloadable datasets. However, algorithms often perform differently on small vs. large datasets. Thus, an environment where learners can gain a practical, hands-on data science education at an acceptable cost remains an open challenge.

1.2 SOLUTION

To address the challenge of providing a practical, hands-on data science education, we propose LiveDataLab, a novel cloud-based platform that provides several key pieces of functionality for enabling data science education at scale. LiveDataLab, based on an auto-scaling cloud backbone, provides a project-based learning platform with open leaderboard competitions, leveraging an Azure cloud file system for supporting large datasets. This provides learners:

1. A central place to find data science assignments and automatically tagged online learning materials relevant to the assignment
2. Robust cloud infrastructure with real world datasets where students can run their code, free of worry about complex and expensive computational requirements
3. Open leaderboard competitions where students can see how their solutions compare to many others

The generalized auto-scaling nature of LiveDataLab allows it to be leveraged for several other use cases as well. In addition to the project-based learning platform and open leaderboard competitions, LiveDataLab enables:

1. Courses to be hosted on the platform, so that traditional university data science courses can also leverage the infrastructure and improve the quality of assignments they deliver to students
2. University research groups to create powerful machines on-demand, avoiding the need for resource sharing and waiting to run experiments, and with up-time optimized for cost effectiveness

Finally, by bringing learners, educators, researchers, and industry professionals onto a singular platform, leveraging leaderboards and projects, LiveDataLab enables the integration of education, research, and application development. Research groups and industry partners can create open data science challenges on the platform, accessible to everyone. This creates a symbiotic ecosystem where students can gain hands-on experience working on practical industry and research problems, while researchers and industry partners may also discover novel or improved solutions to problems, as well as identify promising individuals to join their teams.

As discussed later in this thesis, we have used LiveDataLab to support multiple courses at the University of Illinois successfully with great results, enabling over one thousand students to benefit from the platform in data science training, including both on-campus and online students. We have found that LiveDataLab is able to enhance the quality of assignment provided to students via using larger datasets for more practical skill development, as well as through leaderboard competitions where students can directly evaluate how their solution performs in regards to the rest of the class. Additionally, we have been able to provide all of this infrastructure at a relatively low cost[1]. As LiveDataLab is a general infrastructure, it can be easily used to support courses at other universities around the world and to enhance and accelerate data science education.

The rest of this thesis is organized as follows: In Chapter 2 we review related work in the development of virtual data science laboratories. Then, in Chapter 3, we discuss the system overview and architecture of LiveDataLab. Following, in Chapter 4, we present the project-based learning platform. In Chapter 5, we discuss the ability of LiveDataLab for hosting data science courses. Then, in Chapter 7, we share how LiveDataLab has been deployed to support courses thus far. In Chapter 8, we conclude the thesis and discuss future work.

CHAPTER 2: RELATED WORK

There have been a few other efforts to develop virtual laboratories for students to work on some subsection of data science tasks. The first we will discuss is the Cloud-based Open Lab for Data Science (COLDS)[2]. COLDS main focus is providing an online annotation system for students to enter queries and rank the returned documents. This enables researchers to crowdsource dataset annotation, while also allowing students to contribute to developing a new, novel dataset and also gaining an understanding of part of the data science process. The main limitation of COLDS, however, is that it did not provide a way for the dataset annotations to be extracted from the database in which they're collected and formatted such that the dataset can be used for information retrieval tasks. As part of our work in LiveDataLab, we updated and created an integration with the COLDS annotation system, adding export functionality and storing the exported dataset annotations in the LiveDataLab cloud infrastructure, so that the newly annotated datasets can be used for education and research tasks. COLDS is limited to dataset annotation, however, and does not provide the ability for learners to work on projects or run code alongside those datasets. LiveDataLab provides a more complete virtual laboratory, allowing learners to work on projects, run code, and compete in leaderboard competitions using the real-world datasets annotated in the original COLDS system.

The second major effort at developing a virtual lab was the CLAIRE system[3]. CLAIRE sought to provide a web-based lab for students to run repeatable information retrieval experiments and visualize the results. Within CLAIRE, students can select an existing retrieval function, such as BM25, or they can implement their own retrieval function. Then, they can evaluate the retrieval function on a handful of popular information retrieval datasets. The main limitation of CLAIRE was that all retrieval functions were executed on the same main server, which wouldn't scale past a small classroom setting. As part of our work in LiveDataLab, we migrated CLAIRE to be hosted on LiveDataLab's cloud infrastructure, so that student retrieval functions are executed in an auto-scaling environment, allowing the value of CLAIRE to be expanded and support a much wider audience. However, CLAIRE is strictly limited to running experiments using MeTA's[1] retrieval functions. LiveDataLab, on the other hand, enables any code to be run on its infrastructure.

The third major effort at developing a virtual lab was the Cloud-based Lab for Data Science (CLaDS)[1]. CLaDS provided an auto-scaling cloud infrastructure for executing student code alongside large datasets. Gitlab was used as the core technology for hosting student

[1]https://meta-toolkit.org/

assignments, which allowed CLaDS to leverage Gitlab CI as the auto-scaling management and job coordination software. CLaDS also integrated with two separate leaderboard applications for search engine and classification competitions. Students could submit their code to their Gitlab repository, which would trigger Gitlab CI and clone the student's repository onto a virtual machine, where their code would then be evaluated on a variety of datasets, and the results would be sent to the appropriate leaderboard application. The leaderboard would then display all of the results for the students in the class in a competition style. The main limitation with CLaDS is its requirement of using Gitlab. As a result, all courses that would wish to use the CLaDS infrastructure are required to be hosted on the same Gitlab instance. This is a major limitation for expansion to other courses and universities, since they all have their own methods for operating. This also prevents CLaDS from being used for powering other applications such as a project-based learning platform, since those other platforms cannot exist within Gitlab. LiveDataLab contains its own assignment-hosting platform backed by a custom integration to a Jenkins cluster, which allows for a greater variety of applications and functionality to be provided, as opposed to CLaDS and its reliance on Gitlab and Gitlab CI. Nonetheless, CLaDS served as the original inspiration and starting point for the LiveDataLab project.

Finally, Kaggle[2] and CodaLab[3] both offer excellent platforms for open data science competitions. They have many users competing in leaderboard challenges using real-world datasets. However, unlike LiveDataLab, neither Kaggle nor CodaLab support the ability to host courses or provide recommended learning materials directly alongside assignments. Thus, LiveDataLab provides a stronger educational environment than the platforms that currently exist, while simultaneously supporting open data science competitions and providing research infrastructure for running experiments.

[2]https://www.kaggle.com/
[3]https://codalab.org/

CHAPTER 3: SYSTEM OVERVIEW

In the previous chapter, we discussed related work surrounding the development of virtual data science laboratories. In this chapter, we will present the system design of LiveDataLab, the underlying technologies, the benefits we saw from our architecture decisions, and design challenges we overcame along the way.

3.1 GOAL OF LIVEDATALAB

The LiveDataLab system provides several applications within a singular platform. Below are the key applications, which we will discuss in more detail in their each respective chapter. They are listed here for convenience before discussing the underlying technology behind them.

1. A general API for executing any code alongside large, real-world datasets
2. A project-based learning platform
3. Open leaderboard-style competitions
4. Course hosting and assignment hosting

3.2 SYSTEM DESIGN

Before diving into each of the above applications, we will first discuss the system design and technologies that make them possible. Our system is comprised of a few key components: a Jenkins cluster used for managing virtual machines on demand, a Flask API for submitting and managing jobs to be executed on the Jenkins cluster, and a core web application with a React front-end and Flask API, which operates as the project-based learning platform, providing job management, leaderboards, and course and project hosting. Figure 3.1 shows the high-level design of LiveDataLab. The remainder of this chapter contains the reasoning behind our design decisions and a more detailed breakdown of each component.

3.2.1 Jenkins Cluster

Jenkins[1] is an open-source automation tool for continuous integration and continuous delivery. We host our own Jenkins cluster on Microsoft Azure to provide virtual machine

[1] https://jenkins.io/

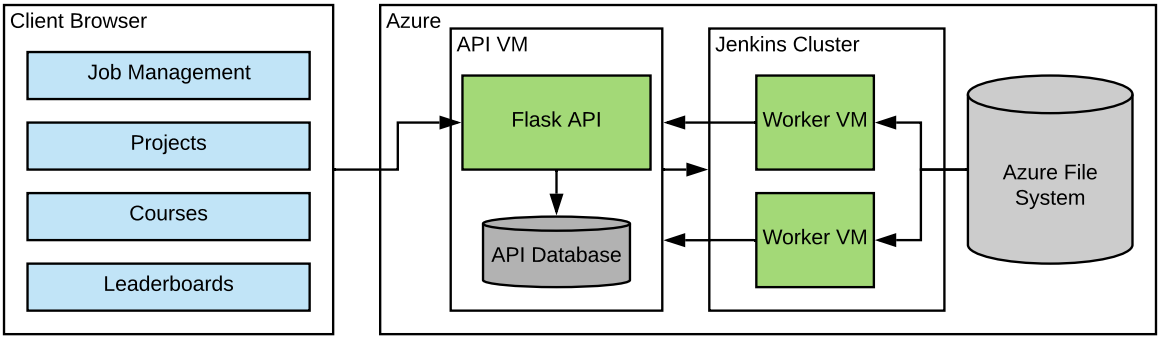

Figure 3.1: The high-level design of LiveDataLab

management for executing code alongside large datasets. Our main API interacts with the Jenkins API to submit jobs to be run on virtual machines. Our Jenkins cluster is configured such that when it creates a new virtual machine, it mounts an Azure file system onto that virtual machine which contains all of our large, real-world datasets. Since all of this is within the Microsoft Azure network, the datasets are mounted very quickly, despite being quite large. Jenkins also allows you to configure how long virtual machines remain up after their initial job completes, as well as how many virtual machines should always be running. Since creating a new virtual machine can take a couple of minutes, this allows us to configure how quickly the cluster can take on initial jobs before needing to spin up more virtual machines. When a virtual machine doesn't have a job to execute for a specified amount of time, Jenkins destroys the virtual machine. This serves as the auto-scaling portion of our infrastructure, allowing our execution environment to be extremely cost effective. This is especially valuable when using powerful hardware, for which time in the cloud is expensive.

3.2.2 Flask

Flask[2] is an open-source, lightweight, bare-bones, Python web application framework. In our system, all of our web APIs are developed in Flask and housed under one main web application in a monolith design. The other main python web frameworks are Django[3] and Pyramid[4]. We chose Python and Flask as our web back-end for the following reasons:

1. Python is a very common language in general and especially among the research community. Thus, any future developers on the project are more likely to be familiar

[2]https://flask.palletsprojects.com/en/1.1.x/
[3]https://www.djangoproject.com/
[4]https://trypyramid.com/

with the language as opposed to other popular back-end languages, such as Ruby and JavaScript (Node.js[5]).

2. Django is a large and opinionated Model-View-Controller framework that has a lot of overhead functionality and pieces that we did not require for our project. Django works well if the front-end is built using Django's templating mechanism; however, we wanted to develop our front-end is a more modern stack using React for a few reasons that will be discussed later. Thus, the lightweight and bare-bones nature of Flask allows us to mold the back-end API exactly to our requirements.

3. Pyramid, another Python web framework, is a lot less mainstream than Flask and Django, thus future developers are less likely to be familiar with Pyramid, and there's less community support on websites such as StackOverflow for support.

Our Flask back-end contains modularized functionalities supporting both the job creation and management tasks coordinated with the Jenkins cluster, as well as the APIs supporting the project-based learning platform, the details of which will be discussed later. We chose a monolith style structure, as opposed to micro-services, to simplify back-end deployments to Microsoft Azure, so that only one deployment is necessary rather than several smaller deployments. The tightly integrated nature of our back-end APIs also supports this decision.

The Flask APIs perform integrations with Github, Jenkins, and MongoDB. As a result of these several integrations, and in the interest of abstraction, each of those integrations is abstracted into a small library within the main back-end package. This allows the APIs to import their functionality and not worry about the underlying, complex integration details. This is very useful for writing business / application logic at the API and route handling level, since it enables very clean implementations as opposed to revealing irrelevant integration code to the task at hand. Figure 3.2 shows the design of the monolith Flask API.

3.2.3 MongoDB

MongoDB[6] is a very popular NoSQL document-oriented database and the main data-store in our system back-end. As a NoSQL database, MongoDB is non-relational and does not operate using the Structured Query Language (SQL); instead, MongoDB provides its own query language. The main advantage of using MongoDB as opposed to traditional rela-

[5]https://nodejs.org/en/
[6]https://www.mongodb.com/

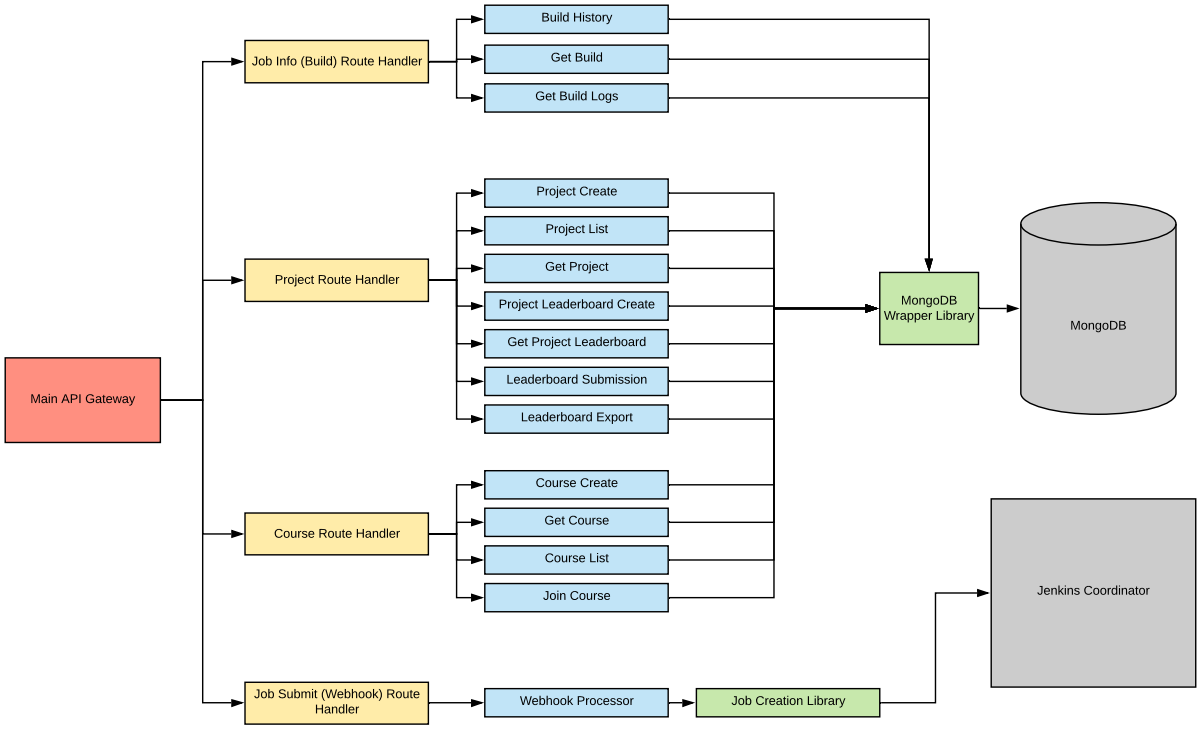

Figure 3.2: The design of the Flask API

tional databases such as MySQL[7] and PostgreSQL[8] is that MongoDB doesn't require fixed schemas for its collections (a collection is the equivalent of a table in relational databases). This flexibility is extremely advantageous for rapid development on an application where scope and features are constantly added and evolving. MongoDB is an essential and tightly integrated part of our back-end system, where we store important user-created data across several collections. Though foreign keys aren't naturally supported as in relational databases (i.e., MongoDB doesn't support JOIN operations), MongoDB does allow you to store keys from one collection in others and perform manual cross-collection lookups. Our database structure, with foreign key relationships, is shown in Figure 3.3.

3.2.4 Github

Github[9] is currently the leading platform for project version control. Github provides a powerful API, enabling users to programatically create, update, and fetch repository data.

[7]https://www.mysql.com/
[8]https://www.postgresql.org/
[9]https://github.com/

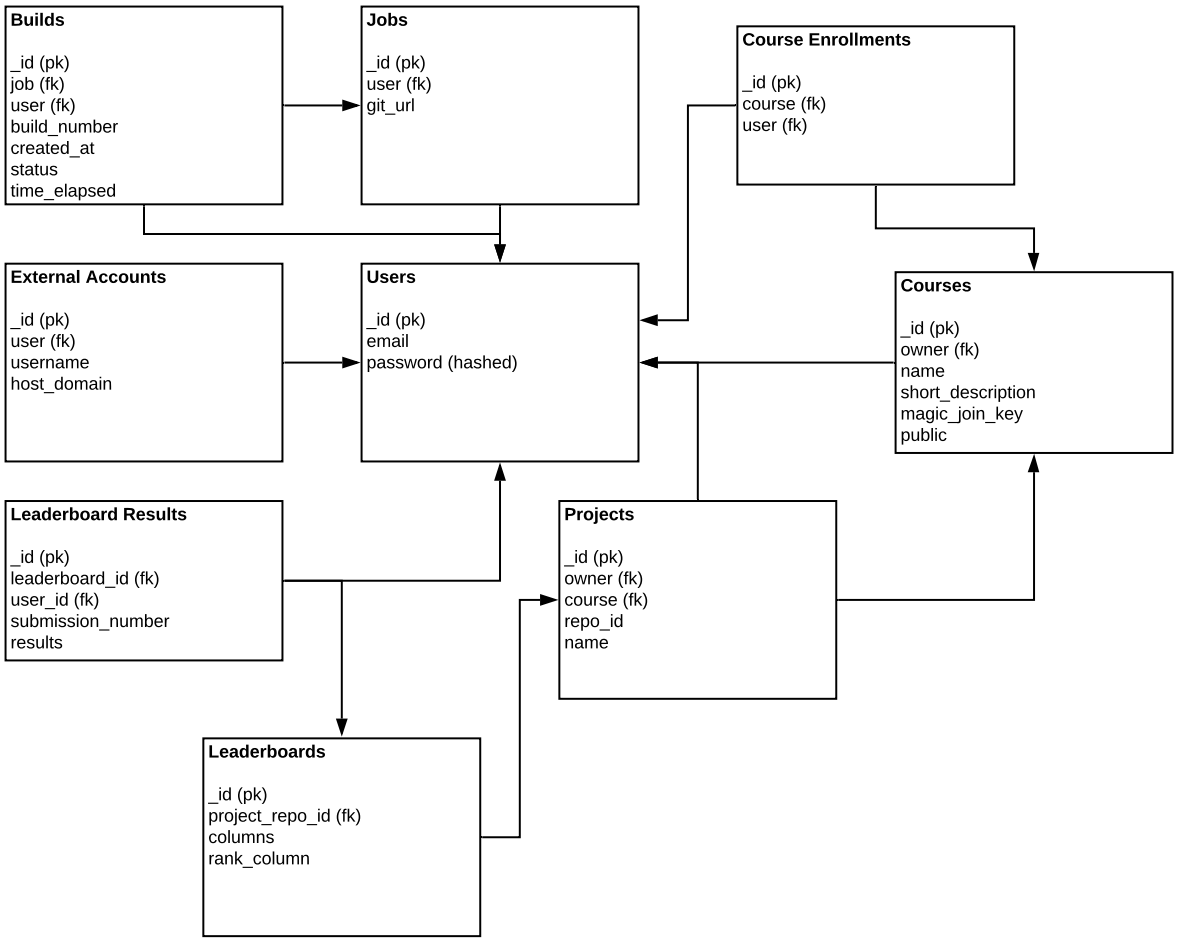

Figure 3.3: The MongoDB database schema

Additionally, Github provides an 'Organization' feature, allowing users to create an organization to house many repositories. Github is core to the project-based learning platform. When instructors create a new project on the platform, our system uses an integration with Github to create a new repository within an organization we created. This allows all of the projects to be stored in one convenient location on Github, and students can get their own version of the project to work on by simply forking the repository in the organization. Our back-end API only stores project metadata in an effort to not duplicate what is stored by Github. As a result, our API will pull project data from Github to display it to the user. However, this additional API request causes a couple-second delay for the user, which isn't acceptable on a consistent basis. Thus, we use a Redis cache to solve this problem.

3.2.5 Redis

Redis[10] is an open-source, in-memory key-value store. One of Redis' most common pieces of functionality is serving as a web cache to more quickly retrieve data for the user. In extremely large scale systems, like the ones seen at many of the major technology companies, Redis is even used as a caching layer before needing to access the main database. Since we aren't scaling to hundreds of millions of users, that isn't quite necessary. However, Redis is very useful for us to temporarily cache data retrieved from the Github API, so that every time a user is navigating around the project-based learning platform, they don't experience multi-second delays while the data is being fetched. Using Redis to cache Github data reduced page load times from a couple of seconds to be near instant. This is accomplished through our Flask API first checking the Redis instance which is in-memory on the same server, before making a cross-network API request to Github to retrieve the project information. Figure 3.4 shows the design of the caching flow.

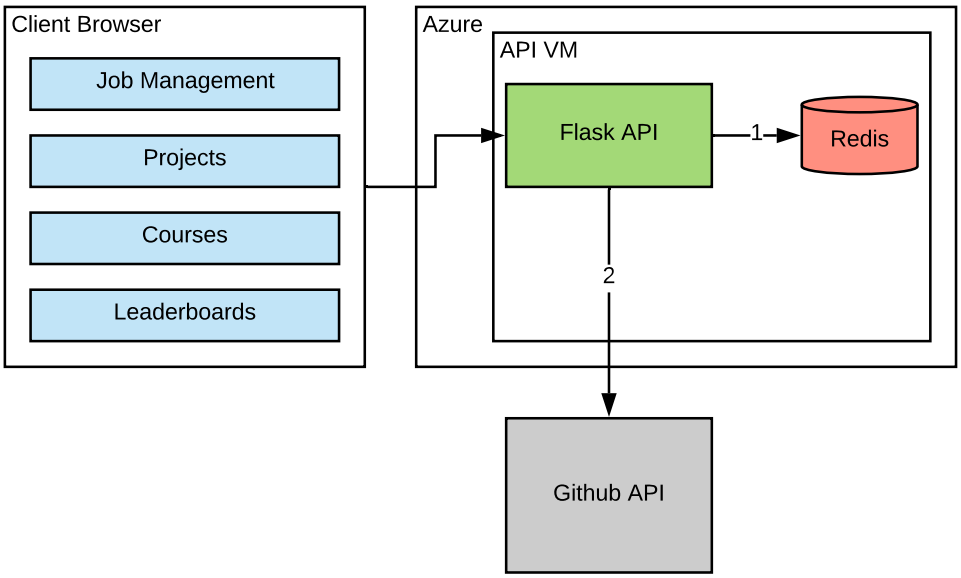

Figure 3.4: The API caching flow using Redis

3.2.6 Microsoft Azure

Microsoft Azure[11] is one of the leading cloud-computing providers and is core to our system. As part of its offerings, Azure provides virtual machines and virtual file systems.

[10] https://redis.io/
[11] https://azure.microsoft.com/en-us/

All of our back-end infrastructure is deployed on Microsoft Azure using Docker[12]. However, since auto-scaling is an essential piece of our platform, we are also able to leverage the virtual machine services that Azure provides to create new virtual machines on demand to run new jobs. This wouldn't be possible on a traditional university cluster with a fixed amount of hardware. Additionally, all of the datasets supported in the platform are stored in an Azure virtual file system. This file system is then mounted onto new virtual machines when they are created, so that jobs can access the datasets as if they were natively stored on the machine. As a result of these capabilities, Microsoft Azure is essential to the system.

3.2.7 React

React[13] is a very popular open-source Javascript library for building front-end user interfaces and applications. Other popular front-end libraries are Vue[14] and jQuery[15], although jQuery is older and not as fully-featured. Angular[16] is also a popular Javascript framework for front-end development. All of our user interfaces are built using React. We chose React for a handful of reasons:

1. React is currently the leading modern front-end development library, thus future developers are more likely to be familiar with it.

2. As opposed to jQuery, React is great for building very complex front-end interfaces while still enabling a modular, clean, and scalable design, particularly through its class and 'Component'[17] capabilities.

3. React renders views in a hierarchical tree structure, so only views that need to be updated get re-rendered. This allows React to have extremely good performance.

React's hierarchical re-rendering capabilities are very useful to the system in particular. One of the front-end functionalities is displaying job logs in real-time, and thus the page gets re-rendered when there are new logs from the server. React allows this functionality to be implemented cleanly and with good performance, not impacting the rest of the application.

[12]https://www.docker.com/
[13]https://reactjs.org/
[14]https://vuejs.org/
[15]https://jquery.com/
[16]https://jquery.com/
[17]https://reactjs.org/docs/react-component.html

3.3 SUMMARY

In this chapter, we discussed the system design and underlying technical components of LiveDataLab. We described the API design, database schema, auto-scaling mechanism, caching process, and front-end tooling. In the next chapter, we will present the project-based learning platform that is powered by this technical architecture.

CHAPTER 4: PROJECT-BASED LEARNING PLATFORM

In the previous chapter, we discussed the underlying system design of LiveDataLab. In this chapter, we will present the project-based learning platform, which helps achieve LiveDataLab's main goal of enabling hands-on data science education through supporting large-scale data science assignments and experiments.

4.1 OBJECTIVE

The main goal of the project-based learning platform is to make hands-on, practical data science education accessible to the widest audience possible. This is accomplished through openly-hosted projects, a project submission and code execution environment, as well as open project leaderboards. It should be noted that Kaggle[1] is a great platform for finding datasets and competing in data science competitions. They have many different datasets and lots of users collaborating with public notebooks to share their findings. However, Kaggle requires the user either to have a copy of the dataset on their local machine to work on the challenge, or the user has to work within Kaggle's browser-based 'Notebooks'[2] feature, whereas LiveDataLab's project-based learning platform allows users to work within their preferred environment but still run their code on LiveDataLab's cloud infrastructure. Additionally, LiveDataLab is geared more specifically for assignment and education support, where we spent special dedication towards enabling thorough customizability for projects and leaderboards to enable project hosts to deliver the most valuable experience for their learners, as well as automatically recommending related learning materials for projects.

4.2 PROJECT HOSTING

In this section, we will discuss the related functionality for projects being hosted on the project-based learning platform. Specifically, we will discuss the process for learners to work on projects, and the process for instructors to create new projects on the platform.

4.2.1 Working on Projects

The first step for any new learner to the platform, other than signing up, is linking their appropriate Github account. The user should navigate to the Github domain where they

[1] https://www.kaggle.com/
[2] https://www.kaggle.com/notebooks

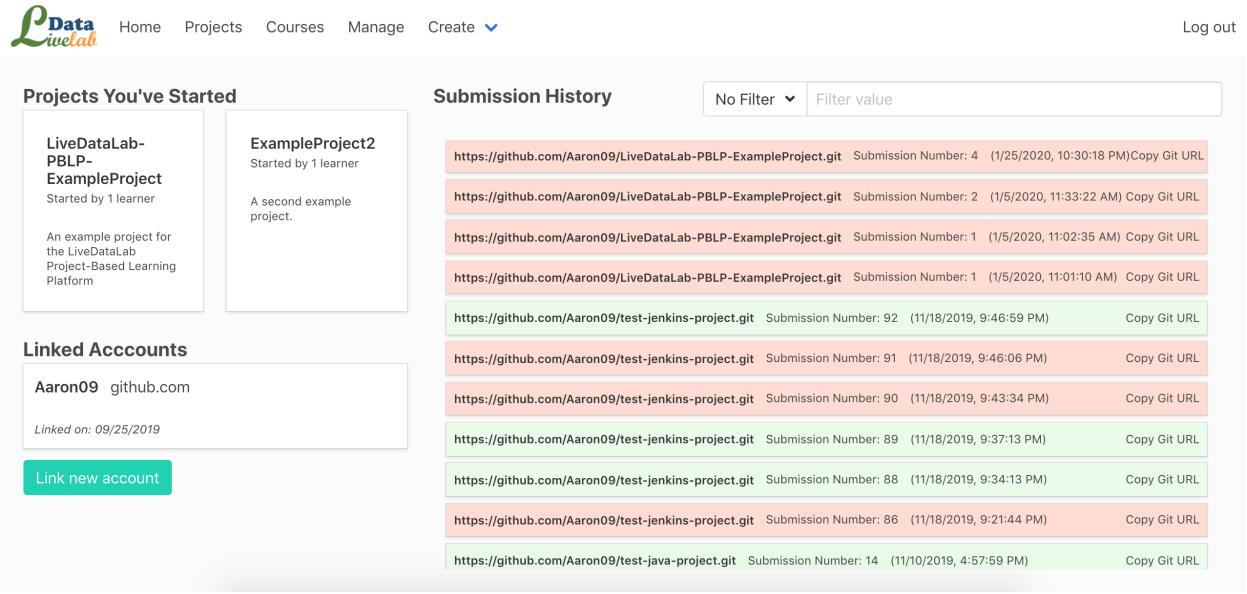

Figure 4.1: The home page of the LiveDataLab project-based learning platform

will be storing their forked projects. Unless the user is working on a privately-hosted Github service, this is almost always going to be 'github.com'. On the Github instance, the user needs to create a new API key that will be used for the project submission engine to clone their projects for execution. The user should go to 'Settings' and then click on 'Developer Settings'. They should then click on 'Personal Access Tokens' and select 'Generate new token'. They can name the token whatever will be helpful for remembering its purpose. The user should then check the checkbox for 'repo' scope. Then the user should click 'Generate Token' at the bottom of the form. The user needs to then copy the token shown and save it for entering into the LiveDataLab account linking system. Once the user has copied the token, they should navigate back to the main LiveDataLab page and click the 'Link New Account' button on the left side of the page. The user should enter the host domain (in most cases, 'github.com'), their username on this domain, and then they should paste in the value of the new personal access token they just created into 'API Key'. Then the user should click the 'Link' button. The user will now be able to integrate projects on that Github account with the LiveDataLab project submission system.

After the user has linked their account, they can begin finding their first projects to begin working on. We made this as simple as possible for users. Upon logging in, users can navigate to the projects portion of the application, where they will be presented with a list of the projects they've started, as well as all of the other projects available on the platform. Within these lists, each project displays the name of the project, how many learners have started it, and a short description of the project.

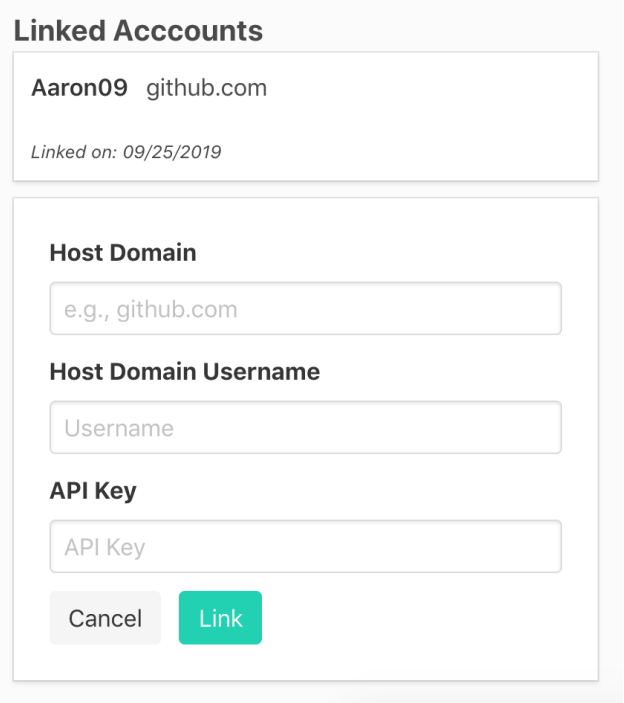

Figure 4.2: The account linking form

When the user finds a new project they want to begin working on, they can click on the project from within the list, bringing them to the projects main information page. From here, they can see their submission history for the project, a link to the leaderboard for the project (if the project host has created a leaderboard), and the link to the Github page where this project is stored.

The user can then click on that link and navigate to the Github page. From there, the user should fork the Github repository to create their own version of the project. Finally, all that remains is the user integrating the project with the submission engine of the LiveDataLab infrastructure. The submission environment integration point is through Github webhooks and will be described more later. In order to integrate, the user should click on the project settings for their fork of the project, then click on 'webhooks' and select 'Add webhook'. The user should enter 'http://livedatalab.centralus.cloudapp.azure.com/api/webhook/trigger' in the 'Payload URL' field and then select 'application/json' as the content type. The user should ensure that 'Just the push event' is selected for 'Which events would you like to trigger this webhook?' and that the 'Active' checkbox is checked. Finally, the user should click 'Add webhook'. Now the user can commit to their fork of the project, and it will trigger a new job within the LiveDataLab auto-scaling infrastructure, where the user's code will be executed.

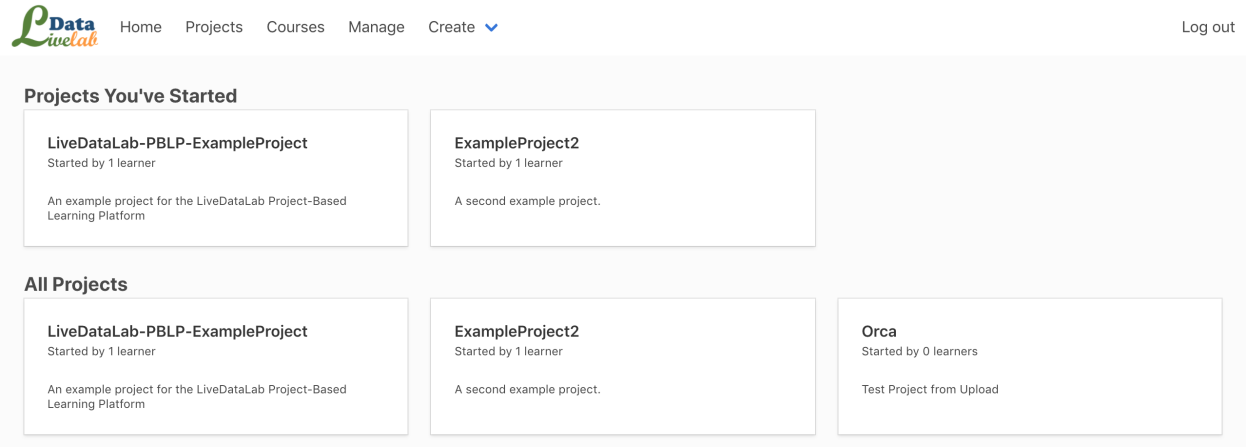

Figure 4.3: The project listing page

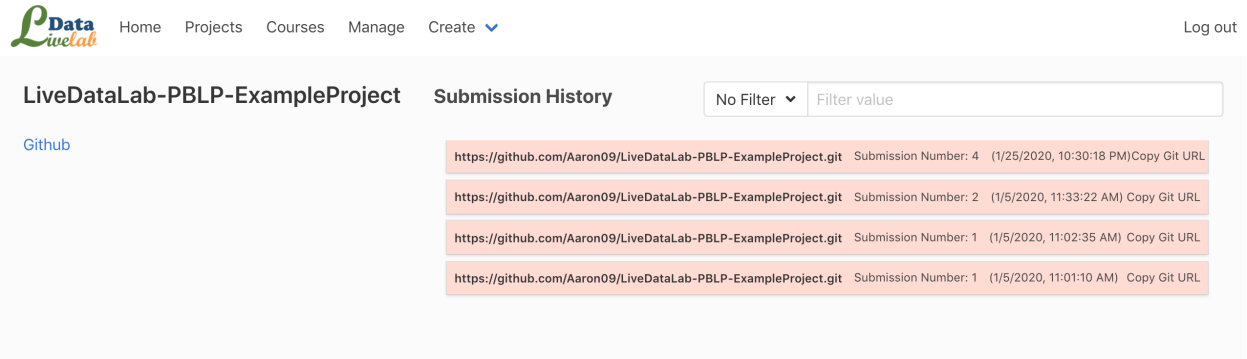

Figure 4.4: The project information page

4.2.2 Creating Projects

Equally as important as a user's ability to find projects to work on is providing an easy and intuitive process for creating new projects. To begin the project creation process, the user should go the navigation bar at the top of any page, reveal the drop-down menu under 'Create', and click 'Project'.

The user will be directed to the project creation page and prompted with a form to fill out with the project information. The first two fields are the name of the project, which will be the name of the Github repository, and the short description of the project, which is shown to learners when they're looking for new projects to work on. The third field is for the user to upload the project README.md file. Since projects are hosted on Github, this is the main documentation source for the project. Users should make sure this README.md file contains all of the instructions for the project. Next, the user should upload a .zip file which contains all of the starter files that learners will have for the project. This should include any skeleton code that learners begin with, as well as evaluation scripts and scripts

for leaderboard submission. Leaderboard submission will be described in more detail in the 'Leaderboard Creation' section. Finally, the user has the option to set the project to be a part of a course. This is not mandatory, and courses will be described more in a later chapter. Now the user can click the 'Create' button at the bottom of the form and create their new project.

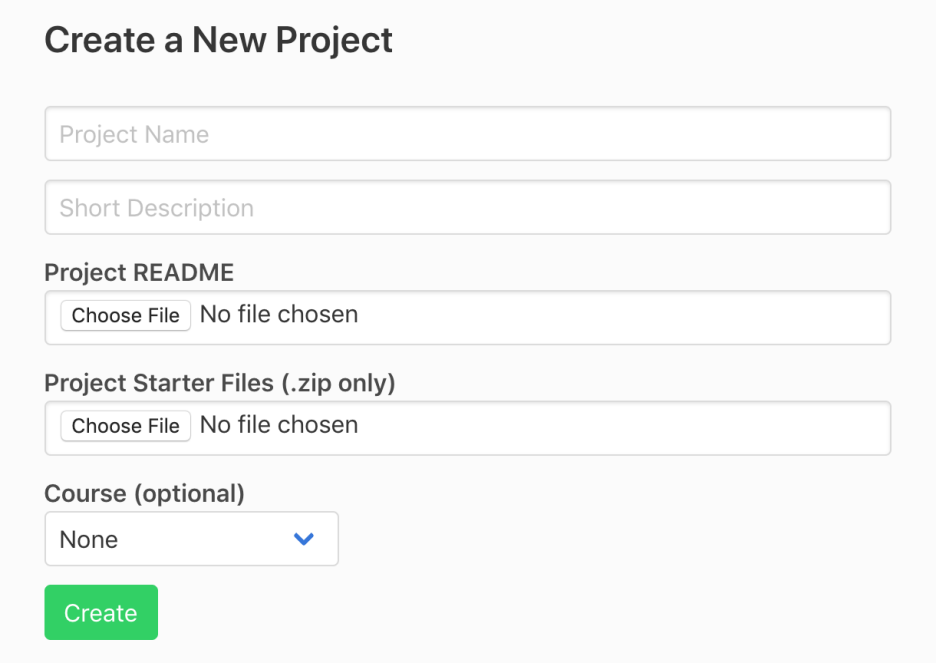

Figure 4.5: The project creation form

4.3 PROJECT SUBMISSION ENVIRONMENT

In this section, we will discuss the code submission process and execution environment for projects. This includes a technical walkthrough of the process, the submission configuration structure for projects, and how to view submission results as a user.

4.3.1 Technical Walkthrough

Before describing how project hosts can customize the submission environment and how learners interact with it, it is important to understand exactly how it works. As described in the previous section, the project submission and code execution environment is integrated with via Github webhooks. On a repository where the user has added the webhook appropriately, when they push to the repository, the webhook is triggered and sends an API request

to our main Flask API.

Our API receives this request and gets the unique ID of the Github repository that triggered the webhook. From this unique ID, the API is able to determine the user that owns this project and triggered the webhook. The API then creates an instance of a 'Job' which is our library for submitting a new job to be executed on the Jenkins cluster. When creating the job, the API specifies the Git URL that triggered the webhook.

The Jenkins cluster receives an API request from our API for creating a new job. Upon receiving this request, Jenkins will either allocate an already existing virtual machine for the task if one is open or it will begin creating a new virtual machine. When a virtual machine is ready to begin the job, the Git repository specified via the provided Git URL is cloned onto the virtual machine. The virtual machine then runs a universal execution script we have stored in Microsoft Azure that was mounted with the dataset file system during the virtual machine's creation.

This execution script will look for a file named 'livedatalab_config.json' in the cloned repository. If this configuration file is not found, the job aborts and reports as failed. Assuming the file is found, the execution script opens the file, installs the specified dependencies, and then runs the entrypoint command specified. The details of the 'livedatalab_config.json' file are described later in the 'Configuration' section. When the entrypoint command is executed, the user's code will be invoked. When the user's code finishes running, the exit status of the process will be returned to the main execution script which uses that value to determine the job as either succeeded or failed.

4.3.2 Configuration

Project execution entrypoints and dependencies are specified via the 'livedatalab_config.json' file that must be included in every project. The format of the file is as follows:

```
{
  "prescript_commands": [
    "pip3 install metapy"
  ],
  "entrypoint": "python3 -u main.py"
}
```

In the 'prescript_commands' field, users should provide a list of commands used to install any dependencies required for the project. Then they should specify the main entrypoint into the project via the 'entrypoint' field. Note that 'main.py', in this example case, is a file created by the project host. The '-u' option in the entrypoint command is required in

python scripts to return standard out logs in real-time to the parent process. This is used for rendering job logs in real-time to users in the LiveDataLab application. If the '-u' is not provided, logs from the job won't be shown in real-time and only shown in full once the entire user submission is finished executing.

4.3.3 Project Submission Logs

Viewing logs from executing projects is extremely important. Not only for providing learners feedback on why their code may have failed, but also for showing progress on long-running jobs, which is very common for data science tasks that involve training large models. As a result, it is important that users are able to access their logs as they're created in real-time, so they can see failures and track progress, making necessary changes as soon as possible. Fortunately, we provide a simple interface for users to access their logs. Within the submission histories shown throughout the application, users can click on any project submission and view its information. On the submission page, users can view the Git URL for the project, the submission number for this project, the status of the submission (e.g., QUEUED, RUNNING, SUCCESS, FAILURE), and the time duration for which the submission has been executing. Additionally, in the 'Submission Logs' section, the user will be able to see all of the outputted logs from the submission as they are created in real-time.

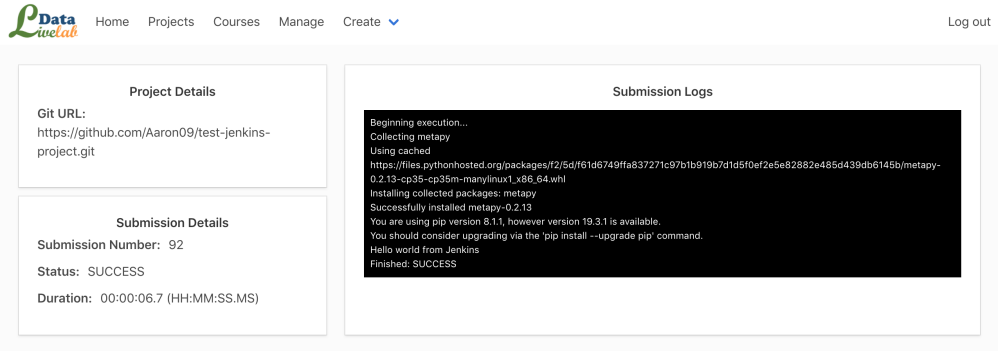

Figure 4.6: The project build page

4.4 PROJECT LEADERBOARDS

In this section, we will discuss support for project leaderboards. Specifically, we will show how to view leaderboards as a user, how to create a leaderboard for a specific project, and how to export leaderboard results to a .csv file.

4.4.1 Leaderboard Viewing

Leaderboards are an essential aspect of practical data science projects, allowing users to see how their solution compares to all other users' submissions, as well as for tracking improvement in their own submissions. Any project hosted in the project-based learning platform can have an attached leaderboard. If a project has a leaderboard, learners will see a link on the project information page that will lead them to the leaderboard. On the leaderboard page, users will be able to view the full leaderboard, including the ranks of all submissions and the corresponding columns with the submission data. One of the columns is specified as the ranking column which determines the actual rank of the submission. For example, there could be many columns for accuracy scores on individual datasets and a ranking column that is the weighted combination of these accuracy scores.

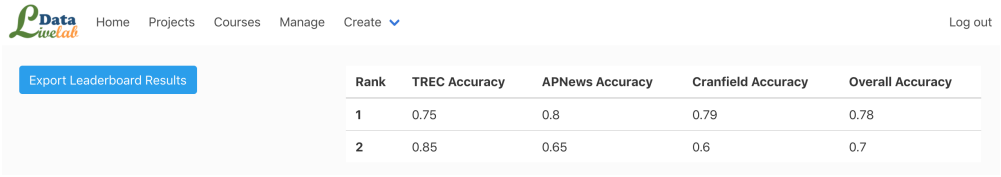

Figure 4.7: The leaderboard view as the project owner

4.4.2 Leaderboard Creation

As there are a wide variety of areas within data science, it is important that the leaderboard system is as general and customizable as possible. As mentioned before, any project is capable of having an attached leaderboard. In order to add a leaderboard to a project, the user must be the owner of the project. To add the leaderboard, the user should go to the navigation bar at the top of any page, hover over 'Create', and click on 'Leaderboard'. Here, the user selects the project the leaderboard is for from the drop-down menu. Then, the user enters the name of each of the columns they want in the leaderboard and clicks 'Add' after each one. Finally, the user must select which of the columns they created will be used for determining the final rank. Once the form is complete, the user can click 'Create' which will create the leaderboard for the project.

4.4.3 Leaderboard Exporting

When the owner of a leaderboard goes to view the leaderboard, they will see an additional button called 'Export Leaderboard Results'. When the user clicks this button, it will create

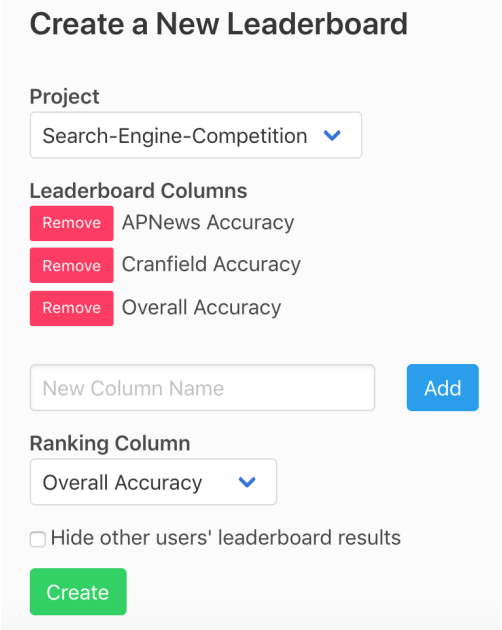

Figure 4.8: The leaderboard creation form

a .csv file containing all of the leaderboard information, which the user can download. This is useful for projects that are also assignments in courses, so the instructors can download the leaderboard results to use for grading.

4.5 SUMMARY

In this chapter, we presented the LiveDataLab project-based learning platform. We described the project hosting capabilities, project submission process, and project leaderboard functionality. In the next chapter, we will discuss how the project-based learning platform can be leveraged for hosting data science courses.

CHAPTER 5: HOSTING DATA SCIENCE COURSES

In the previous chapter, we introduced the project-based learning platform. In this chapter, we will discuss the preliminary course hosting system of LiveDataLab, and then how the project-based learning platform can be applied for improved hosting of data science courses. Additionally, we will discuss our extension of the COLDS[2] dataset annotation system and how we leverage it for hosting new kinds of assignments within a data science course.

5.1 OBJECTIVE

The goal of LiveDataLab is to enable hands-on, practical data science education for the widest audience possible. To achieve this, we also need to support formal data science courses in a university-style setting. While Kaggle[1] provides a great environment for open data science competitions, they don't provide any support for running a course on the platform. Thus, we have spent particular effort building functionality that allows the project-based learning platform to serve as a platform for hosting data science courses privately (or openly) for students. Hosting a course on the platform means that created projects are specific to the course, the course can be public or private, and students enroll in the course. This is useful for private university courses, public online learning courses, as well as for data science training programs for corporations to educate their employees. Course hosting is one of the oldest goals of LiveDataLab, and thus the mechanisms by which we support it have evolved. Course hosting was first supported using a preliminary Gitlab[2] system largely based on CLaDS[1]. This preliminary system suffered from large limitations, and as we built-out the project-based learning platform, we aimed to resolve these issues. In this chapter, we will discuss the preliminary Gitlab system, its design and limitations, then we will proceed with how the project-based learning platform supports hosting data science courses.

5.2 PRELIMINARY GITLAB HOSTING SYSTEM

In this section, we will discuss the preliminary course and assignment hosting system of LiveDataLab based on Gitlab and Gitlab CI[3]. We will provide an overview of the system and discuss its limitations.

[1]https://www.kaggle.com/
[2]https://about.gitlab.com/
[3]https://about.gitlab.com/stages-devops-lifecycle/continuous-integration/

5.2.1 System Overview

As mentioned, the preliminary Gitlab hosting system was largely based on the CLaDS[1] infrastructure. Gitlab was the platform used for hosting the courses, where all assignments would be stored and released to students, as well as where students would submit their code. Gitlab's 'Groups'[4] functionality was used for organizing students and assignments into a course. Gitlab CI was used as the auto-scaling software[5] for running student assignment submissions on virtual machines. Creating student accounts, enrolling students in a group, and exporting assignment grades was all performed manually by running various python scripts developed over time. Thus, Gitlab was the main interface for students to interact with course assignments, and instructors had to use a variety of tools to operate a course.

5.2.2 Limitations

Many limitations came from trying to use Gitlab in a manner for which it was never intended. Gitlab isn't built for running university data science courses, and many administrative tasks such as collecting grades required additional work outside of the platform. Gitlab is meant for version control, and Gitlab CI is meant for running tests and checks during the software deployment process.

As a result, many of the course management actions were accomplished through hacky workarounds with Gitlab. Students would submit their assignments and run their code in the Gitlab CI infrastructure; however, for grading, the teaching assistants would run a script to clone all of the student repositories to their local machine, run the student's assignments there, and collect grades manually. Gitlab didn't provide a way for recording these student submissions. Generating student accounts on the Gitlab system required downloading the course roster and running a script to parse it and send an API request to Gitlab to create each individual account and then add the student to the specified Gitlab Group that served as the course home for the semester. Every time a student joined the course late in the first few weeks of the semester, this script would need to be re-run to enroll the student in the Gitlab system. There was no way for the student to easily self-enroll themselves in the Gitlab ecosystem. Ultimately, the most limiting factor with Gitlab is that any course that wants to leverage the auto-scaling infrastructure for submitting student code would need to be hosted on that specific Gitlab instance. This created a serious problem for expanding the platform to be used outside the initial couple of courses at the University of Illinois.

[4] https://docs.gitlab.com/ee/user/group/

[5] https://docs.gitlab.com/runner/configuration/autoscale.html

5.3 PROJECT-BASED LEARNING PLATFORM FOR COURSE HOSTING

In the previous section, we discussed the preliminary course hosting system. In this section, we will discuss how we can leverage the project-based learning platform for improved course hosting. Specifically, we will show how to create a course, enroll students, view courses as a student and instructor, and collect assignment grades, all using the project-based learning platform.

5.3.1 Course Creation and Enrollment

The project-based learning platform was developed with course hosting in mind and aims to resolve the many limitations that existed in the preliminary Gitlab version. Improvements exist in many areas including course creation, student enrollment, grade tracking and exporting, assignment deployment, and more.

To create a new course on the platform, the user should go to the navigation bar at the top of any page, hover over 'Create' and select 'Course'. This brings them to the course creation form. Simplicity in user experience was a main focus, and so the user is only presented with the essential fields for creating a course: the name of the course, a short description for it, and whether or not the course is private or public.

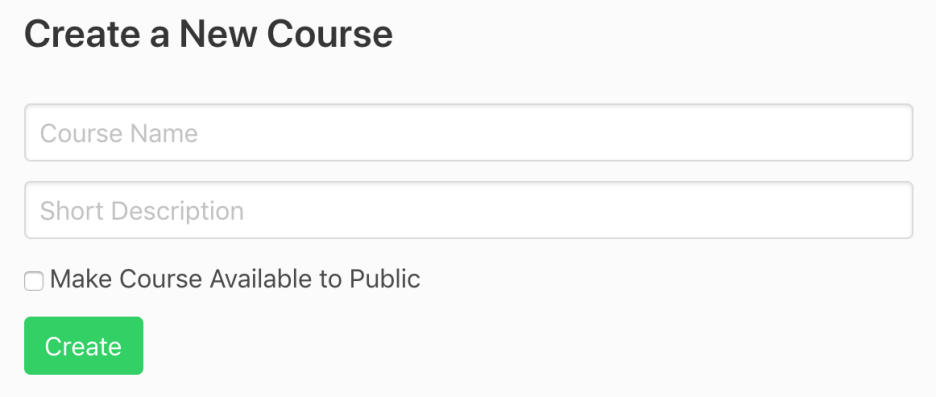

Figure 5.1: The course creation form

Upon completing the form and clicking 'Create' the user is redirected and shown a magic enrollment link. Instead of manually enrolling all of the students in the course themselves, the instructor only needs to share that link with the students. When the students go to that link, they will be prompted to log into the LiveDataLab system, and when that's complete, they will be automatically enrolled in the course. If the link is ever needed in the future, the instructor has easy access to it from the course viewing page.

5.3.2 Course Viewing

There are two main ways to view courses on the LiveDataLab system. As an instructor, the user should click on the 'Manage' field in the navigation bar at the top of any page, and this brings them to the main course and project management screen.

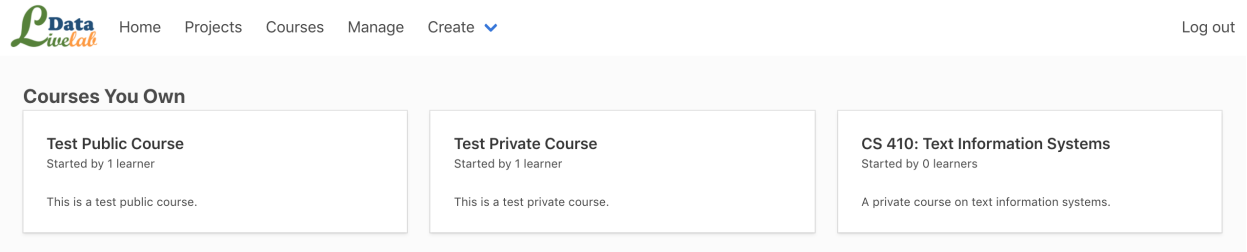

Figure 5.2: The course management viewing page

The user will be shown a list of all courses they own and all projects they own. The user can then click on a course and be taken to the main view for the course. Here, the instructor can see the name of the course, their owner status, how many students are enrolled, the magic enrollment link, and a list of the projects in the course.

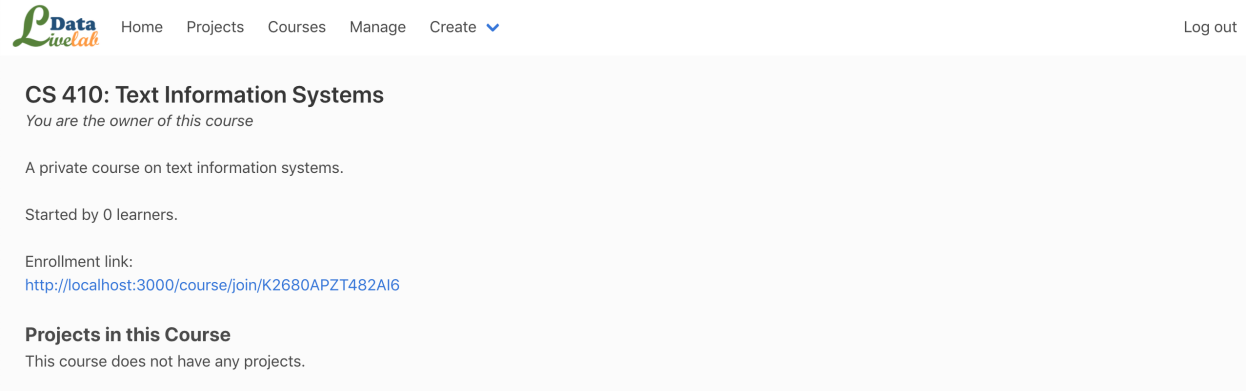

Figure 5.3: The course information page

For students trying to find a course they're enrolled in, or looking for a new public course to join, they can click on the 'Courses' field in the navigation bar at the top of any page. This shows the user a list of the courses they're enrolled in and a list of all public courses they can join. If the user clicks on a course, they are presented with the main course view from the student perspective. Very similar to the instructor's view, students can see the name of the course, the number of students enrolled, the magic enrollment link, and a list of the projects in the course.

5.3.3 Course Grading

The automation of grading is essential for running a large course, private or public, in-person or online. LiveDataLab's heavily-customizable leaderboard functionality allows it to also serve as a gradebook. For tracking grades, instructors can create a leaderboard with the fields they want for tracking grades for each assignment. Then, when instructors deploy assignments, they can have their submission script post the relevant grade information to the leaderboard to be stored. This gradebook can be easily exported for external use by the instructor using the leaderboard export functionality. If they wish, instructors can also make the leaderboard private so that students don't have access to the public gradebook. In doing so, when students navigate to the leaderboard for this assignment, they will only be presented with their grade for the assignment and not all of the other entries in the leaderboard.

5.4 DATASET ANNOTATION SYSTEM

In Chapter 2 discussing related work, we introduced the COLDS[2] system for dataset annotation. In this section, we will present our extension of the original dataset annotation system. In particular, we will discuss the system overview and the major use cases the system enables.

5.4.1 System Overview

The dataset annotation system exists as a separate application from the project-based learning platform, but still lives within the LiveDataLab cloud ecosystem. The dataset annotation system was initially implemented as part of the COLDS system[2], but we have expanded upon it and adapted it to best aid in supporting assignments in data science courses.

The dataset annotation system remains generally the same from a user interface perspective as described in the COLDS paper[2]; however, the underlying cloud infrastructure has changed. The LiveDataLab Azure file system is now mounted onto the application server. When a user uploads a dataset to the annotation system, the dataset is now stored on the cloud file system instead of the application server. Firstly, this allows more and much larger datasets to be stored since the application server won't run out of storage. Secondly, this allows the uploaded datasets to be used in data science assignments on the project-based learning platform, since the same file system is mounted on the virtual machines for execut-

ing student code. We also implemented annotation exporting functionality, which takes all of the stored annotations in the annotation system database and exports them into a usable dataset format and stores it in the cloud file system, enabling student assignments to also leverage the newly annotated dataset.

5.4.2 Major Use Cases

An important part in the end-to-end data science process is gathering data and having it labelled to be used for supervised model training tasks and evaluating new algorithms. The dataset annotation system enables students to take part in the dataset labelling portion of the data science process. This allows students to gain a bigger picture view of how all of the pieces in practical data science work fit together.

Additionally, the dataset annotation system enables the crowd-sourced generation of labels for datasets. This enables new, large datasets to be labelled by many students, especially in large courses at major universities and online courses like MOOCs. With the ability to easily create labels for new, large datasets, data science research, education, and applications alike are all able to develop, test, and evaluate novel algorithms for real data science problems.

5.5 SUMMARY

In this chapter, we shared how the LiveDataLab project-based learning platform can be leveraged, in addition to the dataset annotation system, for hosting data science courses. In the next chapter, we're going to describe how the project-based learning platform can be used to support data science research groups and research experiments.

CHAPTER 6: LIVEDATALAB AS A RESEARCH PLATFORM

In the previous chapter, we discussed how LiveDataLab's project-based learning platform can be used for hosting data science courses. In this chapter, we are going to present how the project-based learning platform can be applied as a platform for supporting data science research groups.

6.1 OBJECTIVE

LiveDataLab seeks to facilitate the intersection of data science education, industry applications, and research. Infrastructure to support research groups and experiments is essential for accomplishing this core part of LiveDataLab's long-term goals. Thus, within the same ecosystem that supports data science education, we sought to generalize the platform to aid researchers in running experiments on scalable compute and tracking results. By having researchers and students on the same platform with the same datasets, students can be exposed to open research problems if the research group desires, and research groups can view student solutions to the novel problems they're tackling, as well as how the students' solutions compare to those of the group.

6.2 RESEARCH EXPERIMENTS

In this section, we are going to discuss how research experiments operate on the LiveDataLab project-based learning platform. Specifically, we will describe the infrastructure supporting experiments, how to run experiments, and how to track experiment results.

6.2.1 Experiment Infrastructure

The auto-scaling cloud infrastructure LiveDataLab provides simultaneously works for supporting running computationally expensive research experiments, such as many machine learning tasks. The nature of the cloud infrastructure allows any virtual machine type Microsoft Azure offers to be used, ranging from single-core CPU machines to very powerful GPUs. This allows researchers to not worry about resource constraints they often run into on university-owned physical compute clusters. Additionally, by working on the project-based learning platform, researchers only need to store a small sample of the dataset they're working with on their local machine, since the full, large dataset can be stored in the cloud

file system that attaches to the virtual machines for execution. Finally, the dataset annotation system within LiveDataLab enables new datasets to be created and available for use, attracting researchers to the platform.

6.2.2 Running Experiments

Just like for creating and submitting to an assignment, researchers can create a project on the project-based learning platform, customize their dependency and execution configuration to their needs using the 'livedatalab_config.json' file, and push to their repository triggering a new experiment to run. Since each experiment is triggered with a push to the project repository, there is native support for version tracking and experiment reproducibility through the repository's commit history. On the job viewing page, researchers will be able to view the logs of their code running in real-time, as well as the amount of time it's been running for. This allows them to gain insights as they happen into their code and make adjustments and correct errors as soon as possible. This is especially useful for researchers running large machine learning training experiments that run for long periods of time.

6.2.3 Experiment Result Tracking

Researchers can also attach custom leaderboards to the research projects to track experiment results. The general nature of the leaderboards allows researchers to set any columns they want and lends itself towards supporting numerical result tracking for any kind of data science research. Additionally, the leaderboard tracks the submission number for each entry, so researchers can easily identify the version of code associated with each set of results.

6.3 SUMMARY

In this chapter, we have discussed how the LiveDataLab project-based learning platform can be leveraged to support data science research groups and run research experiments. In the next chapter, we will present how LiveDataLab has been deployed thus far to support courses at the University of Illinois.

CHAPTER 7: DEPLOYMENT OF LIVEDATALAB

In the previous chapter, we discussed how the LiveDataLab project-based learning platform can be used to support research groups. In this chapter, we're going to share how LiveDataLab has been deployed thus far to support several courses at the University of Illinois.

7.1 CS 410: TEXT INFORMATION SYSTEMS

In this section, we're going to present how LiveDataLab has been used in CS 410: Text Information Systems[1]. Specifically, we're going to discuss how LiveDataLab has been used to host the course, provide a novel end-to-end data science assignment, and enable leaderboard competitions for students.

7.1.1 Course Hosting

CS 410: Text Information Systems has been run using LiveDataLab for several semesters now, supporting over two hundred students each semester. Thus far, CS 410 has mostly been run using the preliminary Gitlab version of the system hosted on Microsoft Azure. On the Gitlab platform, a 'Group' is created for the course each semester, and a script is run to generate student accounts and add them to the group. Students are emailed to confirm their account and reset their password upon the creation of their account via a SendGrid[2] integration with the Gitlab system on Azure. Teaching assistants also have accounts created and are added as owners in the course's Gitlab group.

Assignments are deployed to students by creating a new repository within the Gitlab group. Students then fork this repository and begin working off of their fork. When students are ready to submit their code, they push to their repo, triggering Gitlab CI to spin up a virtual machine and execute the students' code. When teaching assistants are ready to grade assignments, they run a script to clone the student repositories to their local machine and run the test cases for each student's code.

Running a large university course on the cloud infrastructure has enabled over a thousand students to complete hands-on style data science assignments without worrying about storing large datasets on their computer and other environment setup tasks. We published a demonstration of this preliminary system at ACM Learning at Scale 2019[4]. In the paper,

[1] https://courses.illinois.edu/schedule/2019/spring/CS/410
[2] https://sendgrid.com/

we discuss the design of the preliminary system, its potential broader impact, and the impact it has had on courses at the University of Illinois thus far.

7.1.2 Novel End-to-End Data Science Assignment

During the Spring 2019 offering of CS 410, we used LiveDataLab to offer a novel assignment exposing students to the end-to-end data science process. In the first stage of the assignment, students wrote web scrapers to collect data on computer science faculty members at universities all around the United States. Then, we aggregated this data to create a new dataset and stored it in LiveDataLab's cloud infrastructure. The next stage of the assignment was exposing the dataset to students in the Dataset Annotation System described in Chapter 5. For this stage, each student submitted a search engine query for the dataset and submitted relevance judgements for the returned faculty member documents. As a result, we had created a new dataset with labels crowd-sourced via the several hundred students enrolled in the course. For the next stage of the assignments, students developed their own ranking functions to be plugged into a search engine based on existing popular datasets such as the Cranfield dataset. Finally, we used LiveDataLab's leaderboard capabilities to host a competition among the students where their ranking functions were evaluated on the computer science faculty dataset they helped create. LiveDataLab enabled this end-to-end assignment to be offered to several hundred students, allowing the students to gain hands-on experience with data science in a practical setting while also creating a novel dataset for finding information on computer science faculty across the nation. Overall, we created 4,720 annotations for 283 unique search engine queries with 6,598 total documents in the dataset. This experience was published in more detail at the ACM Technical Symposium for Computer Science Education 2020[5].

7.1.3 Leaderboard Competitions

Throughout the semesters of supporting CS 410, LiveDataLab's leaderboard functionality has been used to host search engine and classification competitions among the students. Each semester, hundreds of students create their own solution to search engine and classification problems using standard research datasets such as APNews, Cranfield, Trec, and others, and submit their results to the leaderboard in an open competition. LiveDataLab enables all of the students' code to be evaluated without computational restraints on these real-world datasets and without placing heavy responsibility on the students for resource requirements.

7.2 ECE 418: IMAGE AND VIDEO PROCESSING

In this section, we're going to share how LiveDataLab has been used to support ECE 418: Image and Video Processing[3]. Specifically, we're going to focus on how this demonstrates the ability for supporting a variety of subareas within data science, as well as support for different languages in assignments.

7.2.1 General Data Science Support

In the spring 2019 semester at the University of Illinois, LiveDataLab supported ECE 418: Image and Video Processing. This was a large step for LiveDataLab as it was the first time supporting a course that was not based on text datasets. Throughout the semester, nearly one hundred students worked on assignments and submitted code for evaluation on image datasets. This served as the validation that LiveDataLab has the capability for general data science support across many education and research areas.

7.2.2 Multi-Language Support

Alongside supporting image datasets for the first time, this was LiveDataLab's first time officially supporting a language other than Python for assignments. Some of ECE 418's assignments were to be completed in Matlab throughout the semester and not just Python. This experience showed that LiveDataLab's code execution environment was very customizable and robust in being able to support a variety of languages. This was essential for promoting LiveDataLab as a truly generalized system for data science tasks across many areas in a variety of languages.

7.3 CS 510: ADVANCED INFORMATION RETRIEVAL

In this section, we're going to share how LiveDataLab has supported leaderboard competitions with CS 510: Advanced Information Retrieval[4].

7.3.1 Leaderboard Competition

Throughout the semesters, LiveDataLab has also hosted leaderboard competitions for CS 510: Advanced Information Retrieval as well. For these competitions, students created

[3]https://courses.illinois.edu/schedule/2019/spring/ECE/418
[4]https://courses.illinois.edu/schedule/2019/fall/CS/510

more involved and complicated algorithms for data science tasks, and LiveDataLab was able to execute their solutions on its auto-scaling cluster without issue. This showcases LiveDataLab's robust dependency support for a variety of algorithms and toolkits, as well as its ability to execute more intensive solutions.

7.4 CS 410: ONLINE MCS VERSION

In this section, we're going to share how LiveDataLab has supported a completely online offering of CS 410[5] at the University of Illinois.

7.4.1 Support for Large-Scale Online Courses

LiveDataLab has also supported the completely online Masters in Computer Science version of CS 410. Throughout the semesters of this course, we were able to support the hundreds of working professionals enrolled in this course and help them gain a practical data science education. Even in an online course, students were able to participate in the same quality of assignment, competing in leaderboard competitions and implementing state of the art data science algorithms using real toolkits, such as MeTAPy[6]. The online MCS version of CS 410 was LiveDataLab's first step in expanding to the widest audience possible and supporting completely online data science education.

7.5 SUMMARY

We have seen how LiveDataLab has been used to support several courses at the University of Illinois. LiveDataLab has been very effective for offering new kinds of data science assignments with real-world datasets to over one thousand students from both on-campus and online degree programs, without worry of hardware limitations. We have been able to offer a novel end-to-end data science assignment in CS 410: Text Information Systems[5], image processing assignments in ECE 418: Image and Video Processing, leaderboard competition assignments in multiple courses, and offer the same level of assignment quality to a completely online version of CS 410. All of this support is enabled at a relatively low cost of $7.40[1] per student over the course of a semester.

[5]https://courses.illinois.edu/schedule/2019/fall/CS/410
[6]https://github.com/meta-toolkit/metapy

CHAPTER 8: CONCLUSION AND FUTURE WORK

In this chapter, we conclude the thesis with sharing our final thoughts and where there is room for future work on LiveDataLab.

8.1 CONCLUSION

There is a rapidly growing demand for skilled data scientists; however, developing practical data science skills requires access to hands-on data science assignments using real-world datasets. Providing these hands-on assignments has remained an open challenge due to hardware and cost limitations. In this thesis, we proposed LiveDataLab, a cloud-based system for facilitating data science education, research, and industry applications. LiveDataLab provides a novel project-based learning platform for hosting assignments, open projects, research problems, courses, and executing code with any dependencies backed by a novel auto-scaling cloud infrastructure, integrated with a variety of real-world datasets using an Azure cloud file system. Thus far, LiveDataLab has supported over one thousand students in courses at the University of Illinois in both in-person and online degree programs. These students have been able to gain practical data science skills at an affordable cost without worry of compute resources and socioeconomic or geographic background. The results of LiveDataLab have been published at ACM Learning at Scale 2019[4] and ACM SIGCSE 2020[5]. However, these results only showcase the potential of LiveDataLab, which can be immediately used by many other universities around the world to enhance their data science courses. LiveDataLab can bring hands-on, practical data science education to an extremely wide audience, while also creating a big data ecosystem promoting the intersection of education, research, and industry.

8.2 FUTURE WORK

The LiveDataLab initiative[1] has shown promise thus far, yet has much more room for expansion. The core infrastructure and platform of LiveDataLab is built and available for use[2]. The next major step is expansion to more courses at the University of Illinois, courses at other universities, online courses, hosting more open projects for learners of any background to gain experience, and forming relationships with research groups and industry partners

[1] http://livedatalab.org/

[2] http://livedatalab.centralus.cloudapp.azure.com/

to host challenges and datasets for students to compete in. These partnerships will allow students to gain the most practical education possible on real-world challenges, while also allowing research groups and industry partners to potentially discover novel solutions to their problems, as well as identify candidates that would be a good fit for their group. Ultimately, LiveDataLab seeks to bring students, researchers, and application developers together on a single unified platform to collaborate in a highly efficient big data ecosystem where everyone is simultaneously both a contributor and beneficiary.

REFERENCES

[1] C. Geigle, I. Lourentzou, H. Sundaram, and C. Zhai, "Clads: a cloud-based virtual lab for the delivery of scalable hands-on assignments for practical data science education," in *Proceedings of the 23rd Annual ACM Conference on Innovation and Technology in Computer Science Education*. New York, NY, USA: ACM, 2018. [Online]. Available: https://doi.org/10.1145/3197091.3197135 p. 176–181.

[2] X. Yu, "Design and implementation of the search engine module in colds," M.S. thesis, University of Illinois at Urbana-Champaign, Champaign, 2018.

[3] B. Jiang, "Claire: A cloud-based lab for information retrieval," M.S. thesis, University of Illinois at Urbana-Champaign, Champaign, 2018.

[4] A. Green and C. Zhai, "Livedatalab: A cloud-based platform to facilitate hands-on data science education at scale," in *Proceedings of the Sixth (2019) ACM Conference on Learning @ Scale*. New York, NY, USA: ACM, 2019. [Online]. Available: https://doi.org/10.1145/3330430.3333665 pp. 1–2.

[5] A. Boughoula, A. Green, and C. Zhai, "Collective development of large scale data science products via modularized assignments: An experience report," in *Proceedings of the 51st ACM Technical Symposium on Computer Science Education*. New York, NY, USA: ACM, 2020. [Online]. Available: https://doi.org/10.1145/3328778.3366961 pp. 1200–1206.