Project Part 4

Project Title: DataWage Navigator

Team: Group no. 15

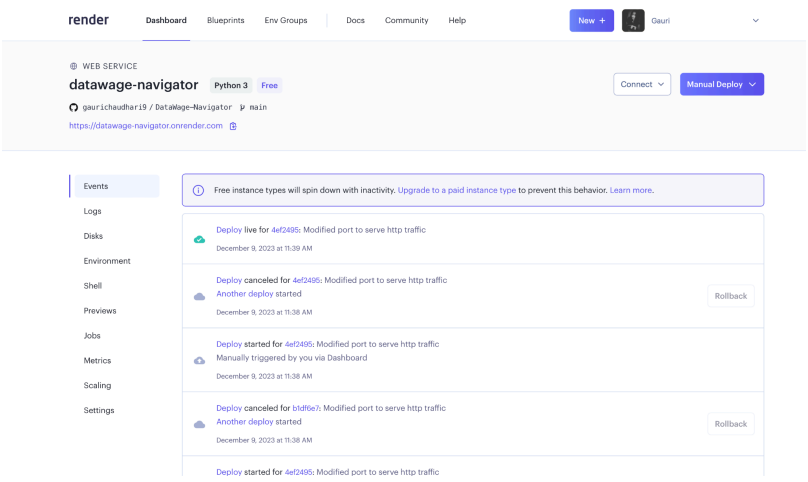
1. Gauri Chaudhari (gchaudh)
2. Shibani Dcosta (sdcosta)
3. Shruti Houji (sghouji)

Section 1

Application URL:

We have hosted the application on Render. Render offers an all-in-one cloud platform for developing and running applications. With its seamless integration with Git, our application auto-deploys effortlessly, streamlining our development process and ensuring smooth operations.

Web URL: https://datawage-navigator.onrender.com



Full Github URL (IU):

https://github.iu.edu/gchaudh/DataWage-Navigator/tree/main/code

For hosting we had to move the repository to our personal github:
https://github.com/gaurichaudhari9/DataWage-Navigator

Video URL: https://iu.mediaspace.kaltura.com/media/t/1_n0zbbrpw

Section 2

Purpose -

Our innovative application, leveraging data from ai-jobs.net, delves into the expansive realm of data science. This comprehensive resource caters to job seekers worldwide, offering detailed insights into career prospects and salary expectations linked to diverse factors such as job roles, experience levels, and company locations. Through our secure database and interactive visualizations, we provide a comprehensive view of the evolving data science landscape from 2020 to 2023. Our app shows new jobs and the salaries offered. It helps job seekers understand the job market better, so they know what to expect in pay and what jobs are growing. With the application, we're set to redefine how professionals navigate the dynamic data science job market.

Usefulness:

Our database application aims to address limitations found in existing tools like levels.fyi and ai-jobs.net. It will offer a user-friendly experience with minimalistic design for easy navigation, focusing on comprehensive salary insights across diverse AI/ML/Data Science roles worldwide. Users can interactively filter and sort data to swiftly access specific information, enabling quick answers to tailored queries. The standout feature will be dynamic visualizations on a dashboard, presenting critical trends and statistics for rapid decision-making. Our target users span from beginners to professionals, hiring managers, recruiters, and career-switchers. While levels.fyi offers broad insights but lacks navigational ease, ai-jobs.net focuses on our domain but lacks immediate analytical capabilities. Our application seeks to bridge these gaps, providing intuitive navigation coupled with in-depth insights for informed decision-making in the AI/ML/Data Science job landscape.

Technical Description:

Data:

The dataset that we have decided to use is a raw single flat file (csv). This dataset, sourced from foorilla LLC's open-source project ai-jobs.net, encompasses 8102 records collected from 2020 to the present. It focuses on job openings and associated salary compensation across diverse fields like Artificial Intelligence, Machine Learning, Natural Language Processing, Computer Vision, Data Engineering, Analytics, and more. These records offer insights into the year of salary payment, experience levels, employment types, job titles, salary figures, currencies, USD equivalents, employee residence countries, remote work ratios, company locations, and sizes. Compiled through an anonymous survey, this dataset aims to foster transparency in AI/ML/Data Science job salaries, aiding individuals seeking roles in these fields to make informed career decisions based on real-world data.

Link: https://ai-jobs.net/salaries/

How did you build your project(tools/database technologies):

The first step of the project was to create the database and relevant tables. The process involves structuring a database using MySQL as the database management system. It begins by establishing a core fact table containing primary records and creating supporting dimension tables to provide additional context or descriptive details. Within the fact table, a primary key is implemented for record uniqueness, while each dimension table is equipped with its own primary key for maintaining unique attributes. Relationships are established by introducing foreign keys in the fact table, referencing the primary keys in dimension tables, facilitating cohesive data retrieval and analysis.

In our project, we've integrated CRUD (Create, Read, Update, Delete) operations to interact with the MySQL database. To facilitate communication between the database and the user interface, we have used Flask, a Python-based web framework, to develop backend APIs. These APIs are designed to perform CRUD operations, enabling the retrieval, manipulation, addition, and deletion of data within the database.

Through these APIs, frontend components can easily communicate with the database, allowing users to interact with the application's features, perform data operations, and receive real-time updates based on their actions.

We've crafted visualizations that offer valuable insights. We have integrated plotly dash platform with our Flask app to build upon the dashboard which uses the same MySQL database in the backend and several callbacks to display correct data on the dashboard. These visuals provide a statistical understanding of various roles,

empowering users with comprehensive perspectives on job trends and dynamics within the Data Science field.

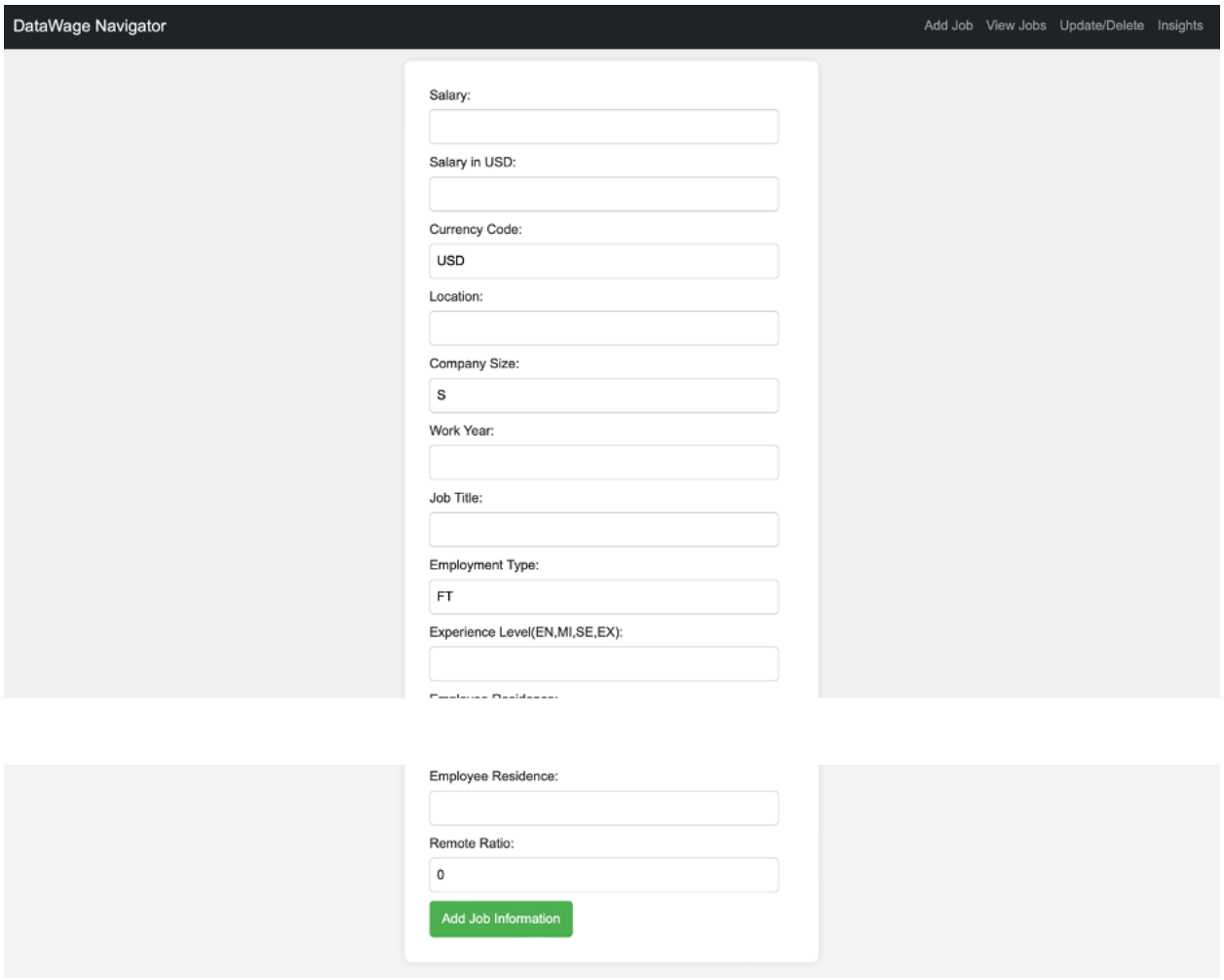
Tech stack:

Front End: HTML, CSS, Bootstrap, JavaScript
Back End: Flask, Plotly Dash
Database: MySql
Tools: Visual Studio Code, MySQL workbench(For testing and setting up database), MAMP server, PHPMyAdmin(For final deployment)

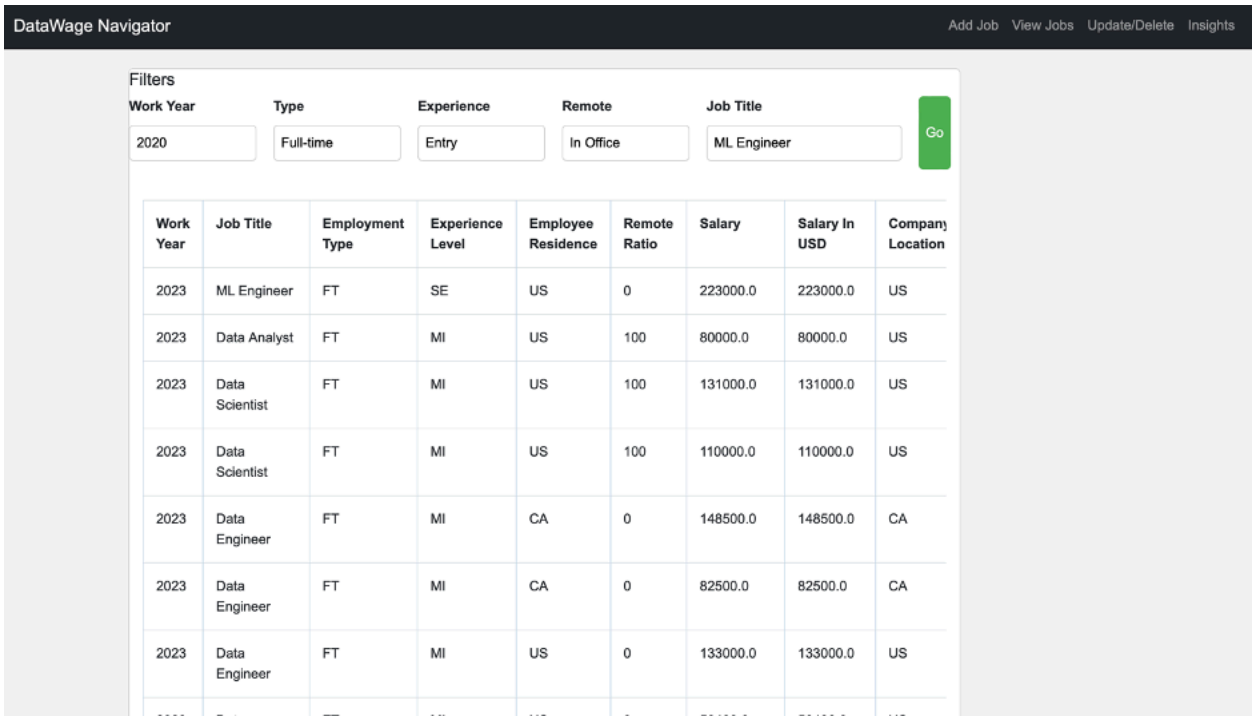
Functionalities:

Below are the functionalities that are offered in the web app.

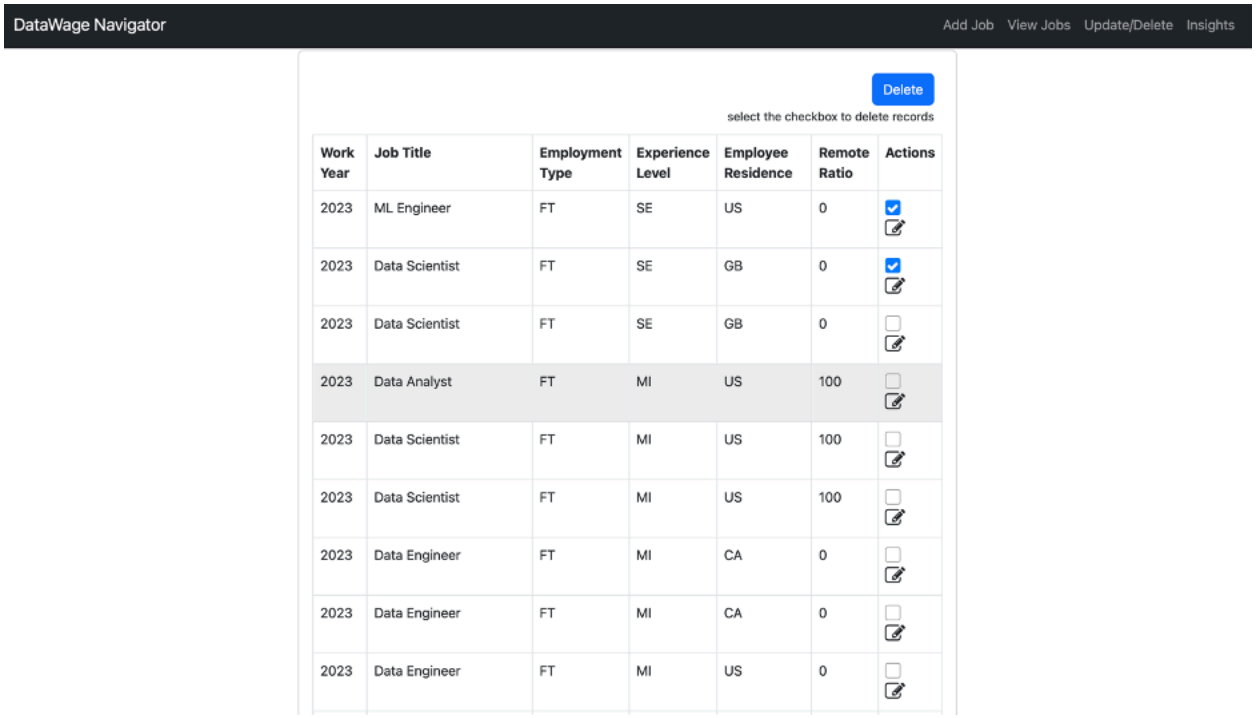
Add a new job: Users can create a new job record by clicking on the Add job button from the nav bar and fill out the form with the required information. On submission of the form, the data will be stored in the database.



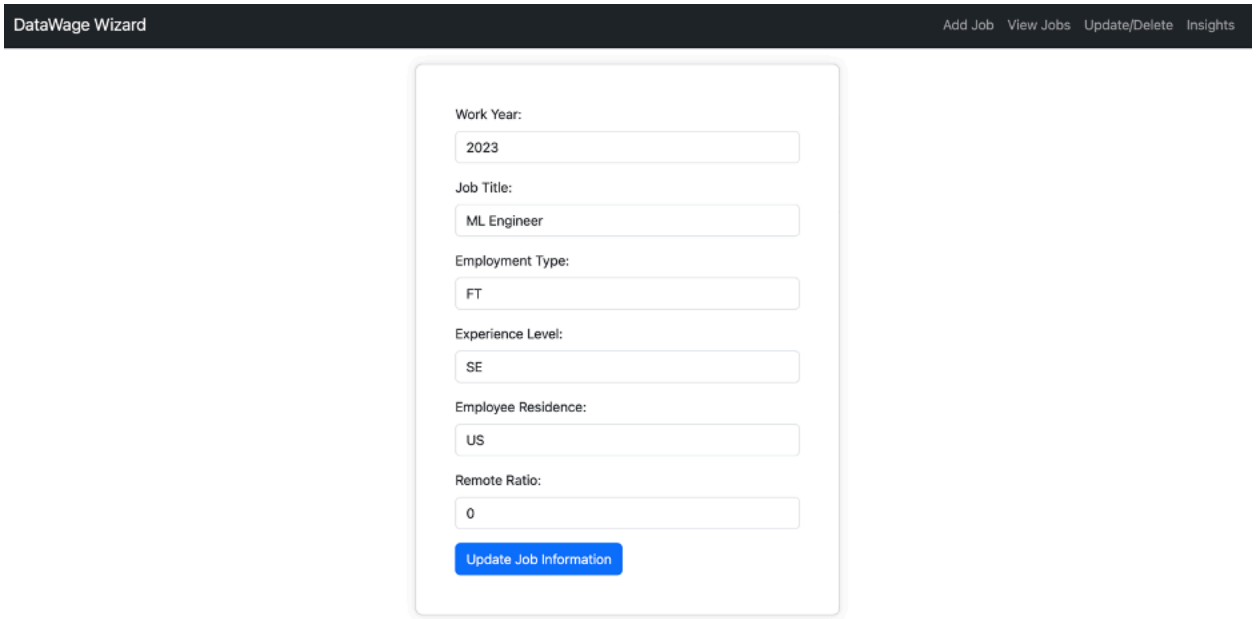
View jobs: Upon selecting 'View Jobs,' users gain access to all available job listings. The page offers filter inputs to customize job views based on specific preferences.



Update/Delete jobs: When users click the "Update/Delete" button, they can select checkboxes to delete multiple jobs. Additionally, by clicking the update icon, they can modify individual records, enabling precise updates to job details.



On click of update icon, below form is displayed in which user can provide inputs to update the job record.



Insights: When users click on "Insights," they gain access to a range of multiselect filters like company size, employment type, remote ratio, work year, experience level, and job title. These filters allow users to generate personalized visualizations specifically tailored to Data Science jobs. The users can view below visual distribution:

- The line graph showcases the distribution of data science jobs by year, providing a chronological perspective.
- Bar graph presents the distribution of roles within the field.
- The insights also include pie charts detailing job distribution based on experience levels and employment types, offering comprehensive insights into the diverse landscape of data science roles.

DataWage Insights - USA (2020-2023)

Company Size: M, L, S

Employment Type: FT, CT, PT, FL

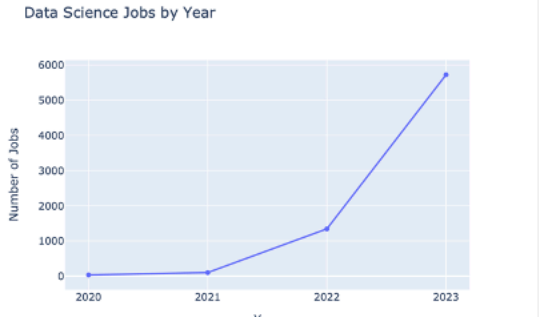
Remote Ratio: 0, 100, 50

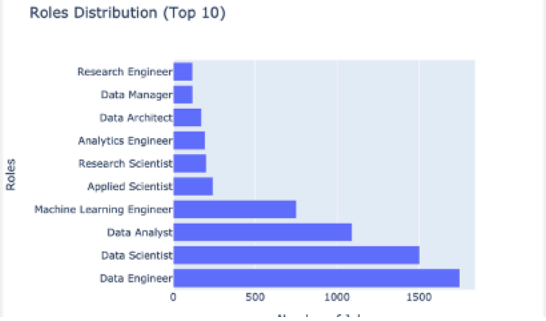
Work Year: 2023, 2022, 2021, 2020

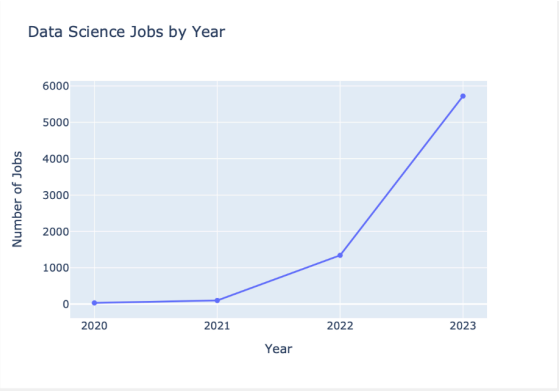
Job Title: All

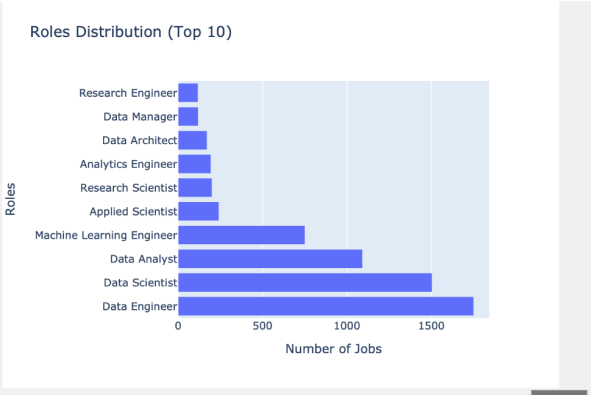
Experience Level: SE, MI, EN, EX

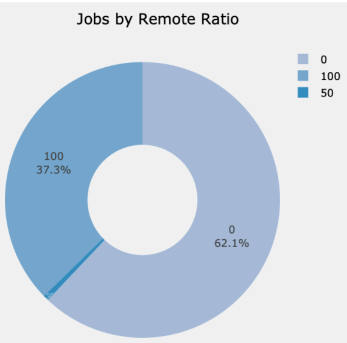
Total Jobs	Median Salary
7196	150000.0

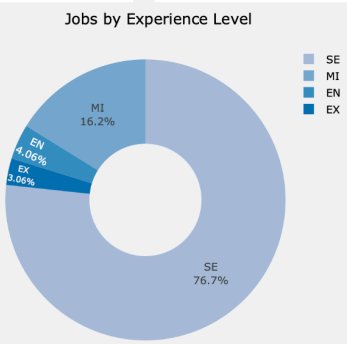


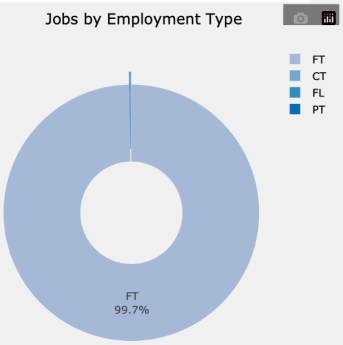












Section 3:

Issues:

- We also faced a few issues while working for the insert queries in the backend while working on the flask because we had 1 fact table and 3 dimension tables, hence once we added the record, it was not getting updated in all the tables. Later on we found out a solution and fixed it
- We face a few difficulties while navigating between the different pages of a web application.
- We faced issues while deploying the web application. We were not able to figure out where we'd host our MySQL database before deploying the application on Render. Hence, later on, we deployed our database on PHPMyAdmin. Hence, we were successfully able to deploy our application on render.

Individual Contributions:

Name	Task	Contribution
Gauri Chaudhari	Dataset collection Database Code Report	● Gather dataset ● Formulate ideas ● Designed attribute datatypes ● Create MySQL database ○ Section: CSV to Database Table ○ Section: Adding primary key to data_jobs ○ Section: Creating Dimension Table - Company Location, Inserting into Company Location, Adding foreign key location_id to data_jobs, updating data_jobs ○ Section: Fact Table - Modifying data_jobs ● Building relevant views ● Testing consistency of database ● Developed Filtering feature on the dynamic table created by Shibani ● Built Insights page using sql queries for analysis and plotly dash framework

		• Proofread report and added missing content/corrected content in sections

Shibani Dcosta	Conceptual Schema
	Code (Application Functionality design)
	Report
		• Formulate ideas
		• Create conceptual schema diagram
		  ○ Before normalization relational schema diagram
		  ○ After normalization relational schema diagram
		• Build views relevant to our application functionality design
		  ○ Section: Relevant to application functionality code - Views
		• Testing consistency of database
		• Designed and developed the UI layout for Add job, Update/Delete, and Update form
		• Created a dynamic table using HTML, CSS, Bootstrap and Javascript to view the job records
		• Worked with Shruti on Flask setup and connecting the database with the application
		• Integrated the apis with the template that was developed
		• Section 1, Section 2 - Report

Shruti Houji	Database
	Code
	Report
		• Formulate ideas
		• Design database constraints
		• Create MySQL database
		  ○ Section: CSV to Database Table
		  ○ Section: Creating Dimension Table - job_information, Inserting into Job Information table
		  ○ Section: Creating Dimension Table - Currency, Inserting into Currency, Adding foreign key currency_id to data_jobs, updating data_jobs
		• Worked on the backend part of the project in flask to insert, update and delete queries
		• Worked with Gauri to deploy the app on render
		• Section 3, Proofread report