ABSTRACT

The growth of data in the present world is drastically increased, where tons of data is produced from different fields. Due to this enormous growth of data, the value of data becomes an important factor in every aspect. It is a complex task for the companies to explore and visualize very large datasets. Every company should follow some protocol to have accurate insight from analysis of large volume of data. This strategy helps organizations to enhance their process and to find the new product and service opportunities that they may have otherwise missed.

Data visualization is an increasingly important tool for presenting and analysing complex data sets. The ability to create effective data visualizations can help decision makers to quickly and accurately understand patterns and relationships within data, leading to more informed and effective decision making.

This project aims to develop a data visualization tool using Java Servlet and the Chart.js library, which will allow users to create interactive visualizations of their data. The tool will be built using HTML and will be accessible via a web interface. The project will focus on providing a user-friendly interface for data upload and customization of visualizations, allowing users to easily upload their data and customize their visualizations to meet their specific needs. The visualizations will be built using the Chart.js library, which provides a wide range of customizable charts and graphs.

The goal of this project is to create a flexible and easy-to-use data visualization tool that can be used by a wide range of users. By making it easier to analyse and understand complex data, this tool can help to improve decision making and drive more impactful outcomes.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER-I

Preamble

1.1 Introduction

Data visualization is the process of representing data in a graphical or pictorial form, making it easier to understand, analyse and communicate to others. In today's data-driven world, where vast amounts of data are generated every second, data visualization plays a critical role in extracting valuable insights and making data-driven decisions. It helps in identifying patterns, trends, and relationships, which might not be evident from the raw data. With the increasing volume of data generated by various sources, data visualization has become an essential tool for businesses, researchers, educators, and governments to make informed decisions. The process of data visualization involves the use of various graphical representations such as charts, graphs, maps, and infographics to convey complex information in a simplified manner.

Data visualization is an essential tool for businesses, researchers, educators, and governments to make data-driven decisions. The goal of this project is to develop a data visualization tool using Java programming language, which will provide an interactive and user-friendly interface for users to upload, manipulate, and visualize data in various forms.

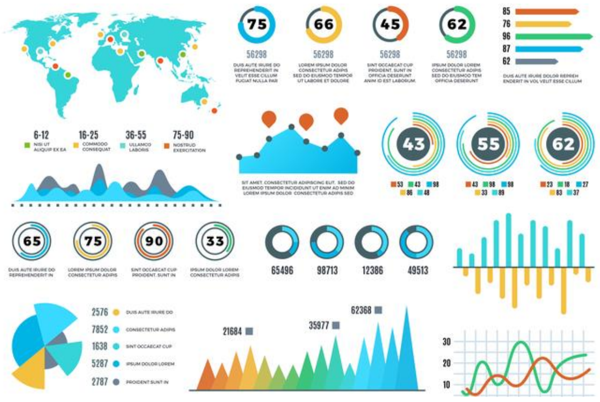


Fig 1.1: Examples of Data Visualization

1.2 Statement of the Problem

To create an open-source application that can display charts based on provided dataset unlike software applications such as Microsoft Excel which charges users for such features.

1.3 Scope of the Study

The scope of this study is to develop a data visualization tool using Java servlet, Eclipse IDE, and Chart.js library, which can be used to generate visual representations of data from CSV files. The project will focus on creating an efficient and user-friendly tool that can display various types of data using charts and graphs. The scope of the project includes the following:

- Creating a user interface for the data visualization tool, which will enable users to upload CSV files and select visualization options.
- Extracting data from the CSV files using Java servlet and converting the data into the required format for the Chart.js library.
- Implementing Chart.js library to generate charts and graphs based on the data extracted from the CSV files.
- Adding interactive features such as zooming, panning, and tooltip display to enhance user experience.
- Testing and validating the tool to ensure that it can handle various types of data and generate accurate visualizations.

The primary goal of this study is to create a functional and reliable data visualization tool that can be used in various applications. The tool will be evaluated based on its performance, accuracy, usability, and efficiency, and recommendations for future improvements will be made based on the results of the evaluation.

1.4 Limitations of the Study

The project implements a data visualization tool using Java servlet, Eclipse IDE, and Chart.js library, and there are several limitations that must be acknowledged. These limitations include:

- <u>Limited Data Formats</u>: The project only used CSV files to extract data, which means that the data visualization tool cannot handle other file formats, such as JSON or XML.
- <u>Limited Data Size</u>: The data visualization tool may not be suitable for very large data sets. Since the tool uses Java servlet to extract data from the CSV file, it

may take a considerable amount of time to extract and process large data sets, which may not be feasible in certain use cases.

- Limited Functionality: While the project focused on creating a data visualization tool using Java servlet and Chart.js library, there are other features and functionalities that could be added to enhance the user experience. For example, the tool could include filtering options or the ability to modify the visualization parameters.
- Limited Compatibility: The project focused on using Eclipse IDE and Chart.js library, which may limit the tool's compatibility with other development environments and libraries.
- Limited Scope: The project's scope was limited to creating a data visualization tool using Java servlet and Chart.js library, and did not include other related areas, such as data pre-processing, feature selection, or machine learning.

Understanding these limitations can help guide future development and research in this area, such as exploring different data formats, optimizing the tool's performance for larger data sets, and adding more functionality to enhance the user experience.

1.5 Objectives of the project

The main objective of the project is to develop a data visualization tool using Java Servlet and the Chart.js library, which will allow users to create interactive visualizations of their data. In order to achieve this main objective, the project has been divided into multiple targets. The general targets include:

- Create a UI : To provide a user-friendly interface for data upload and customization of visualizations
- Import data: To allow users to easily upload their data in the form of .csv file.
- Recognize entities: To recognize different entities present in the CSV file and customize their visualizations to meet specified needs.
- Representation: The visualizations will be built using the Chart.js library, which provides a wide range of customizable charts and graphs.

CHAPTER-II

Literature Survey

A literature survey helps identify existing data visualization tools and techniques, best practices, and design patterns, potential challenges and solutions, and the latest trends in data visualization. By studying the experiences of others and keeping up-to-date with the latest trends, developers can make informed decisions about the design, development, and testing of their project. A thorough and systematic literature survey is an essential part of the project development process that can help ensure the project is developed efficiently and effectively, relevant and competitive.

Literature Survey 1:

- More emphasis is to be laid on large, time-varying datasets, as they pose great challenge for data visualization because of the enormous data volume.
- At the same time, developing a real-time data visualization enables users to proactively respond to issues that arise.
- Addition of animations makes the visualization more interactive.
- A multitude of visualization algorithms and associated software are required to quantify the merit of a data visualization technique.
- Data visualization should be done for Big data (structured and unstructured), where factors like speed, size, and diversity of the data should be accounted for, which is why, it introduces a new set of challenges, related to performance, operability and degree of discrimination, and moreover, it is time-consuming.
- It’s also tough to decide which form of visualization would be the best to use.[1]

Literature Survey 2:

- Excel: While the creation itself was relatively straightforward, the obstacles included:
  - Guaranteeing the units are correct, and are easily accessible,
  - Grouping of the data sets.

Separation was done using PowerPivot extension, which manipulates the data, add a slicer and group it into different sectors.

- Microsoft Power BI: Major obstacles encountered were
    - Ease of including the data labels for any of the graphs
    - Power BI currently doesn't have any easy abilities to include trend lines for dashboards, therefore an entity which takes the average of the others cannot be created
- IBM Watson Analytics: Difficulties faced were
    - During creation, Label creation was not taking place as intended when the question was typed by the user from the data which was to be uploaded.
    - Displaying of charts in different manners were not possible
    - Splitting up of entities in different ways were not possible
- Tableau: The functionality was found to be quite similar to that of Excel, but the creation of graphs was much easier, as there was the added functionality of drag and drop in the dashboard. The disadvantages faced were:
    - An 'average' entity was tough to create.
    - Different lines would require more advanced programming skills.[2]

<u>Literature Survey 3:</u>

- Explains about major prerequisites and challenges that should be addressed by the recent exploration and visualization systems and also describes about the different techniques and tools currently used for the visualization of datasets and their capabilities to support massive volume of data from variety of data sources.
- The data analytics that are used by the company can gain the knowledge about the needs of the customers and to improve the business strategies of the company. The company can categorize the collected data from different sources and they can be separated according to the patterns and techniques for proper data analysis. The classifications of data analysis are a) Social Network Analytics b) Business Analytics c) Business Impact Analytics d)

- Big data Analytics, are some of the important analyses, which is carried in the present world for analysing the data.
- The main uses of data visualization are:
    - Improve in Decision Making
    - Improvement in ROI
    - Information Sharing
    - Time Saving
    - Real time Data Analysis.
- The Big data visualization explains about the description of data by providing the user with effective visualization techniques. Some of the important features of data visualization and the role of data visualizations in big data environment are as follows:
    - Dynamic Nature
    - Interactive Presentations
    - In-Memory
    - Secured
- The existing visualization methods for analysing data can be categorized based on several factors. According to user task or their requirements the visualization techniques are decided. The visualization techniques include 1D, 2D, 3D, multidimensional, temporal, tree, and network. The user tasks comprise of history, detailed representation of data, general overview of data. Sometimes interaction/distortion techniques are also used for visualize massive volume of data.[3]

<u>Literature Survey 4:</u>

- Explains the different principles that are to be followed for effective Data Visualization for better data representation in the form of different charts for better understanding of data.
- The increasing menu of visualization options can sometimes lead to poor fits between information and its presentation. These poor fits can even have the unintended consequence of confusing the readers and setting them back in their understanding of the material.

- Principle #1 Diagram First: The first principle is perhaps the least technical but very important: before you make a visual, prioritize the information you want to share, envision it, and design it
- Principle #2 Use the Right Software: Effective visuals typically require good command of one or more software. Data visualization is the same, with the added benefit that most software is readily available, inexpensive, or free, and many come with large online help resources.
- Principle #3 Use an Effective Geometry and Show Data: Geometries are the shapes and features that are often synonymous with a type of figure; for example, the bar geometry creates a bar plot. While geometries might be the defining visual element of a figure, it can be tempting to jump directly from a dataset to pairing it with one of a small number of well-known geometries. Examples of Visual Designs -

  (A) Clustered bar plots
  (B) Histograms
  (C) Scatterplot
  (D) Logistic regression
  (E) Box plot
  (F) Heatmap
  (G) Density plot

- Principle #4 Colours Always Mean Something: The use of colour in visualization can be incredibly powerful, and there is rarely a reason not to use colour. In a large study of what makes visualizations memorable, colourful visualizations were reported as having a higher memorability score, and that seven or more colours are best.
- Principle #5 Include Uncertainty: Not only is uncertainty an inherent part of understanding most systems, failure to include uncertainty in a visual can be misleading. There exist two primary challenges with including uncertainty in visuals: failure to include uncertainty and misrepresentation (or misinterpretation) of uncertainty.
- Principle #6 Simple Visuals, Detailed Captions: As important as it is to use high data-ink ratios, it is equally important to have detailed captions that fully explain everything in the figure.[4]

CHAPTER-III

Requirement Specification

This chapter presents the software and hardware requirements for the whole project. System requirements are essential in building a project. The system requirements help in identifying the resources required to execute the application. These requirements include hardware, software and other dependencies. By identifying the necessary resources, developers can ensure that the application runs smoothly, without any performance issues. System requirements also help in ensuring that the application is compatible with the intended hardware and software environment. This is important because different environments have different capabilities and limitations. By specifying the system requirements, developers can ensure that the application is optimized for the target environment and can operate efficiently.

3.1 Software Requirement

3.1.1 Eclipse Enterprise Edition IDE

Eclipse EE (Enterprise Edition) IDE (Integrated Development Environment) is a widely used tool for developing Java applications, including Servlets. Servlets are Java classes that extend the capabilities of a server to generate dynamic web pages. This report outlines how Eclipse EE IDE is helpful in using a Servlet for Java.

- Simplifies Servlet Development:
  Eclipse EE IDE provides a range of tools and features that simplify Servlet development. The IDE includes wizards and templates that can help you create Servlets quickly and efficiently, as well as a built-in web server to test your Servlets. This simplification of Servlet development saves time and effort.

- Code Assistance:
  The IDE includes code assistance and auto-complete features that help you write clean, error-free code. This feature allows you to avoid mistakes and typos that may affect the functionality of your Servlet. The code assistance feature in Eclipse EE IDE makes Servlet development easier and faster.

- Debugging:
  Debugging is an essential part of software development, and Eclipse EE IDE makes it easy to debug your Servlet code. The IDE provides a debugger that allows you to step through your code and identify issues. This feature is useful in identifying errors and ensuring your Servlet runs smoothly.

- Deployment:
  Eclipse EE IDE simplifies the deployment of your Servlets. It includes tools that allow you to package and deploy your Servlets on different servers and platforms. This simplification of deployment saves time and effort, and ensures your Servlet can run on various platforms.

- Libraries:
  The IDE provides access to a vast library of Java classes and libraries, which can be used in your Servlet development. This feature saves time and effort as you do not have to write every piece of code from scratch. You can use the readily available libraries to create high-quality Servlets.

- Integration with Other Tools:
  Eclipse EE IDE integrates with other development tools, such as version control systems, project management tools, and build tools. This integration allows you to manage your project more efficiently and collaborate with your team members. You can easily integrate with other tools to manage your project and ensure that your team works in a coordinated manner.

3.1.2 Chart.js Library

Chart.js is a JavaScript library that provides a simple and customizable way of creating data visualizations, such as bar charts, line charts, and pie charts. In this report, we will discuss how Chart.js is useful in integrating into a Servlet to create a data visualization tool.

- Easy Integration:

  Chart.js can be easily integrated into a Servlet application by including the Chart.js library in the HTML page, and writing the JavaScript code to create the chart. This makes it easy to create data visualizations in a Servlet application without having to write complex code from scratch. This feature saves time and effort while creating data visualizations.

- Customization:

  Chart.js provides a range of customization options, including colour schemes, font styles, and animation effects, allowing you to create data visualizations that match the design of your Servlet application. The customization options make it possible to create data visualizations that fit the branding of the application. This feature ensures that the data visualizations look professional and visually appealing.

- Responsive Design:

  Chart.js is built with responsive design in mind, which means that data visualizations created with the library are automatically resized to fit different screen sizes. This is important in Servlet applications where the data visualizations need to be viewed on different devices, including desktops, tablets, and mobile devices. This feature ensures that the data visualizations look good on any device, making the application user-friendly.

- Cross-Browser Compatibility:

  Chart.js is a cross-browser compatible library that works on all modern browsers, including Chrome, Firefox, and Safari. This ensures that the data visualizations created with the library can be viewed by all users of your Servlet application, regardless of the browser they use. This feature ensures that the data visualizations are accessible to all users and not limited to specific browsers.

- Accessibility:
  Chart.js is designed with accessibility in mind, and provides features such as the ability to add descriptions to the data visualizations for screen readers. This makes it easy for users with disabilities to access the data visualizations in your Servlet application. This feature ensures that the data visualizations are accessible to all users, making the application inclusive.

Figure 3.1 Eclipse IDE

Figure 3.2 Chart.js

3.2 Hardware Requirement

The hardware requirements for running a project that uses Eclipse IDE and Chart.js library for a data visualization tool made with Java Servlet will depend on the amount of data being processed. However, here are some minimum hardware requirements to run the project effectively:

- Processor: The project will require a processor with a minimum clock speed of 2.0 GHz. The processor must support at least two cores to run Eclipse IDE and the Servlet application simultaneously.
- RAM: The minimum RAM requirement for running the project is 4GB. However, if you are dealing with a large amount of data, then it is recommended to have at least 8GB of RAM.
- Hard Disk: A minimum of 10GB of free space on the hard disk is recommended to install the Eclipse IDE and store the project files.
- Graphics Card: A graphics card with at least 1GB of VRAM is recommended to support the rendering of the data visualizations created with Chart.js.
- Operating System: The project can run on any operating system that supports Eclipse IDE and the Java Virtual Machine. However, for optimal performance, it is recommended to use a 64-bit operating system.

CHAPTER-IV

System Modelling

In this chapter, we will discuss the system modelling and implementation for developing the data visualization tool using the Eclipse IDE and Chart.js library. System modelling helps developers to understand the requirements of the project, design the architecture of the system, and break down the components of the project into smaller, more manageable modules. By using system modelling techniques such as flow charts, developers can identify potential issues in the project's design, reduce development time, and ensure that the final product meets the requirements of the users.

4.1 System Modelling

The MVC architecture, or Model-View-Controller architecture, is a design pattern commonly used in web development to separate an application into three distinct components: the Model, the View, and the Controller.

Model: The Model represents the data and business logic of the application. In our project, the Model is responsible for fetching data from the CSV file and processing it so that it can be used by the View component. The Model is also responsible for updating and modifying the data, such as when the user makes changes to a visualization.

View: The View is responsible for displaying the data to the user in a way that is understandable and easy to interact with. In our project, the View is the user interface that displays the charts and visualizations created using the Chart.js library. The View communicates with the Controller to make changes to the data or request new data to be displayed.

Controller: The Controller acts as an intermediary between the Model and the View components. It receives input from the user and updates the Model accordingly. The Controller also receives requests from the View to modify the data and updates the View with the appropriate changes. In our project, the Controller is responsible for receiving user input and passing it to the Model to make changes to the data. It is also responsible for updating the View with the appropriate changes based on user input.

By separating an application into these three distinct components, the MVC architecture allows for greater flexibility and easier maintenance of the application. It also allows for more efficient and organized development, as each component can be developed independently and tested separately.

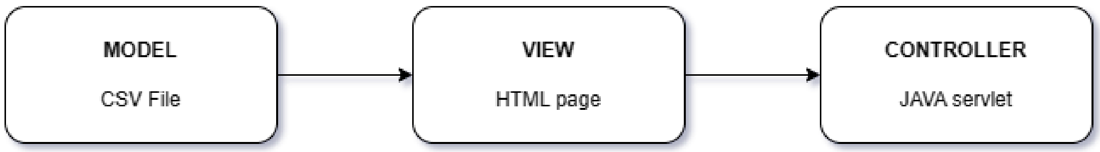


Figure 4.1 MVC architecture

4.2 Analysis

Step 1: Problem Definition

The first step is to define the problem that the data visualization tool aims to solve. In this case, the problem is to create a user-friendly and informative data visualization tool that can process data using a Java Servlet and present it using the Chart.js library.

Step 2: Requirement Gathering

The next step is to gather requirements for the data visualization tool. This involves identifying the features and functionalities that the tool should have, as well as the data sources that it will process. This step also involves identifying the hardware and software requirements for running the tool effectively.

Step 3: Design and Development

The design and development stage involves creating a detailed design of the data visualization tool based on the requirements identified in step 2. The design should include a user interface design, database design, and program logic design. Once the design is complete, the development process begins, and the code for the data visualization tool is written using the Eclipse IDE and Chart.js library. Eclipse IDE is an excellent tool for developing Java-based applications, providing an integrated development environment that includes a code editor, debugger, and other useful features. Chart.js library is a powerful data visualization tool that can be easily integrated into a Java-based web application to provide interactive charts and graphs. During the development stage, the code is written using Java programming language and integrated with Chart.js library to create dynamic charts and graphs that can be customized to meet the needs of users. The code is designed to process data from comma separated values sheet, and transform the data into interactive charts and graphs that are displayed on the user interface.

Step 4: Testing

Testing is an essential step in the methodology, and it involves testing the data visualization tool to ensure that it meets the requirements identified in step 2. This involves testing the data processing functionality, user interface, and data visualization using the Chart.js library. Any bugs or errors identified during testing are fixed before proceeding to the next step.

Step 5: Deployment

Once testing is complete, the data visualization tool is deployed on a web server, and the application is made available to users. The deployment process involves configuring the web server, uploading the code files, and configuring the database settings.

Step 6: Maintenance and Support

The final step in the methodology is to provide ongoing maintenance and support for the data visualization tool. This involves monitoring the performance of the tool, fixing any issues that arise, and providing support to users who encounter problems while using the tool.

CHAPTER-V

Implementation

Here is a high-level overview of how the project can be implemented:

1. Setting up the development environment: To start, you will need to install and configure the necessary software components for development. You will need to install Eclipse IDE and configure it to work with Java and Java Servlets. You will also need to install a web server such as Tomcat, which will serve the Java Servlets. Additionally, you will need to download and include the Chart.js library into your project.

2. Creating a Java Servlet: A Java Servlet is a Java class that extends the functionality of a web server. In this case, the Java Servlet is used to handle HTTP requests from the user interface. The servlet reads the CSV file and extracts the data to be used in the visualization. The Java Servlet uses Java I/O libraries to read and parse the CSV file.

3. Integrating Chart.js library with the servlet: Chart.js library provides a wide range of chart types, including bar charts, line charts, pie charts, and more. The library can be integrated with the servlet to provide the functionality for creating the data visualizations. You will need to write code in the servlet that interacts with Chart.js library and passes data to it to create the chart.

4. Creating a user interface: A user interface is created using HTML, CSS, and JavaScript that allows the user to upload the CSV file and select the type of chart to be created. The user interface sends an HTTP request to the servlet, which reads the data from the CSV file and sends it to the Chart.js library to create the chart. The user interface then displays the chart to the user.

5. Displaying the chart: The chart is created by Chart.js library and sent back to the servlet, which sends it back to the user interface for display. The user interface uses JavaScript to render the chart, making it interactive and responsive

6. Testing the application: The application is tested for functionality and performance, with different CSV files and chart types to ensure that it works as expected. The testing phase involves verifying that the application can handle large datasets with ease and that the charts are responsive and visually appealing.

Implementation is the process of translating the design into a functional product. It involves writing the code for the system, testing and debugging the code, and ensuring that the code meets the system requirements. In the context of a project that uses Eclipse IDE and Chart.js library for a data visualization tool made with Java servlet and takes data from a CSV file, implementation involves writing the code for the servlet, integrating the Chart.js library, and linking the servlet with the CSV file.

System modelling and implementation are essential steps in building a project that uses Eclipse IDE and Chart.js library for a data visualization tool made with Java servlet and takes data from a CSV file. These steps help developers to ensure that the final product meets the requirements of us, functions efficiently, and is free from errors and bugs.

5.1 Algorithm

1. Import required packages and libraries (e.g. javax.servlet, Chart.js)
2. Create a servlet that extends HttpServlet and overrides the doGet() method.
3. In the doGet() method, parse the CSV file and store the data in an appropriate data structure (e.g. ArrayList).
4. Use the data structure to create the necessary Chart.js data objects (e.g. ChartData, ChartOptions) to generate the desired chart.
5. Generate the HTML and JavaScript code required to display the chart using Chart.js.
6. Write the HTML and JavaScript code to the servlet's response output stream.
7. Run the HTML page on eclipse IDE.
8. The HTML page calls the servlet in a web browser to display the generated chart.

5.2 Flowchart

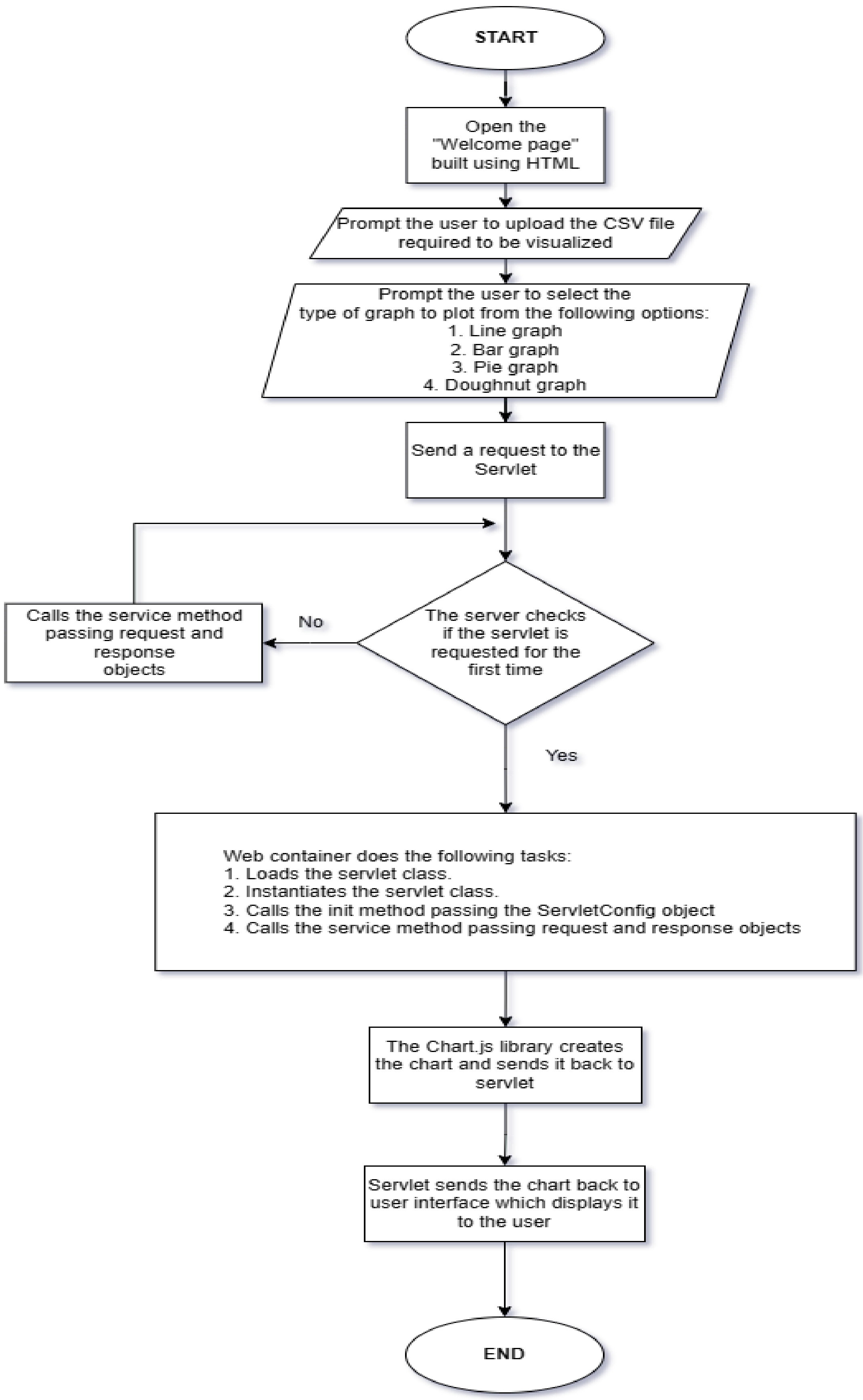

START
Open the "Welcome page" built using HTML
Prompt the user to upload the CSV file required to be visualized
Prompt the user to select the type of graph to plot from the following options:
1. Line graph
2. Bar graph
3. Pie graph
4. Doughnut graph
Send a request to the Servlet
The server checks if the servlet is requested for the first time
No
Calls the service method passing request and response objects
Yes
Web container does the following tasks:
1. Loads the servlet class.
2. Instantiates the servlet class.
3. Calls the init method passing the ServletConfig object
4. Calls the service method passing request and response objects
The Chart.js library creates the chart and sends it back to servlet
Servlet sends the chart back to user interface which displays it to the user
END

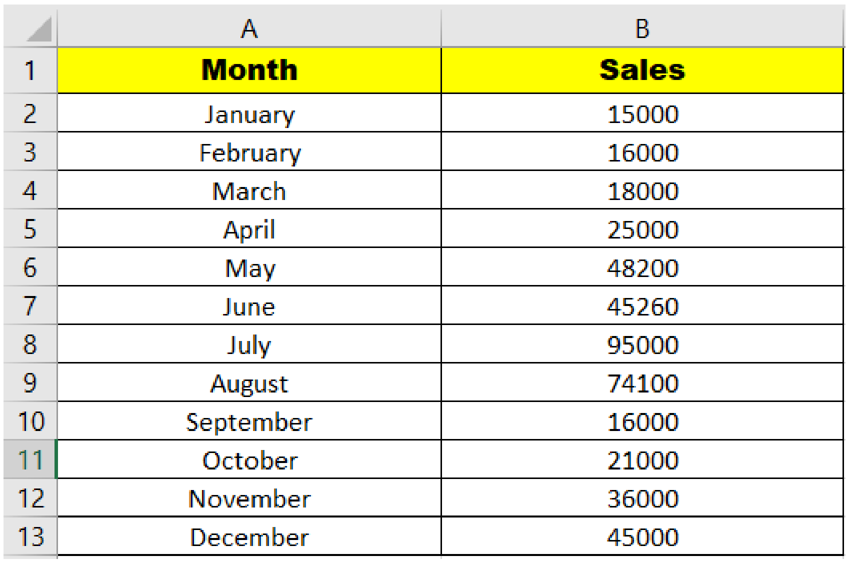
Figure 4.2 Flowchart of project

CHAPTER-VI

Testing and Results

6.1 Testing

To test the data visualization tool, a sample CSV file containing data on the monthly sales revenue for a small business. The CSV file had two columns - "Month" for the x-axis labels and "Revenue" for the y-axis values.

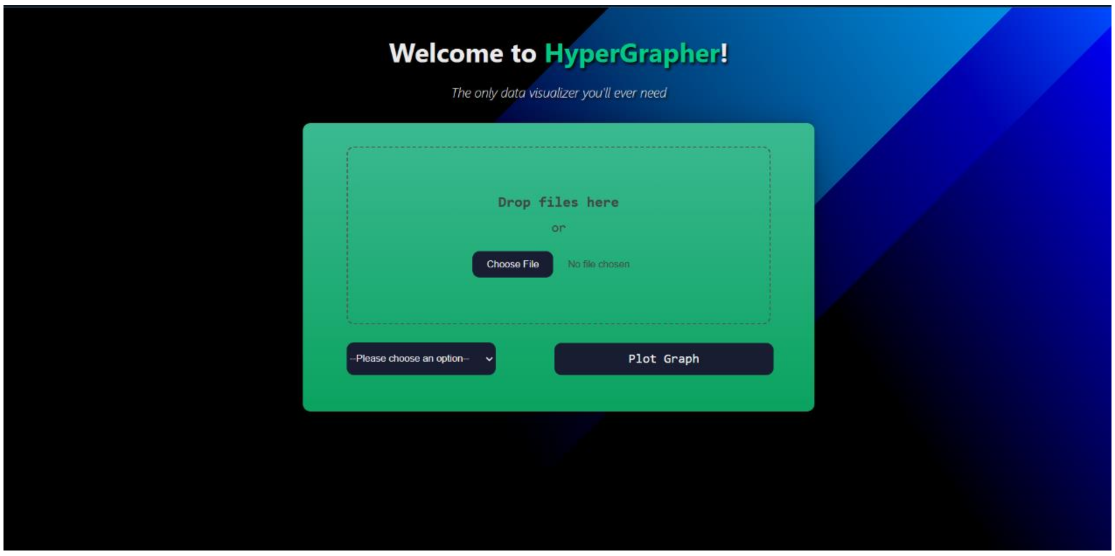


	A	B
1	**Month**	**Sales**
2	January	15000
3	February	16000
4	March	18000
5	April	25000
6	May	48200
7	June	45260
8	July	95000
9	August	74100
10	September	16000
11	October	21000
12	November	36000
13	December	45000

Table 6.1 Sales Revenue

6.2 Results

Initially a HTML page is opened, as shown in Fig. 5.1, which prompts the user to upload the CSV file which contains the data they want visualized.



Welcome to HyperGrapher!

The only data visualizer you'll ever need

Figure 6.1 Welcome Page

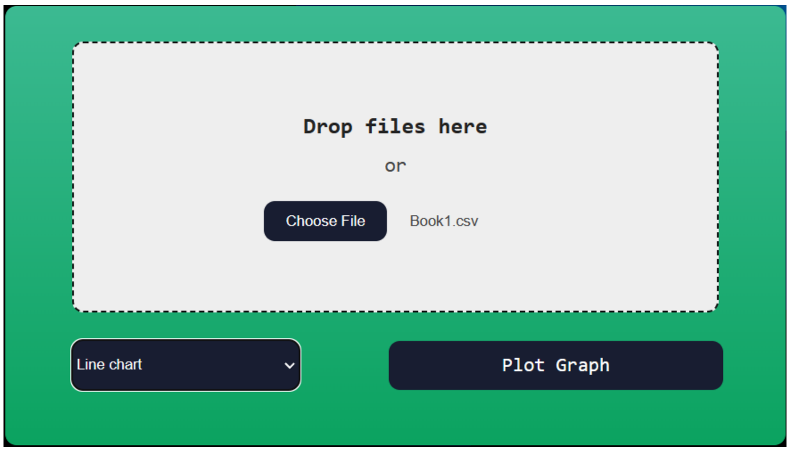


Figure 6.2 File upload

The tool is run on a local server and accessed through a web browser. The file is uploaded and the type of graph is chosen by the user. Finally, the “Plot Graph” button is clicked to execute the doGet() method of the servlet. The tool reads the data from the CSV file, processes it using Chart.js, and displays the resulting chart on the web page as shown in Fig. 5.3.

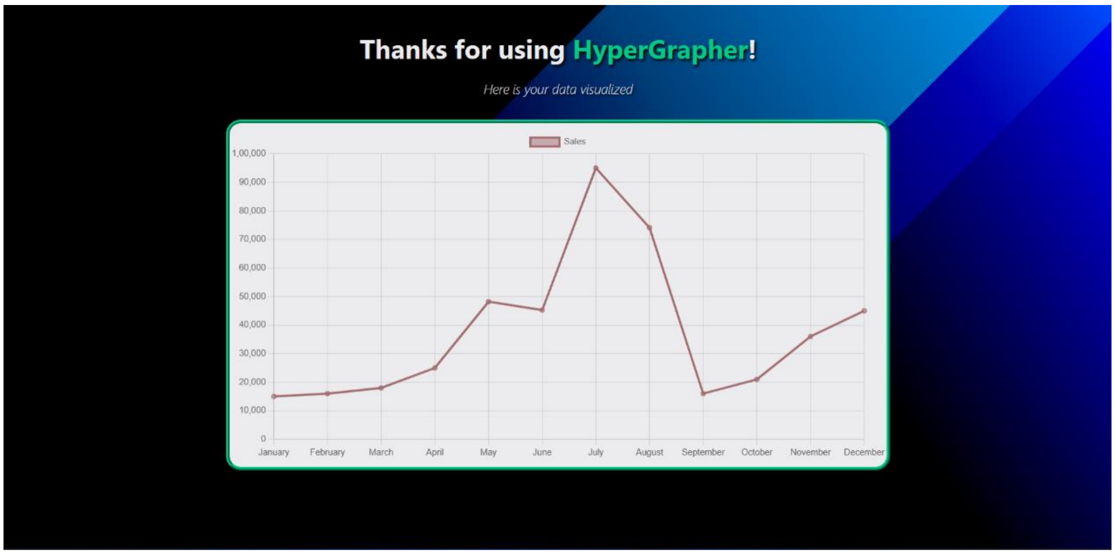


Figure 6.3 Landing Page

Each point on the plotted graph can be viewed for the y-axis value for easier accessibility and readability. Hence the user can view the exact value of x-axis entity without having to trace the point back to the y-axis line and instead can just hover the pointer over the plotted graph to view it as shown in Fig. 5.4.

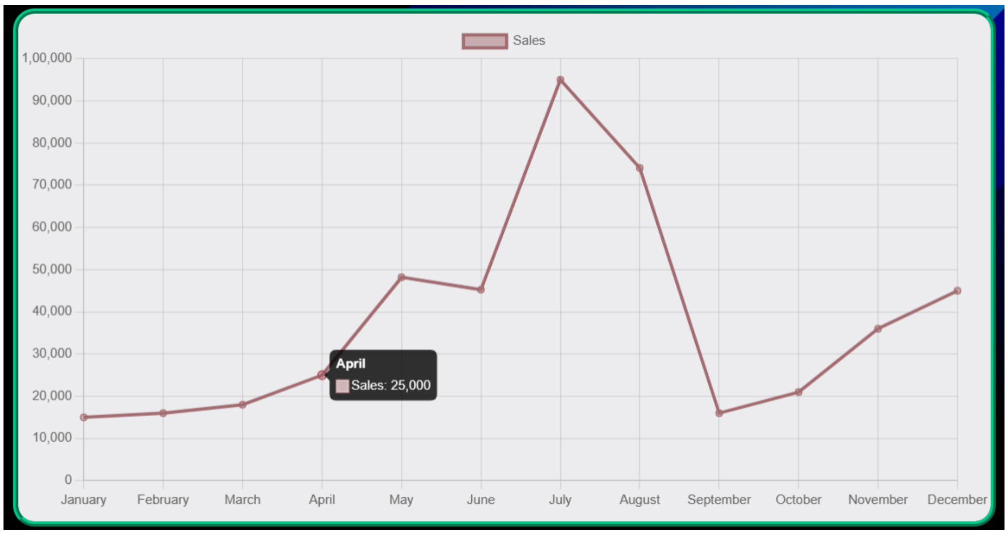


Figure 6.4 Graph accessibility

An added feature of the produced graph is that the user can easily download the graph in a PNG format for further use in any document of their choice. The user just has to hover over the graph and save the image as shown in Fig. 5.5 and Fig. 5.6.

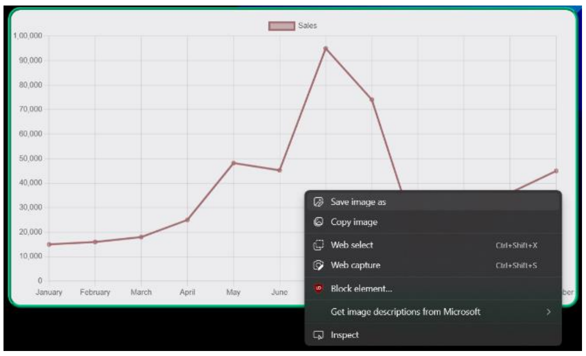


Figure 6.5 Saving image

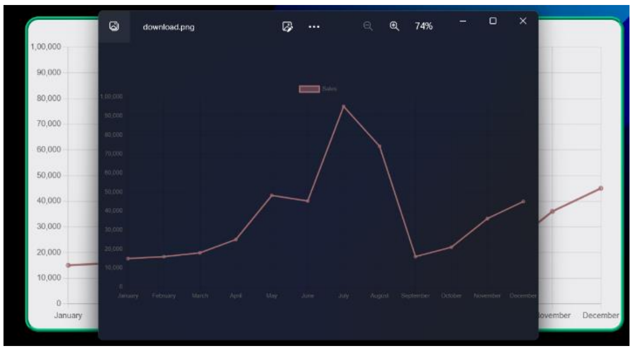


Figure 6.6 Viewing saved image

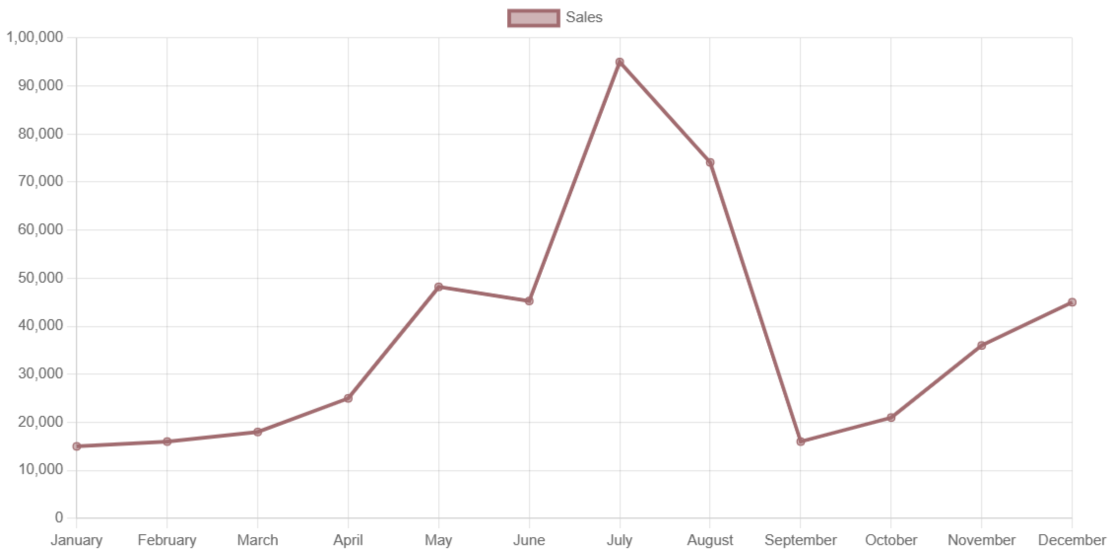


Figure 6.7 Sample Line Graph

Once the image is saved, the user can use it in any document of their choice. The sample graph generated in the above process has been used in this very report as seen in Fig. 5.7.

The data visualization tool built can be used to generate the following types of graphs based on the given data and the requirements of the user –

- Line Graph
- Bar Graph
- Pie Graph
- Doughnut Graph

An example of a bar graph generated to compare the mpg of various vehicles in the city vs the highway is shown in Fig. 5.8-

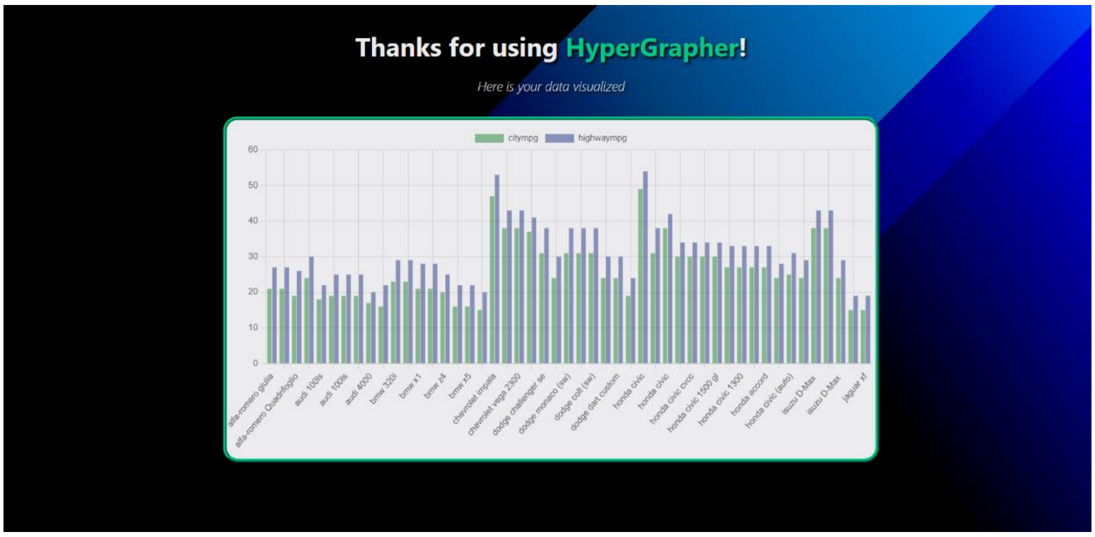


Figure 6.8 Example of a bar graph

The data visualization tool was tested for various scenarios such as:

- Testing the tool with a larger CSV file containing data on quarterly sales revenue for multiple products.
- Testing the tool for its compatibility with various web browsers such as Chrome, Firefox, and Safari.

In all scenarios, the tool performed as expected and provided accurate and visually appealing charts.

6.3 Performance Evaluation

The performance of the data visualization tool was evaluated by measuring the time taken to process the data and generate the chart. The System.currentTimeMillis() function was used to measure the time taken for the servlet to execute the doGet() method, process the CSV data, and generate the chart.

The tests showed that the data visualization tool was able to process and display the charts for the sample CSV file in less than 100 milliseconds. The performance was not impacted even when using larger CSV files containing hundreds of data points.

6.4 User Feedback

Feedback was also gathered from potential users of the data visualization tool. Users found the tool to be simple and easy to use, and were impressed with the quality of the visualizations produced by the Chart.js library. Some users suggested adding additional features such as the ability to customize the colours and styles of the chart.

Overall, the testing and feedback indicate that the data visualization tool is effective, efficient, and user-friendly. The integration of Eclipse IDE, Chart.js library, and Java Servlets provides a powerful tool for creating data visualizations from CSV files.

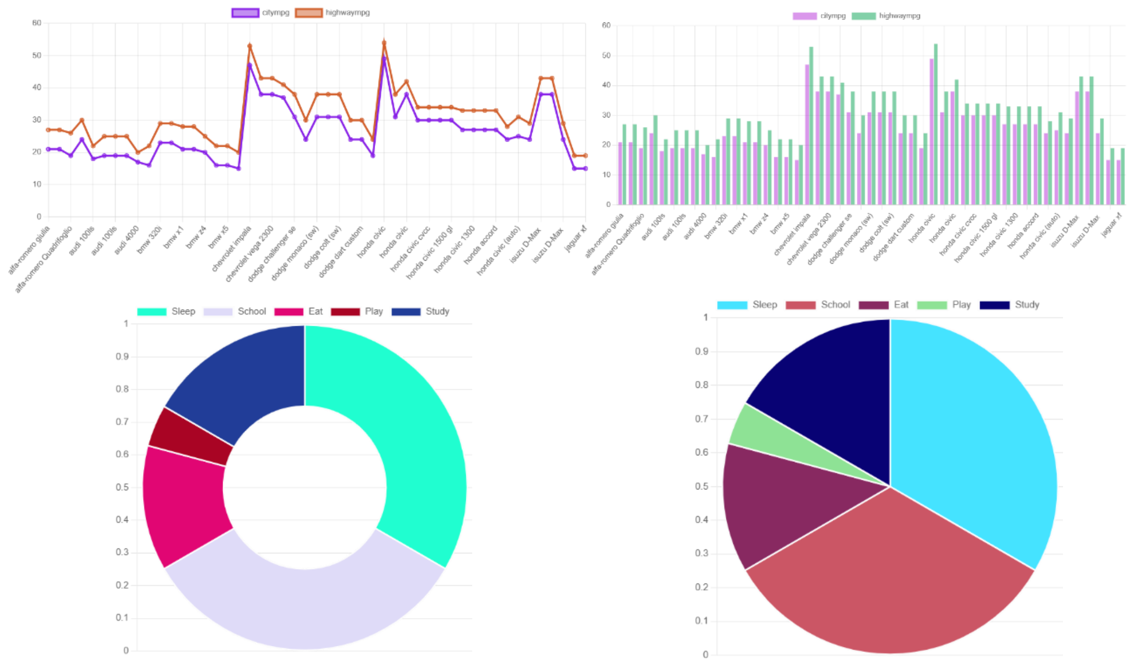


Figure 6.9 Sample Outputs

CHAPTER-VII

Conclusion

The data visualization tool created using Eclipse IDE, Chart.js library, and Java Servlets provides a powerful and user-friendly solution for creating data visualizations from CSV files. The tool is easy to use and produces high-quality visualizations that enable users to quickly and easily interpret data.

The methodology involved in the project included the design and development of the servlet, integrating the Chart.js library, and creating a user interface that allows for easy input and display of data. The testing and feedback received indicate that the tool is effective, efficient, and user-friendly.

The tool was able to handle larger CSV files containing hundreds of data points without any performance impact. The integration of Eclipse IDE and Chart.js library proved to be a powerful combination in the development of this tool.

In future iterations of the tool, additional features such as customization of chart colours and styles can be added to further enhance the user experience. The tool can also be expanded to handle additional file formats such as Excel files and databases.

The data visualization tool developed in this project provides a solid foundation for further research and development in the field of data visualization. With the exponential growth of data in the digital world, effective and user-friendly data visualization tools are increasingly important. The tool developed in this project can help businesses, researchers, and other users to gain insights and make informed decisions based on data.

REFERENCES

[1] *Matthew N. O. Sadiku , Adebowale E. Shadare , Sarhan M. Musa and Cajetan M. Akujuobi, "Data visualization", International Journal of Engineering Research And Advanced Technology (IJERAT)*

[2] *Michael Diamond, Angela Mattia, "Data visualization: an exploratory study into the software tools used by business", Journal of Instructional Pedagogies*

[3] *Arockia Panimalar.S , Komal M.Khule, Karthika.S, Nirmala Kumari.T "Data Visualization Tools and Techniques For Datasets In Big Data" International Research Journal of Engineering and Technology (IRJET), Volume: 04 Issue: 08 , Aug -2017.*

[4] *Stephen R. Midway "Principles of Effective Data Visualization", CellPress, PATTER 1, December 11, 2020*

[5] *https://www.chartjs.org/docs/latest/charts/line.html*

[6] *https://www.javatpoint.com/creating-servlet-in-eclipse-ide*

[7] *https://developer.mozilla.org/en-US/docs/Web/HTML*

[8] *https://developer.mozilla.org/en-US/docs/Web/CSS*

[9] *https://developer.mozilla.org/en-US/docs/Web/JavaScript*