**Data-analysis-chatbot Streamlit App with DevOps & Jenkins**

A Project report

Submitted in partial fulfilment of the requirements for the award of a degree of Bachelor of Technology

(Computer Science and Engineering) Submitted to

LOVELY PROFESSIONAL UNIVERSITY PHAGWARA, PUNJAB



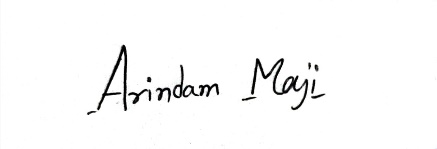
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | | Submitted by | |  |
| Name of the Student | |  | | Name of the Supervisor |
| Arindam Maji  Registration Number:12208405 | |  | | Divya Thakur |
|  |  | |
|  |  | |
|  |  | |
|  |  | |
| **Signature of Student** | |  | | **Signature of Supervisor** |

##### DECLARATION BY STUDENT

###### To whom so ever it may concern.

I, **Arindam Maji-12208405** hereby declare that the work done by me on **"Data-analysis-chatbot Streamlit App with DevOps & Jenkins"** under the supervision of **(Divya Thakur-Assistant Professor)**, Lovely professional University, Phagwara, Punjab, is a record of original work for the partial fulfilment of the requirements for the award of the degree, **in Computer Science Engineering from Lovely Professional University, Phagwara.**

Name of Student : Arindam Maji Registration Number:12208405



Signature of Student Date:02/05/2025

Date:

# DECLARATION BY SUPERVISOR

This is to certify that, **(Divya Thakur-Assistant Professor)** from Lovely Professional University, Phagwara, Punjab, has worked on **"Data-analysis-chatbot Streamlit App with DevOps & Jenkins”** under my supervision from. It is further stated that the work carried out by the student is a record of original work to the best of my knowledge for the partial fulfilment of the requirements for the award of the degree, degree name.

### Signature and Name of the Mentor:

Divya Thakur

**Designation:** Assistant Professor

### School of Computer Science and Engineering,

Lovely Professional University, Phagwara Punjab

**Date:**

## ACKNOWLEDGEMENT

We humbly take this opportunity to present our votes of thanks to all those guideposts who really acted as lightening pillars to enlighten our way throughout this project that has led to successful and satisfactory completion of this study.

We are grateful to our mentor Divya Thakur for providing us with an opportunity to undertake this project in this university and providing us with all the bright and innovative ideas for making our project worthwhile of running in an organization. We are highly thankful for his active support, valuable time and advice, whole-hearted guidance, sincere cooperation, and painstaking involvement during the study and in completing the capstone project within the time stipulated.

We are thankful to all those, particularly our friends, who have been instrumental in creating proper, healthy, and conductive environment and including new and fresh innovative ideas for us during the project, without their help, it would have been extremely difficult for us to prepare the project in a time bound framework.

###### Arindam Maji

Bachelor of Technology (Computer Science and Engineering)

# LIST OF CONTENT

|  |  |  |
| --- | --- | --- |
| **S. No.** | **Title** | **Page No.** |
| 1 | Declaration by student | 2 |
| 2 | Declaration by supervisor | 3 |
| 3 | Acknowledgement | 4 |
| 4 | Abstract | 6 |
| 5 | Introduction | 7 |
| 6 | Literature Survey | 8 |
| 7 | Existing System | 9 |
| 8 | Proposed System | 10 |
| 9 | System Architecture | 11 |
| 10 | Hardware and Software Requirements | 12 |
| 11 | System Study / Feasibility Study | 13 |
| 12 | System Design | 14 |
| 13 | Implementation | 15 |
| 14 | Software Environment | 16-21 |
| 15 | Machine Learning | 22-30 |
| 16 | System Testing | 31-32 |
| 17 | Screenshots of Working Project | 33-35 |
| 18 | Conclusion | 36 |
| 19 | References | 37 |

**ABSTRACT:**

This project presents the design and implementation of a **Data Analytics Chatbot** that leverages **Natural Language Processing (NLP)** and **machine learning techniques** to enable users to perform exploratory data analysis through a conversational interface. The system is developed using **Streamlit** for the frontend and integrates **Hugging Face transformer models** to interpret and process user queries.

The chatbot accepts CSV files uploaded by the user and performs rule-based as well as intelligent analyses, such as identifying missing values, generating summary statistics, and creating visualizations like bar charts and histograms. The model classifies user intent and maps it to specific data operations, enabling a seamless and intuitive user experience.

Moreover, the project incorporates **Docker** for containerized deployment and **Jenkins** to establish a **CI/CD pipeline**, ensuring that the system can be tested, updated, and deployed automatically. This infrastructure ensures maintainability, scalability, and readiness for real-world use. The solution bridges the gap between non-technical users and complex data analysis tools, promoting accessibility and automation in the field of data science.

# INTRODUCTION:

In the age of big data, organizations and individuals are constantly dealing with vast amounts of information. While numerous tools exist for analyzing and visualizing data, many of them require technical expertise or steep learning curves. This creates a barrier for non-programmers who need to derive insights from datasets.

This project addresses that challenge by proposing a **Data Analytics Chatbot**—a system that allows users to perform data analysis using natural language inputs. Users simply upload a dataset (in CSV format) and type their analytical query in plain English. The chatbot understands the query using NLP, analyzes the data accordingly, and presents the results in an interpretable format, including visualizations.

Key motivations for developing this system include:

* Lowering the entry barrier for data analysis.
* Providing a conversational, intuitive interface.
* Ensuring scalable deployment through modern DevOps practices.

The solution combines multiple open-source technologies to offer an integrated, user-friendly analytics platform.

# LITERATURE SURVEY:

The concept of combining chatbot technology with data analysis intersects various research areas. This section reviews prior work and technologies relevant to the project.

**6.1 Chatbot Technology**

Chatbots have evolved from rule-based systems to AI-powered interfaces capable of understanding context and performing actions. Platforms like Rasa and Dialogflow offer customizable chatbots, while transformers enable more nuanced understanding. Chatbots are now widely used in customer service, healthcare, education, and analytics.

**6.2 Natural Language Processing (NLP)**

NLP is the backbone of modern AI chatbots. Pretrained transformer models, such as BERT, RoBERTa, and DistilBERT, have enabled breakthroughs in contextual understanding. These models can classify user intent and extract relevant entities from queries.

**6.3 Rule-Based Data Analysis**

In traditional analytics, rule-based methods provide deterministic ways to assess and transform data. For instance, missing value detection, computing aggregates (mean, median), and identifying outliers can all be programmed through logical rules.

**6.4 Streamlit Framework**

Streamlit is a Python-based framework that enables the creation of interactive, web-based dashboards with minimal effort. It is widely adopted in data science projects for prototyping and deploying data apps.

**6.5 Hugging Face Transformers**

Hugging Face is a leading platform providing state-of-the-art transformer models for NLP tasks. Their Python library simplifies integration and supports fine-tuning for custom intent recognition.

**6.6 CI/CD with Jenkins & Docker**

Jenkins enables Continuous Integration and Continuous Deployment, facilitating reliable, automated updates. Docker ensures consistent deployment by packaging the application with its dependencies, making it portable across environments.

These technologies serve as the foundation for building an intelligent and scalable analytics chatbot.

# EXISTING SYSTEM:

Existing data analytics tools such as Excel, Tableau, and Power BI allow users to perform data visualizations and analysis through graphical interfaces. However, they have limitations:

* Require manual setup of data fields and visuals.
* Often demand intermediate-level technical knowledge.
* Do not support natural language queries.
* Lack real-time NLP-driven analysis.

These systems are efficient for users familiar with them, but pose challenges for non-technical individuals who seek quick insights from raw data. Furthermore, existing systems may involve licensing costs and do not integrate automated deployment and testing pipelines.

# PROPOSED SYSTEM:

The proposed **Data Analytics Chatbot** aims to address the shortcomings of existing systems by offering:

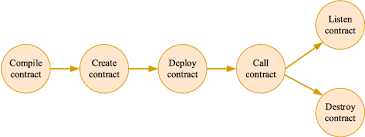
* A simple UI to upload CSV datasets.
* NLP-powered interpretation of user queries.
* Automatic execution of rule-based and model-based data analysis.
* Dynamic generation of visual insights.
* Containerized deployment using Docker.
* CI/CD automation with Jenkins.

**Key Benefits:**

* No programming knowledge required.
* Open-source and cost-effective.
* Intuitive user experience through conversational interface.
* Scalable and production-ready via Docker + Jenkins.

This approach democratizes access to data analytics and provides a streamlined workflow for users with diverse technical backgrounds.

SYSTEM ARCHITECTURE:

****

### Components:

* **Frontend**: Streamlit web app
* **Backend**: Python scripts (utils.py)
* **NLP Engine**: Hugging Face transformer
* **Analysis Layer**: Pandas, Matplotlib
* **CI/CD**: Jenkins
* **Deployment**: Docker

# HARDWARE & SOFTWARE REQUIREMENTS:

### Hardware:

* Processor: Intel i5 or higher
* RAM: 8GB minimum
* Disk: 20GB free space

### Software:

* OS: Windows/Linux/MacOS
* Python 3.8+
* Streamlit
* Hugging Face Transformers
* Jenkins
* Docker

# SYSTEM STUDY FEASIBILITY STUDY

The feasibility of the project is analyzed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential.

Three key considerations involved in the feasibility analysis are

* + ECONOMICAL FEASIBILITY
  + TECHNICAL FEASIBILITY
  + SOCIAL FEASIBILITY

### Technical Feasibility:

The technologies used are open-source and widely supported.

### Operational Feasibility:

Intuitive UI ensures ease of use for analysts and non-programmers.

### Economic Feasibility:

Low-cost system using free tools; Docker and Jenkins ensure cost-effective deployment.

# SYSTEM DESIGN :

### 9.1 Architecture Diagram

The system consists of the following key components:

* **Frontend (Streamlit):** Provides an interface for uploading data and interacting with the chatbot.
* **Backend (utils.py):** Handles data loading, rule-based checks, NLP model inference, and visualization generation.
* **NLP Model:** Hugging Face transformer to classify user intent.
* **CI/CD Tools:** Docker and Jenkins for testing and deployment.

### 9.2 Data Flow

1. User uploads CSV via Streamlit UI.
2. User types query in plain English.
3. Query is sent to the Hugging Face model for intent classification.
4. Based on intent, rule-based logic or analytical functions are executed.
5. Results (text/visualizations) are rendered back in the Streamlit app.

# IMPLEMENTATION

### 10.1 File Structure

├── app.py # Streamlit UI for the chatbot

├── utils.py # Logic for data handling and analysis

├── requirements.txt # Project dependencies

├── Dockerfile # Docker image definition

├── Jenkinsfile # CI/CD pipeline configuration

├── model/ # Pretrained/fine-tuned Hugging Face models

│ └── intent\_model.pt

└── sample\_data/

└── sample.csv # Sample input CSV for testing

### 10.2 Key Code Snippets

* **app.py** sets up the Streamlit UI, handles file uploads and user input, and displays responses.
* **utils.py** loads the CSV, interprets queries, applies rules or calls model, and generates plots.

### 10.3 Technologies Used

* **Python 3.8+**
* **Streamlit**
* **Hugging Face Transformers**
* **Matplotlib / Seaborn** for visualizations
* **Docker & Jenkins** for deployment

# SOFTWARE ENVIRONMENT

###### What is Python :

Below are some facts about Python.

Python is currently the most widely used multi-purpose, high-level programming language.

Python allows programming in Object-Oriented and Procedural paradigms. Python programs generally are smaller than other programming languages like Java.

Programmers have to type relatively less and indentation requirement of the language, makes them readable all the time.

Python language is being used by almost all tech-giant companies like – Google,

Amazon, Facebook, Instagram, Dropbox, Uber… etc.

The biggest strength of Python is huge collection of standard library which can be used for the following –

* [Machine Learning](https://www.geeksforgeeks.org/machine-learning/)
* GUI Applications (like Kivy, Tkinter, PyQt etc. )
* Web frameworks like Django (used by YouTube, Instagram, Dropbox)
* Image processing (like Opencv, Pillow)
* Web scraping (like Scrapy, BeautifulSoup, Selenium)
* Test frameworks
* Multimedia

### Advantages of Python :

Let’s see how Python dominates over other languages.

#### Extensive Libraries

Python downloads with an extensive library and it *contain code for various purposes like regular expressions, documentation-generation, unit-testing, web browsers, threading, databases, CGI, email, image manipulation, and more.* So, we don’t have to write the complete code for that manually.

#### Extensible

As we have seen earlier, Python can be **extended to other languages**. You can write some of your code in languages like C++ or C. This comes in handy, especially in projects.

#### Embeddable

Complimentary to extensibility, Python is embeddable as well. You can put your Python code in your source code of a different language, like C++. This lets us add **scripting capabilities** to our code in the other language.

#### Improved Productivity

The language’s simplicity and extensive libraries render programmers **more productive** than languages like Java and C++ do. Also, the fact that you need to write less and get more things done.

#### IOT Opportunities

Since Python forms the basis of new platforms like Raspberry Pi, it finds the future bright for the Internet Of Things. This is a way to connect the language with the real world.

When working with Java, you may have to create a class to print **‘Hello World’**. But in Python, just a print statement will do. It is also quite **easy to learn, understand,** and **code.** This is why when people pick up Python, they have a hard time adjusting to other more verbose languages like Java.

#### Readable

Because it is not such a verbose language, reading Python is much like reading English. This is the reason why it is so easy to learn, understand, and code. It also does not need curly braces to define blocks, and **indentation is mandatory.** This further aids the readability of the code.

#### Object-Oriented

This language supports both the **procedural and object-oriented** programming paradigms. While functions help us with code reusability, classes and objects let us model the real world. A class allows the **encapsulation of data** and functions into one.

#### Free and Open-Source

Like we said earlier, Python is **freely available.** But not only can you [**download Python**](https://data-flair.training/blogs/install-python-windows/)for free, but you can also download its source code, make changes to it, and even distribute it. It downloads with an extensive collection of libraries to help you with your tasks.

#### Portable

When you code your project in a language like C++, you may need to make some changes to it if you want to run it on another platform. But it isn’t the same with Python. Here, you need to **code only once**, and you can run it anywhere. This is called **Write Once Run Anywhere (WORA)**. However, you need to be careful enough not to include any system-dependent features.

#### Interpreted

Lastly, we will say that it is an interpreted language. Since statements are executed one by one, **debugging is easier** than in compiled languages.

*Any doubts till now in the advantages of Python? Mention in the comment section.*

# Advantages of Python Over Other Languages :

#### Less Coding

Almost all of the tasks done in Python requires less coding when the same task is done in other languages. Python also has an awesome standard library support, so you don’t have to search for any third-party libraries to get your job done. This is the reason that many people suggest learning Python to beginners.

#### Affordable

Python is free therefore individuals, small companies or big organizations can leverage the free available resources to build applications. Python is popular and widely used so it gives you better community support.

**The 2019 Github annual survey showed us that Python has overtaken Java in the most popular programming language category.**

#### Python is for Everyone

Python code can run on any machine whether it is Linux, Mac or Windows. Programmers need to learn different languages for different jobs but with Python, you can professionally build web apps, perform data analysis and [**machine learning**](https://data-flair.training/blogs/machine-learning-tutorials-home/), automate things, do web scraping and also build games and powerful visualizations. It is an all-rounder programming language.

#### Disadvantages of Python

So far, we’ve seen why Python is a great choice for your project. But if you choose it, you should be aware of its consequences as well. Let’s now see the downsides of choosing Python over another language.

###### Speed Limitations

We have seen that Python code is executed line by line. But since [Python](https://www.python.org/) is interpreted, it often results in **slow execution**. This, however, isn’t a problem unless speed is a focal point for the project. In other words, unless high speed is a requirement, the benefits offered by Python are enough to distract us from its speed limitations.

###### Weak in Mobile Computing and Browsers

While it serves as an excellent server-side language, Python is much rarely seen on the **client- side**. Besides that, it is rarely ever used to implement smartphone-based applications. One such application is called **Carbonnelle**.

The reason it is not so famous despite the existence of Brython is that it isn’t that secure.

###### Design Restrictions

As you know, Python is **dynamically-typed**. This means that you don’t need to declare the type of variable while writing the code. It uses **duck-typing**. But wait, what’s that? Well, it just means that if it looks like a duck, it must be a duck. While this is easy on the programmers during coding, it can **raise run-time errors**.

###### Underdeveloped Database Access Layers

Compared to more widely used technologies like **JDBC (Java DataBase Connectivity)** and **ODBC (Open DataBase Connectivity)**, Python’s database access layers are a bit underdeveloped. Consequently, it is less often applied in huge enterprises.

###### Simple

No, we’re not kidding. Python’s simplicity can indeed be a problem. Take my example. I don’t do Java, I’m more of a Python person. To me, its syntax is so simple that the verbosity of Java code seems unnecessary.

This was all about the Advantages and Disadvantages of Python Programming Language.

#### History of Python : -

Both the alphabet and Python programming language begin with the same initial letter ABC. Right, both start with ABC. The programming language ABC represents the meaning of ABC when discussed within the Python framework. ABC represents an Amsterdam-based general- purpose programming language along with programming environment which developed at CWI (Centrum Wiskunde & Informatica) in the Netherlands. The main success of ABC became the inspiration that led Python developers to create their design. Python started developing during the late 1980s. During the time that Guido van Rossum worked on the Amoeba distributed operating system project at CWI he was also based in the CWI facilities. Guido van Rossum described his work experience in the early 1980s as an implementer at Centrum voor Wiskunde en Informatica (CWI) while developing "ABC" through an interview with Bill Venners1. The extent to which ABC impacted Python remains unknown to most people. I discuss ABC's importance due to my many learnings during that project together with the team members I worked with. The main objective of my scripting language endeavor was to adopt ABC strengths while eliminating its problems. So I started typing. During the development I wrote basic code for a simple virtual machine combined with a straightforward runtime system and parser. With the ABC parts I appreciated I built my own versions of the elements. The syntax design included indentations for grouping statements without utilize of curly braces or begin-end blocks and I created four central data types comprising dictionaries and lists along with strings and numbers.

# What is Machine Learning

The following analysis covers machine learning methods without first defining what we mean by machine learning and its limits. Artificial intelligence includes machine learning as its subfield although its initial classification might prove confusing. Research into machine learning came from artificial intelligence investigations but data science practice benefits more from viewing machine learning as model-building for data application.

The core concept of machine learning includes developing mathematical procedures which analyze data for understanding purposes. The introduction of adjustable parameters allows our models to connect with data points and this enables programmers to characterize the process as "data-driven learning." The models' application on previously seen data enables predicting and understanding newly gathered data. I will let readers explore the philosophical question about how model-based mathematical learning compares with human brain learning.

#### Categories Of Machine Leaning :-

At the most fundamental level, machine learning can be categorized into two main types: supervised learning and unsupervised learning.

*Supervised learning:* Imitating the pattern that connects data features to their corresponding labels forms the basis of supervised learning because the developed model then gets applied to make predictions for new and unobserved data. Supervised learning contains two separate task types known as classification tasks and regression tasks where classification requires discrete categories as labels but regression uses continuous quantities for labeling. Two varieties of supervised learning appear in the upcoming examples.

*Unsupervised learning* allows the dataset to showcase itself by modeling data features during procedures that omit label information. This grouping contains models which conduct clustering tasks alongside operations that reduce data dimensions. The two categories of unsupervised learning algorithms perform data classification by grouping techniques and by simplifying complex data structures into enhanced compact data models. The next part demonstrates both unsupervised learning approaches.

Earth contains no other species more developed than humans since humans possess the ability to consider problems and perform objective evaluation to find solutions. The emergence of AI remains at a primary level since it has not reached human-level intelligence in several domains. What reasons exist to implement programming capabilities in machines? An ideal justification of this practice exists in the ability to construct efficient scalable decisions from substantial data sets.

Business organizations are spending significant funds to acquire artificial intelligence and machine learning and deep learning technologies to extract essential information from data for executing practical operations and solving complex issues. The practice of machine-based data- driven choices serves as automated process handling through which we can refer to as data- driven automated solutions. The data-based choice function operates better than programming logic because it addresses inherent unprogrammable issues. Our society still depends on human intelligence, yet we want to solve big-scale real-world problems with maximum efficiency. The requirement for machine learning develops because of this reason.

#### Challenges in Machines Learning :-

While Machine Learning is rapidly evolving, making significant strides with cybersecurity and autonomous cars, this segment of AI as whole still has a long way to go. The reason behind is that ML has not been able to overcome number of challenges. The challenges that ML is facing currently are −

**Quality of data** − Having good-quality data for ML algorithms is one of the biggest challenges. Use of low-quality data leads to the problems related to data preprocessing and feature extraction.

**Time-Consuming task** − Another challenge faced by ML models is the consumption of time

especially for data acquisition, feature extraction and retrieval.

**Lack of specialist persons** − As ML technology is still in its infancy stage, availability of

expert resources is a tough job.

**No clear objective for formulating business problems** − Having no clear objective and well- defined goal for business problems is another key challenge for ML because this technology is not that mature yet.

**Issue of overfitting & underfitting** − If the model is overfitting or underfitting, it cannot be

represented well for the problem.

**Curse of dimensionality** − Another challenge ML model faces is too many features of data

points. This can be a real hindrance.

**Difficulty in deployment** − Complexity of the ML model makes it quite difficult to be

deployed in real life.

### Applications of Machines Learning :-

Machine Learning is the most rapidly growing technology and according to researchers we are in the golden year of AI and ML. It is used to solve many real-world complex problems which cannot be solved with traditional approach. Following are some real-world applications of ML

−

* Emotion analysis
* Sentiment analysis
* Error detection and prevention
* Weather forecasting and prediction
* Stock market analysis and forecasting
* Speech synthesis
* Speech recognition
* Customer segmentation
* Object recognition
* Fraud detection
* Fraud prevention
* Recommendation of products to customer in online shopping

### Advantages of Machine learning:-

###### Easily identify trends and patterns -

Machine Learning can review large volumes of data and discover specific trends and patterns that would not be apparent to humans. For instance, for an e-commerce website like Amazon, it serves to understand the browsing behaviors and purchase histories of its users to help cater to the right products, deals, and reminders relevant to them. It uses the results to reveal relevant advertisements to them.

###### No human intervention needed (automation)

With ML, you don’t need to babysit your project every step of the way. Since it means giving machines the ability to learn, it lets them make predictions and also improve the algorithms on their own. A common example of this is anti-virus softwares; they learn to filter new threats as they are recognized. ML is also good at recognizing spam.

### Continuous Improvement

As [**ML algorithms**](https://data-flair.training/blogs/machine-learning-algorithms/)gain experience, they keep improving in accuracy and efficiency. This lets them make better decisions. Say you need to make a weather forecast model. As the amount of data you have keeps growing, your algorithms learn to make more accurate predictions faster.

### Handling multi-dimensional and multi-variety data

Machine Learning algorithms are good at handling data that are multi-dimensional and multi- variety, and they can do this in dynamic or uncertain environments.

### Wide Applications

You could be an e-tailer or a healthcare provider and make ML work for you. Where it does apply, it holds the capability to help deliver a much more personal experience to customers while also targeting the right customers.

### Disadvantages of Machine Learning :-

###### Data Acquisition

Machine Learning requires massive data sets to train on, and these should be inclusive/unbiased, and of good quality. There can also be times where they must wait for new data to be generated.

### Time and Resources

ML needs enough time to let the algorithms learn and develop enough to fulfill their purpose with a considerable amount of accuracy and relevancy. It also needs massive resources to function. This can mean additional requirements of computer power for you.

### Interpretation of Results

Another major challenge is the ability to accurately interpret results generated by the algorithms. You must also carefully choose the algorithms for your purpose.

### High error-susceptibility

[Machine Learning](https://en.wikipedia.org/wiki/Machine_learning) is autonomous but highly susceptible to errors. Suppose you train an algorithm with data sets small enough to not be inclusive. You end up with biased predictions coming from a biased training set. This leads to irrelevant advertisements being displayed to customers. In the case of ML, such blunders can set off a chain of errors that can go undetected for long periods of time. And when they do get noticed, it takes quite some time to recognize the source of the issue, and even longer to correct it.

### Python Development Steps : -

Guido Van Rossum published the first version of Python code (version 0.9.0) at alt.sources in February 1991. This release included already exception handling, functions, and the core data types of list, dict, str and others. It was also object oriented and had a module system. Python version 1.0 was released in January 1994. The major new features included in this release were the functional programming tools lambda, map, filter and reduce, which Guido Van

Rossum never liked.Six and a half years later in October 2000, Python 2.0 was introduced. This release included list comprehensions, a full garbage collector and it was supporting unicode.Python flourished for another 8 years in the versions 2.x before the next major release as Python 3.0 (also known as "Python 3000" and "Py3K") was released. Python 3 is not backwards compatible with Python 2.x. The emphasis in Python 3 had been on the removal of duplicate programming constructs and modules, thus fulfilling or coming close to fulfilling the 13th law of the Zen of Python: "There should be one -- and preferably only one -- obvious way to do it."Some changes in Python 7.3:

* Print is now a function
* Views and iterators instead of lists
* The rules for ordering comparisons have been simplified. E.g. a heterogeneous list cannot be sorted, because all the elements of a list must be comparable to each other.
* There is only one integer type left, i.e. int. long is int as well.
* The division of two integers returns a float instead of an integer. "//" can be used to have the "old" behaviour.
* Text Vs. Data Instead Of Unicode Vs. 8-bit

### Purpose :-

We demonstrated that our approach enables successful segmentation of intra-retinal layers— even with low-quality images containing speckle noise, low contrast, and different intensity ranges throughout—with the assistance of the ANIS feature.

### Python

Python is an interpreted high-level programming language for general-purpose programming. Created by Guido van Rossum and first released in 1991, Python has a design philosophy that emphasizes code readability, notably using significant whitespace.

Python features a dynamic type system and automatic memory management. It supports multiple programming paradigms, including object-oriented, imperative, functional and procedural, and has a large and comprehensive standard library.

* Python is Interpreted − Python is processed at runtime by the interpreter. You do not need to

compile your program before executing it. This is similar to PERL and PHP.

* Python is Interactive − you can actually sit at a Python prompt and interact with the interpreter

directly to write your programs.

Python also acknowledges that speed of development is important. Readable and terse code is part of this, and so is access to powerful constructs that avoid tedious repetition of code. Maintainability also ties into this may be an all but useless metric, but it does say something about how much code you have to scan, read and/or understand to troubleshoot problems or tweak behaviors. This speed of development, the ease with which a programmer of other languages can pick up basic Python skills and the huge standard library is key to another area where Python excels. All its tools have been quick to implement, saved a lot of time, and several of them have later been patched and updated by people with no Python background - without breaking.

**Modules Used in Project :-**

### Tensorflow

TensorFlow is a [free](https://en.wikipedia.org/wiki/Free_software) and [open-source](https://en.wikipedia.org/wiki/Open-source_software) [software library for dataflow and differentiable](https://en.wikipedia.org/wiki/Library_(computing)) [programming](https://en.wikipedia.org/wiki/Library_(computing)) across a range of tasks. It is a symbolic math library, and is also used for [machine](https://en.wikipedia.org/wiki/Machine_learning) [learning](https://en.wikipedia.org/wiki/Machine_learning) applications such as [neural networks](https://en.wikipedia.org/wiki/Neural_networks). It is used for both research and production at [Google.](https://en.wikipedia.org/wiki/Google)

TensorFlow was developed by the [Google Brain](https://en.wikipedia.org/wiki/Google_Brain) team for internal Google use. It was released under the [Apache 2.0](https://en.wikipedia.org/wiki/Apache_License) [open-source license](https://en.wikipedia.org/wiki/Open-source_license) on November 9, 2015.

### Numpy

Numpy is a general-purpose array-processing package. It provides a high-performance multidimensional array object, and tools for working with these arrays.

It is the fundamental package for scientific computing with Python. It contains various features including these important ones:

* A powerful N-dimensional array object
* Sophisticated (broadcasting) functions
* Tools for integrating C/C++ and Fortran code
* Useful linear algebra, Fourier transform, and random number capabilities

Besides its obvious scientific uses, Numpy can also be used as an efficient multi-dimensional container of generic data. Arbitrary data-types can be defined using Numpy which allows Numpy to seamlessly and speedily integrate with a wide variety of databases.

### Pandas

Pandas functions as a free open-source system for programming in Python that delivers quick data handling operations through sophisticated data structures. Python served primarily as the tool for handling big data and making it workable. The data analysis segment received minimal assistance from the software. Pandas solved this problem. The Pandas library enables users to execute five standard processing steps for data analysis between the phases of data load and analysis. Python worked alongside Pandas to serve users throughout academic institutions and commercial financial sectors as well as analytical and economic organizations and statistical laboratories.

### Matplotlib

The Python 2D plotting library Matplotlib generates published-quality images in extensive hard-copy formats as well as interactive platforms for multiple operating systems. Matplotlib offers functionality for Python scripts as well as Python shells and Jupyter Notebook applications and web servers and four graphical user interface toolkits. The Matplotlib library achieves simple operations with simplicity yet allows complex functions through accessibility. The code requires a few lines to create both plots and histograms along with power spectra and bar charts and error charts and scatter plots. You can find multiple examples in the sample plots and thumbnail gallery.

The pyplot module functions as a MATLAB-based plotting platform through its convenient interface when used along with IPython. The power user obtains full control of line styles as well as font properties and axes properties through an object-oriented interface or they can access similar functions which MATLAB users will recognize.

### Scikit – learn

Scikit-learn provides a range of supervised and unsupervised learning algorithms via a consistent interface in Python. It is licensed under a permissive simplified BSD license and is distributed under many Linux distributions, encouraging academic and commercial use.

###### Python

Python is an interpreted high-level programming language for general-purpose programming. Created by Guido van Rossum and first released in 1991, Python has a design philosophy that emphasizes code readability, notably using significant whitespace.

Python features a dynamic type system and automatic memory management. It supports multiple programming paradigms, including object-oriented, imperative, functional and procedural, and has a large and comprehensive standard library.

* Python is Interpreted − Python is processed at runtime by the interpreter. You do not need to

compile your program before executing it. This is similar to PERL and PHP.

* Python is Interactive − you can actually sit at a Python prompt and interact with the interpreter

directly to write your programs.

Python also acknowledges that speed of development is important. Readable and terse code is part of this, and so is access to powerful constructs that avoid tedious repetition of code. Maintainability also ties into this may be an all but useless metric, but it does say something about how much code you have to scan, read and/or understand to troubleshoot problems or tweak behaviors. This speed of development, the ease with which a programmer of other languages can pick up basic Python skills and the huge standard library is key to another area where Python excels. All its tools have been quick to implement, saved a lot of time, and several of them have later been patched and updated by people with no Python background - without breaking.

SYSTEM TEST

The main reason behind testing activity is to find mistakes. A work product testing procedure aims to detect all possible defects and weaknesses within it. Testing involves subjecting Software products and their components through exercises in order to verify their alignment with specifications as well as user needs and prevent system breakdowns. The testing field contains different categories of evaluation. A different test type exists to evaluate a unique testing concern.

## TYPES OF TESTS

### 16.1 Unit Testing

Functions in utils.py tested with mock data.

**16.2 UI Testing**

Streamlit components tested manually.

**16.3 Integration Testing**

Tested full pipeline from input → output.

Test strategy and approach

Field testing will be performed manually and functional tests will be written in detail.

###### Test objectives

* All field entries must work properly.
* Pages must be activated from the identified link.
* The entry screen, messages and responses must not be delayed.

###### Features to be tested

* Verify that the entries are of the correct format
* No duplicate entries should be allowed
* All links should take the user to the correct page.

### Integration Testing

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects.

The task of the integration test is to check that components or software applications, e.g. components in a software system or – one step up – software applications at the company level – interact without error.

**Test Results:** All the test cases mentioned above passed successfully. No defects encountered.

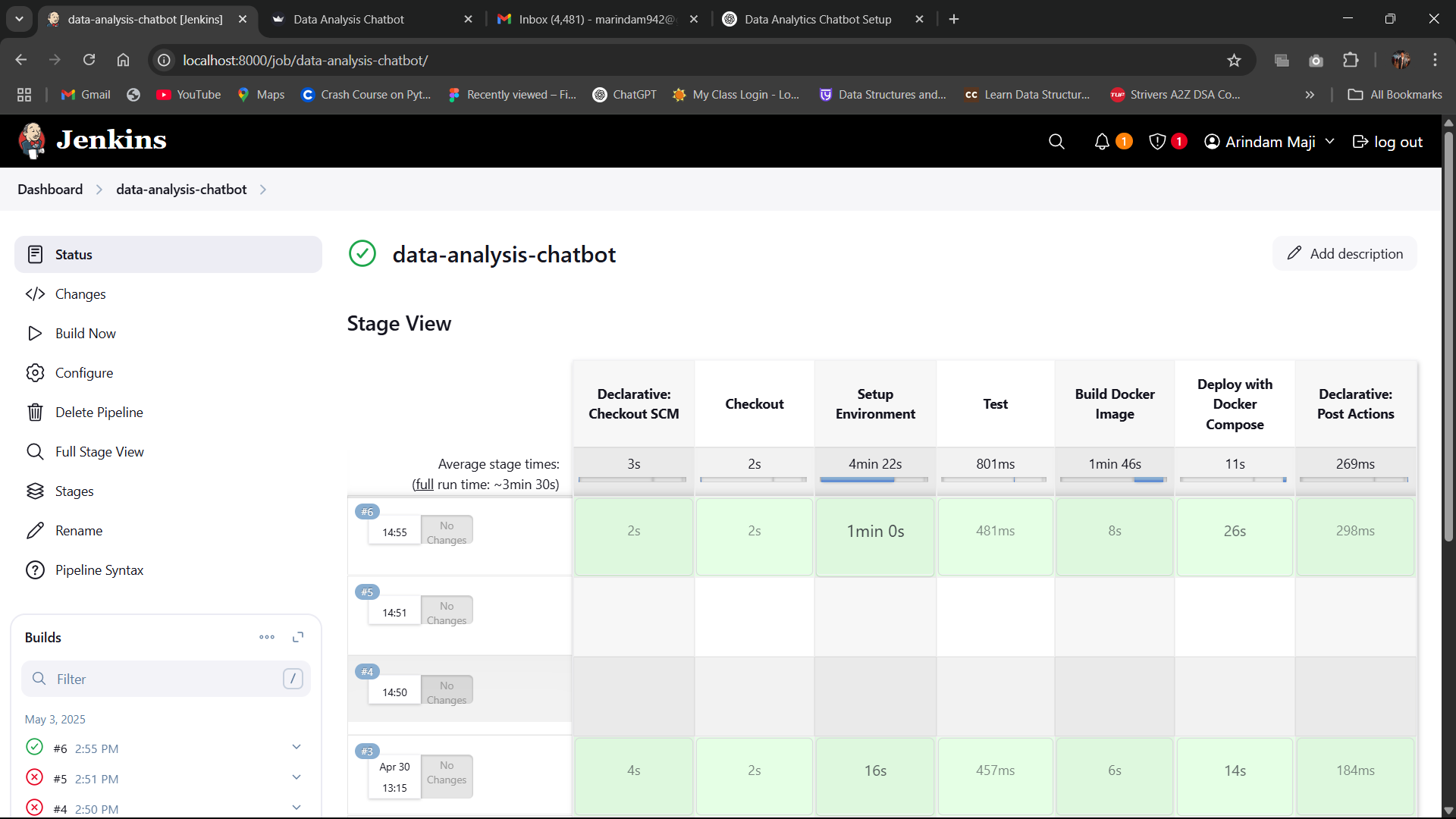
### Acceptance Testing

User Acceptance Testing is an important stage of any project and needs active involvement of the end user. It also makes sure that the system fulfills the functional requirements.

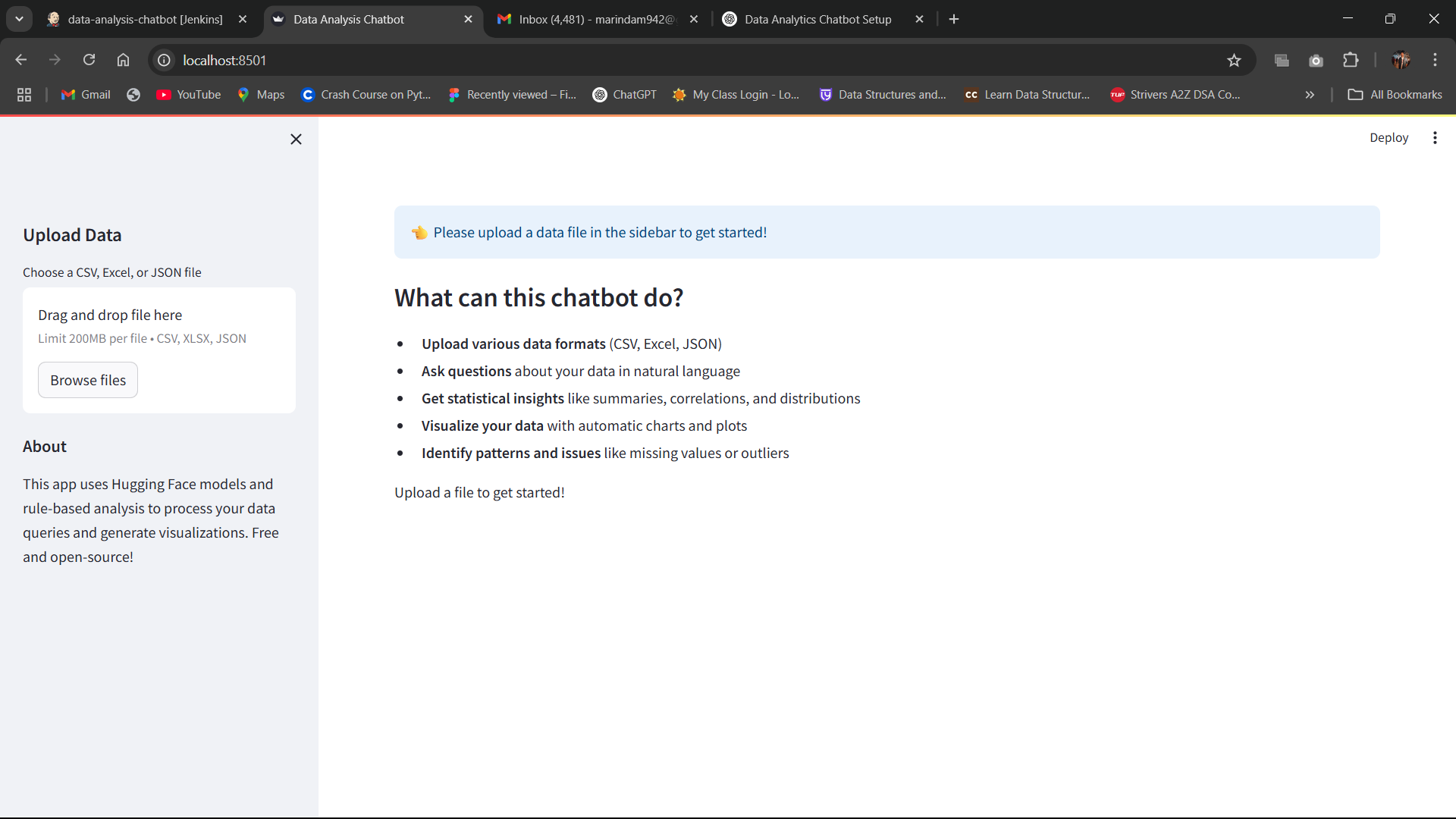
**Test Results:** All the test cases mentioned above passed successfully. No defects encountered.

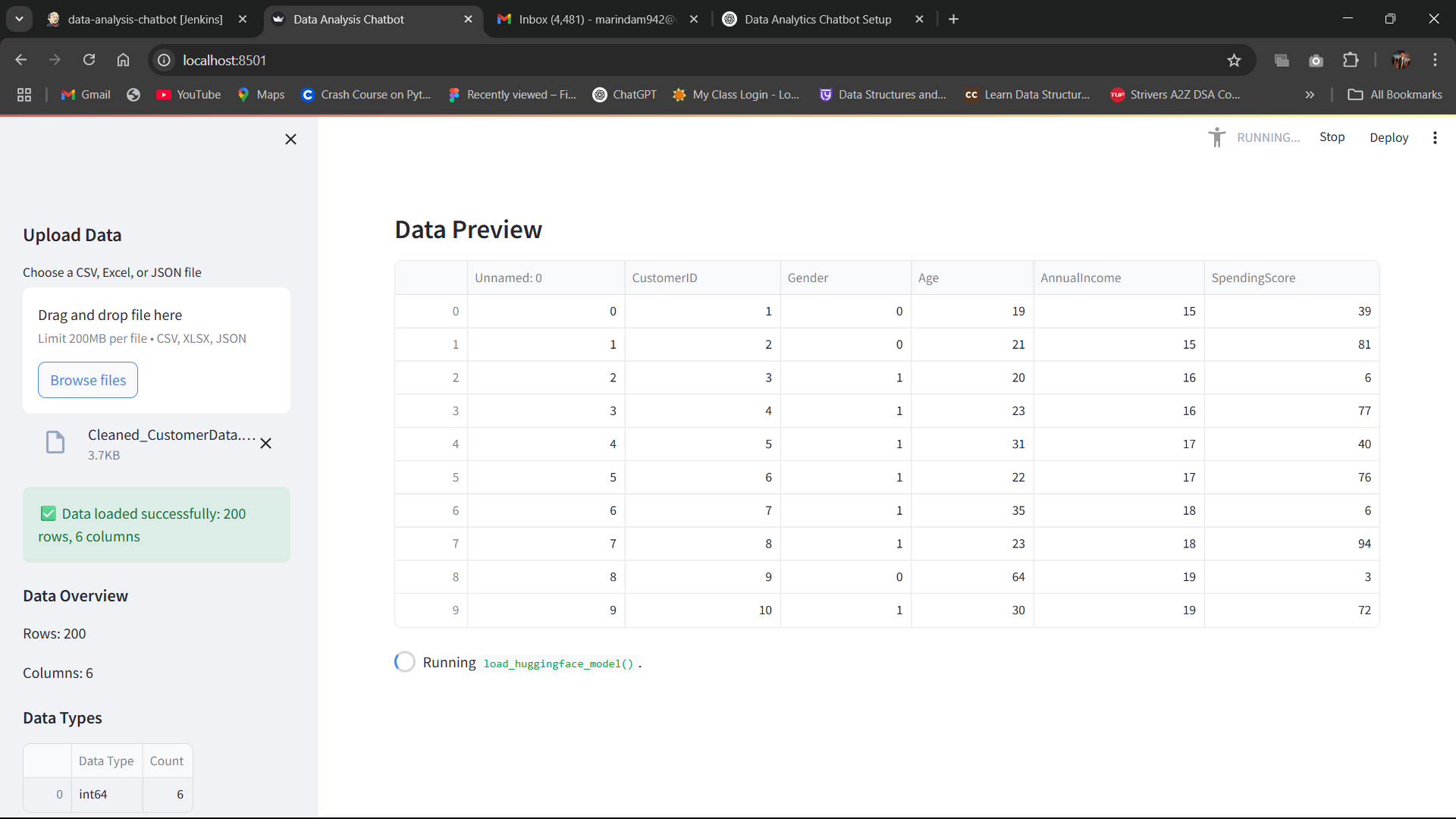
# SCREEN SHORTS

Jenkins Build:



ChatBot Ui:

Csv upload:



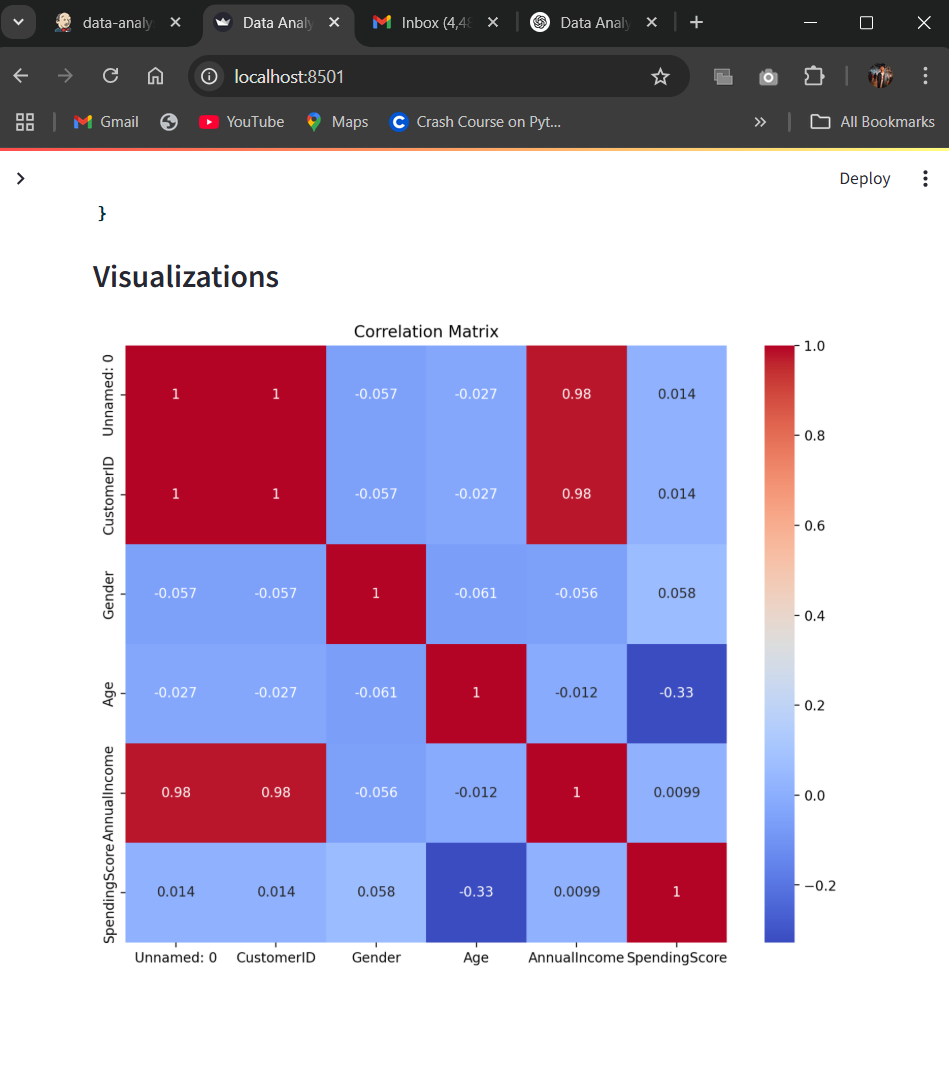
Analyze:

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.



# CONCLUSION

The **Data Analytics Chatbot** enables non-technical users to analyze datasets conversationally. Through NLP and rule-based logic, it interprets queries and delivers actionable insights. Docker and Jenkins integration ensures maintainable deployment.

Future enhancements could include:

* Voice input integration
* Multilingual support
* Support for non-tabular data formats
* Auto-insight generation

This system highlights the synergy between conversational AI and data science tools, paving the way for inclusive and intelligent analytics platforms.

# REFERENCES

1. https://streamlit.io/
2. https://huggingface.co/
3. https://www.jenkins.io/
4. https://www.docker.com/
5. Python documentation (https://docs.python.org)
6. Matplotlib and Pandas documentation
7. Research articles on chatbot and NLP systems