NEW HORIZON
COLLEGE OF ENGINEERING
Autonomous College Permanently Affiliated to VTU, Approved by AICTE & UGC
Accredited by NAAC with 'A' Grade, Accredited by NBA

A MINI PROJECT REPORT

on

Airline Price Chart

Submitted by
MAHESH DODMANI
NH25CSD810-T

Under the guidance of
Ms.Divyanshi Chhabra,
Assistant Professor

In partial fulfillment for the award of the degree of

BACHELOR OF ENGINEERING

in

COMPUTER SCIENCE AND ENGINEERING

Academic Year: 2025-26 (ODD SEM)

NEW HORIZON
COLLEGE OF ENGINEERING
Autonomous College Permanently Affiliated to VTU, Approved by AICTE & UGC
Accredited by NAAC with 'A' Grade, Accredited by NBA

CERTIFICATE

This is to certify that the mini project work titled "**Airline price char**" is a bonafide work carried out by **MAHESH DODMANI (NH25CSD813-T)** in partial fulfillment of the degree of **Bachelor of Engineering** in **Computer Science and Engineering** of the New Horizon College of Engineering during the year **2025-2026.**

Signature of Guide

Signature of HOD

SEMESTER END EXAMINATION

Name of the Examiner *Signature with date*

1. _____ _____

2. _____ _____

ABSTRACT

This project focuses on the development of a fully interactive airline price visualization tool built using HTML, CSS, and Chart.js. The primary objective is to provide users with a clear, intuitive way to observe and analyze fluctuations in airline ticket prices over a selected period of time. Airline pricing is often influenced by a complex combination of market demand, seasonality, fuel costs, competition, and booking patterns. Because of this dynamic environment, consumers, analysts, and researchers frequently benefit from tools that transform raw price data into meaningful visual insights. The system designed in this project addresses that need by enabling real-time interaction with data through a browser-based interface that requires no installation or specialized software.

The application includes several key features that enhance usability and analytical value. Users can select from multiple airlines, adjust the displayed time range, and instantly view updated visualizations. The interface employs a responsive layout supported by Tailwind CSS, ensuring compatibility across desktops, tablets, and mobile devices. The interactive line chart, powered by Chart.js, highlights daily price variations and provides hover-based tooltips for precise value inspection. Additionally, users can download the chart as a PNG image, allowing for easy inclusion in reports or presentations, and export the underlying dataset as a CSV file for further statistical analysis.

To facilitate testing and demonstration, the project includes a built-in random data generator that simulates realistic airline price movements based on a random-walk model. This design choice allows the visualization to operate independently of external data sources, while still showcasing how real pricing information would appear within the system. However, the tool can easily be adapted to integrate live or historical airfare data from APIs or databases, making it suitable for real-world applications such as forecasting, market comparison, travel planning tools, or academic research in transportation economics.

Overall, this project demonstrates how modern web technologies can be combined to create a powerful yet user-friendly platform for aviation market analysis. By transforming complex numerical data into an accessible visual format, the system supports informed decision-making and enhances understanding of airline price behavior. Its modular design and extensibility make it a practical foundation for future enhancements or integration into larger analytical systems.

ACKNOWLEDGEMENT

The satisfaction and euphoria that accompany the successful completion of any task would be impossible without the mention of the people who made it possible, who's constant guidance and encouragement crowned our efforts with success.

I have great pleasure in expressing gratitude to **Dr. Mohan Manghnani**, Chairman, New Horizon Educational Institutions, for providing necessary infrastructure and creating good environment.

I take this opportunity to express my profound gratitude to **Dr. Manjunatha,** Principal, New Horizon College of Engineering, for the constant support and encouragement.

I would like to thank **Dr. Anandhi R J**, Professor and Dean-Academics, NHCE, for her valuable guidance.

I would also like to thank **Dr. B. Rajalakshmi,** Professor and HOD, Department of Computer Science and Engineering, for the constant support.

I also express my gratitude to **Ms.Divyanshi Chhabra, Assistant Professor**, Department of Computer Science and Engineering, my mini project reviewer, for constantly monitoring the development of the project and setting up precise deadlines. Her valuable suggestions were the motivating factors in completing the work.

MAHESH DODMANI
NH25CSD810-T

CONTENTS

CHAPTER 1 INTRODUCTION

Airline ticket pricing is a dynamic and often unpredictable process influenced by a wide range of factors, including fuel costs, seasonal travel trends, competitive market behavior, operational constraints, and consumer demand. As a result, airfare prices can fluctuate significantly within short periods of time, making it challenging for researchers, analysts, and travelers to identify meaningful patterns or make informed decisions. Effective visualization tools play a crucial role in understanding these fluctuations by transforming raw data into intuitive graphical representations that reveal trends, anomalies, and correlations.

With the increasing availability of web technologies, interactive data visualization has become more accessible and powerful. Modern libraries such as Chart.js allow developers to create responsive, browser-based graphics that offer smooth user interaction and real-time updates. This project leverages these tools to design and implement an interactive Airline Price Chart that enables users to explore historical airfare trends for multiple airlines. By providing an intuitive interface combined with customizable controls, the system supports detailed analysis without requiring specialized software or technical expertise.

The main purpose of this project is to develop a user-friendly visualization system capable of displaying airline price data over selectable time periods. Users can choose a specific airline, modify the date range, and immediately observe corresponding changes in the chart. The integration of download and export features further enhances the practicality of the tool, allowing visual outputs and raw data to be used for reporting, academic studies, or market comparisons. Additionally, the inclusion of simulated price data through a random-walk generator demonstrates the system's functionality independent of external data sources, while also providing a realistic dataset for testing and demonstration purposes.

By combining HTML, CSS, JavaScript, and Chart.js within a cohesive design, the project highlights the potential of web-based visualization to support analytical decision-making in the aviation industry. The system not only helps users gain insights into price fluctuations but also serves as a foundation for future enhancements, such as integrating real-time flight APIs, predictive modeling, or automated alerts based on price thresholds. Ultimately, this project underscores the value of interactive visualization as an essential tool for understanding complex financial and operational

trends in modern air travel.

1.1 PROBLEM DEFINITION

Airline ticket prices exhibit significant variability due to dynamic market conditions, operational considerations, and fluctuating consumer demand. Travelers and analysts often struggle to interpret this variability because price changes occur rapidly and are influenced by numerous interconnected factors. Traditional methods of viewing fare data—such as static tables or manual price checks—do not provide sufficient clarity or visual insight into underlying trends. This creates difficulties in identifying pricing patterns, detecting anomalies, comparing airlines, or making data-driven decisions.

Furthermore, the absence of an accessible, interactive visualization tool limits the ability of users to analyze historical price movements effectively. While raw airfare data may be available through APIs or travel platforms, users often lack a simple method to convert this data into meaningful visual representations.

Without an intuitive interface, the process of analyzing pricing behavior becomes time-consuming, error-prone, and inaccessible to individuals without technical expertise.

The core problem addressed in this project is the need for a web-based system that can visually present airline ticket price variations over time in a clear, interactive, and user-friendly format. Such a system must:

- Allow users to view and compare price trends across different airlines.
- Enable flexible selection of date ranges for focused analysis.
- Provide real-time updates to the visualization based on user input.
- Offer export options for research, reporting, and further analysis.

1.2 OBJECTIVES

The primary objective of this project is to develop an interactive, web-based visualization tool that enables users to analyze and interpret airline ticket price trends effectively. To achieve this, the system aims to convert complex pricing data into an intuitive graphical interface that supports informed decision-making and comparative analysis across airlines and time periods.

The specific objectives of the project are as follows:

1. To design and implement a responsive web interface using HTML, CSS, and JavaScript that allows users to easily interact with airline price data from any modern device or browser.
2. To visualize historical airline price trends through dynamic line charts powered by Chart.js, enabling users to observe fluctuations and identify patterns over time.
3. To provide user-controlled filtering options, including the ability to select specific airlines and define custom date ranges, ensuring flexibility and focused data analysis.
4. To integrate data export and download features, such as the ability to save chart images (PNG) and export the underlying dataset (CSV), supporting academic research, reporting, and external analysis.
5. To simulate realistic price data using a random-walk algorithm for demonstration purposes, while ensuring the tool remains adaptable for integration with real-world airfare APIs or databases.
6. To ensure usability and accessibility by employing clean interface design, intuitive controls, and responsive layout principles, making the tool suitable for both technical and non-technical users.

Through these objectives, the project aims to provide a practical and extensible solution for analyzing airline pricing behavior, with potential applications in research, market studies, and travel decision support systems

1.3METHODOLOGIES TO BE FOLLOWED

The development of the Airline Price Visualization System follows a structured methodology to ensure accurate data representation, user-friendly interaction, and efficient system performance. The methodologies used in this project combine web development practices, data simulation techniques, visualization frameworks, and iterative testing. The following steps outline the approach taken:

Requirement Analysis

The first step involves identifying and understanding the needs of the system's intended users. Key requirements include the ability to visualize airline price trends, filter data by airline and date range, interact with chart elements, and export data or images for further use. This helps define the functional and non-functional specifications of the system.

System Design

A conceptual design of the system is developed, describing how components such as the user interface, chart renderer, data handler, and interaction controls will operate. Wireframes and layout structures are planned to ensure the interface is intuitive, responsive, and accessible. The design also outlines how the chart will update dynamically based on user input.

Data Preparation and Simulation

Since the project uses synthetic airline price data, a random-walk algorithm is applied to generate realistic price fluctuations for multiple airlines. A dataset containing date labels and corresponding price values is created to resemble real-world behavior. This simulated dataset also serves as a placeholder that can be replaced with live airfare API data in the future.

Front-End Development

The system is implemented using:

- HTML for structural layout,
- CSS and Tailwind CSS for styling and responsiveness, and
- JavaScript and Chart.js for interactive chart rendering.

Dynamic elements such as dropdown menus, date selectors, export buttons, and chart updates are coded to respond instantly to user actions.

Chart Integration and Interaction Logic

Chart.js is utilized to generate a smooth, interactive line chart. Programming logic is added to:

- Update datasets based on selected airline or date range,
- Control tooltips and labels,
- Generate downloadable PNG images, and
- Export visible data to CSV format.

This ensures users can explore trends and retrieve information in a usable format.

Testing and Debugging

The system undergoes iterative testing to ensure correctness, usability, and responsiveness. Tests include:

- Verifying chart accuracy with simulated data,
- Ensuring all user inputs correctly filter the visualization,
- Checking compatibility across devices and browsers,
- Testing download and export features, and
- Reviewing user interface clarity and performance..

CHAPTER 2

FUNDAMENTALS OF THE LANGUAGES USED

The development of the Airline Price Visualization System relies on three primary web technologies: HTML, CSS, and JavaScript. Each plays a critical role in the structure, appearance, and functionality of the application. Understanding the fundamentals of these languages is essential for appreciating how the system operates and how its components interact to deliver a seamless user experience.

2.1 HTML

HTML (HyperText Markup Language) is the foundational markup language used for creating the structure and layout of web pages. In this project, HTML serves as the backbone of the Airline Price Visualization System by defining the arrangement of elements such as headers, buttons, forms, and the canvas area where the price chart is rendered. Without HTML, the system would have no organized structure for users to interact with.

HTML uses a collection of tags and attributes to describe content. Each element on a webpage—such as text, images, input fields, or containers—is defined using a corresponding HTML tag. Attributes like id, class, and type provide additional information that helps control behavior and styling.

In this project, HTML performs several key functions:

1. Structural Framework
2.

HTML outlines the visual sections of the interface, including:

- The header, containing the project title and airline selection dropdown.
- The main area, where the price chart and control panels are displayed.
- The footer, which provides descriptive text or system notes.

These sections ensure a logical and accessible layout for the user.

3. User Input Controls

HTML provides form elements such as:

- <select> for choosing airlines,
- <input type="date"> for selecting date ranges,
- <button> for exporting CSV files and downloading the chart.

These elements allow users to interact with the system and customize their view of the data.

4. Chart Rendering Container
5.

A crucial HTML component in this project is the <canvas> element:

<canvas id="priceChart"></canvas>

This canvas provides the drawing space where Chart.js generates interactive visualizations. Without HTML's canvas element, the chart would not appear on the page.

6. Integration Point for Other Technologies

HTML acts as the central point where CSS and JavaScript are linked:

- CSS enhances styling and layout,
- JavaScript controls functionality and chart updates,
- External libraries like Chart.js are imported via <script> tags.

Thus, HTML ensures smooth cooperation between the visual design and the system's dynamic behavior.

2.2 HTML TAGS

HTML (HyperText Markup Language) is built around the concept of *tags*, which serve as the fundamental syntactic units used to define the structure, semantics, and behavior of content on a webpage. In the context of the Airline Price Visualization System, HTML tags form the essential scaffolding upon which styling (CSS) and functionality (JavaScript, Chart.js) are applied. Understanding how these tags operate is crucial for comprehending how the system renders its graphical interface, manages user interactions, and integrates external libraries for dynamic data visualization.

HTML tags are typically written using angle brackets and may appear in pairs—an opening tag such as and a closing tag such as —or as self-closing tags like <input />. Each tag carries specific meaning and purpose, enabling developers to construct structured web documents that browsers can interpret and display correctly. The use of HTML tags in this project ensures that the user interface is intuitive, accessible, and logically organized.

1. Purpose and Function of HTML Tags

HTML tags serve several critical functions, especially in interactive systems like this project:

1. **Structuring information** — Tags allow content to be organized into functional sections such as headers, footers, and main content areas.
2. **Defining semantics** — Semantic tags such as <header>, <main>, and <footer> give meaning to the content, improving accessibility and readability.
3. **Facilitating interactivity** — Tags such as <button>, <input>, and <select> capture user input and trigger JavaScript functions.
4. **Providing containers for scripts and styles** — <script> and <link> tags connect the webpage to external or internal resources like Chart.js and Tailwind CSS.
5. **Enabling graphic rendering** — The <canvas> tag hosts the dynamic price chart generated by JavaScript.

In this way, HTML tags act as the backbone of the entire system, defining *what* appears on the page and *how* other technologies interact with it.

2. Categories of HTML Tags Used in This Project

In building the Airline Price Visualization System, several categories of HTML tags are used. Each category contributes uniquely to the interface, allowing the system to present information clearly while supporting user interaction.

2.1 Structural and Semantic Tags

Structural tags divide the interface into logical sections that help users navigate and interact with the system smoothly.

Common examples used include:

- **<html>** — Represents the root of the document.
- **<head>** — Contains metadata, stylesheet links, and script references.
- **<body>** — Encompasses all visible content.
- **<header>** — Provides space for the project title and dropdown selections.
- **<main>** — Contains the chart and control panels.
- **<footer>** — Displays notes, credits, or instructions.
- ** ** — A generic container used for layout structuring and grouping related elements.

These tags ensure that the webpage is cleanly divided, improving usability and maintaining an organized structure.

2.2 Text and Display Tags

Text elements are crucial for labeling controls, guiding users, and giving context to the displayed data.

Commonly used text tags include:

- **<h1> to <h4>** — Title and section headings.
- **<p>** — Paragraphs for explanations and instructions.
- **<label>** — Descriptions tied to form input fields.
- **** — Emphasizes important information visually.

These text-based tags help users understand what the system does and how to operate its features.

2.3 Form and Input Tags

User input is a core component of the system's functionality. Form tags enable interaction, allowing users to modify the visualization according to their needs.

Key examples include:

- **<input type="date">** — Used for selecting start and end dates for price data.
- **<select>** — Presents a list of airlines from which the user can choose.
- **<option>** — Defines selectable items inside dropdown menus.
- **<button>** — Initiates actions such as resetting filters, exporting CSV files, or downloading charts.

These tags interact directly with JavaScript, enabling data filtering and real-time chart updates.

2.4 Media and Visualization Tags

The most important visualization-related tag used in this system is:

- **<canvas>** — A graphical surface where Chart.js generates the price graph.

The <canvas> tag does not display content by itself; rather, it acts as a container for dynamic drawings produced through JavaScript. Without this tag, the interactive airline price chart could not be rendered.

2.5 Linking and Scripting Tags

To enable styling, scripting, and visualization, HTML uses tags that connect the webpage to external files and libraries.

Examples include:

- **<link>** — Imports external CSS frameworks such as Tailwind CSS.
- **<script>** — Runs JavaScript code and connects the page to external libraries like Chart.js.

These tags ensure that the system benefits from powerful styling options and advanced visualization capabilities.

3. Attributes of HTML Tags

Most HTML tags allow attributes, which provide additional instructions or behavioral details. In this system, attributes play an essential role.

Common examples include:

- **id="priceChart"** — Allows JavaScript to select and manipulate the chart's canvas element.
- **class="control-panel"** — Applies CSS styles to specific interface sections.
- **type="date"** — Specifies the type of input expected from the user.
- **onclick=""** (in general HTML usage) — Triggers JavaScript functions when buttons are pressed.
- **download="filename.csv"** — Suggests a default name when saving exported data.

Attributes allow HTML tags to interact seamlessly with CSS and JavaScript, creating a cohesive and functional interface.

---

4. Importance of HTML Tags in the Airline Price Visualization System

HTML tags play an indispensable role in the functioning of this project:

1. **They define where dynamic components appear**, such as the interactive chart.
2. **They allow users to input data**, which becomes the basis for updating the visualization.
3. **They support styling**, allowing the interface to remain consistent and visually appealing.
4. **They enable integration with external libraries**, making advanced charting and data manipulation possible.
5. **They ensure the system is accessible**, improving usability across devices and browsers.

The correct use of tags ensures that the system operates smoothly and that all elements—from text to forms to interactive graphics—are displayed as intended.

---

5. Summary

HTML tags are the structural and semantic foundation of the Airline Price Visualization System. They determine how content is organized, how users interact with the interface, and how external styling and scripting tools integrate with the webpage. Without these tags, neither the layout nor the dynamic functionalities of the system could exist. Mastery of HTML tags is therefore essential for building, maintaining, or upgrading this type of web-based interactive application.

2.3 CSS

CSS (Cascading Style Sheets) is the styling language responsible for controlling the visual appearance and layout of web pages. While HTML defines the structure of a webpage, CSS determines how that structure is presented to the user. In the Airline Price Visualization System, CSS plays an essential role in creating an organized, attractive, and user-friendly interface. Without CSS, the system would lack visual hierarchy, spacing, color coordination, and overall usability.

CSS operates through selectors, properties, and values. Selectors identify which HTML elements should be styled, properties specify what visual aspect to modify, and values determine the final appearance. Commonly used properties include colors, borders, padding, margins, typography, and layout settings. These allow developers to enhance readability, emphasize key components, and create visually separated sections such as the chart container and control panels.

In this project, CSS handles several key interface elements. It styles the control panels where users select airlines or date ranges, ensuring that these components are clearly visible and easy to interact with. Buttons such as *Reset*, *Download PNG*, and *Export CSV* are enhanced with background colors, rounded corners, and spacing to improve user experience. The chart container is also styled to define proper height and padding, helping the Chart.js visualization stand out on the page.

Additionally, the system uses Tailwind CSS, a utility-first framework that simplifies styling by providing predefined classes for spacing, alignment, color, and layouts. This allows rapid development of a clean and modern interface without writing large amounts of custom CSS. For example, utility classes handle responsive behavior, enabling the layout to adapt across different screens, such as desktops, tablets, and mobile devices.

CSS is also responsible for maintaining visual consistency, which is vital for usability. Through careful selection of colors, text sizes, and layout spacing, CSS ensures the interface is easy to read and navigate. Buttons and input fields follow a consistent design language,

2.1 JAVASCRIPT

JavaScript is a high-level, client-side scripting language used to add interactivity, dynamic behavior, and real-time functionality to webpages. While HTML provides the structure and CSS manages presentation, JavaScript serves as the logic layer that enables the Airline Price Visualization System to respond to user actions and update content dynamically. Without JavaScript, the system would remain static, unable to filter data, update the chart, or export information.

One of JavaScript's central roles in this project is managing data processing and manipulation. The system uses JavaScript to generate simulated airline price data through a random-walk algorithm, creating realistic fluctuations for demonstration purposes. JavaScript also stores and organizes this data in arrays and objects, making it accessible for chart rendering and user-driven filtering.

Another critical function is user interaction handling. JavaScript listens for events such as selecting an airline, choosing a date range, resetting the interface, or clicking export buttons. Event listeners detect these actions and trigger the appropriate functions. For example, when a user changes the selected airline, JavaScript updates the dataset used by the chart, ensuring that the visualization reflects the new selection instantly.

JavaScript also integrates the system with Chart.js, a popular charting library used to render the interactive line graph. Through Chart.js configuration objects, JavaScript defines chart properties such as labels, datasets, colors, gradient fills, tooltips, and animation behaviors. It also enables real-time chart updates by modifying the underlying dataset and calling chart refresh methods.

Additionally, JavaScript supports export features, such as generating downloadable PNG images of the chart and exporting visible data as a CSV file. These features rely on built-in APIs like toDataURL() for downloading images and Blob objects for creating downloadable files. This functionality expands the system's usefulness for reporting and analysis.
JavaScript contributes to interface responsiveness by adjusting displayed content based on

2.4 XML

XML (Extensible Markup Language) is a flexible, text-based markup language designed to store, structure, and transport data in a human-readable and machine-readable format. Unlike HTML, which focuses on displaying information, XML focuses on describing and organizing data through custom-defined tags. This makes XML a widely used standard for data interchange across different systems, platforms, and applications. Although the Airline Price Visualization System primarily uses HTML, CSS, and JavaScript, understanding XML is important because many real-world airline and travel data services use XML-based APIs for transmitting flight information, pricing updates, and scheduling data.

A key characteristic of XML is that it allows developers to create their own tags, making the language highly adaptable to the needs of different industries. For example, an airline might use tags such as <Flight>, <Price>, <DepartureTime>, or <AirlineCode> to describe flight details in a structured manner. This flexibility enables XML to represent complex datasets in a well-organized hierarchy, which is valuable for data integration and system interoperability. XML follows a strict rule-based format. It requires properly nested tags, case sensitivity, and correct closing of elements, ensuring consistency and reducing the risk of data corruption. This makes XML particularly suitable for exchanging data between servers and applications where accuracy and structure are crucial. The ability to validate XML documents using schemas such as DTD (Document Type Definition) or XSD (XML Schema Definition) further enhances reliability by ensuring that data adheres to predefined formats.

Although modern systems often use JSON due to its simplicity and lighter syntax, XML remains widely used in enterprise environments, especially in aviation, finance, booking systems, and legacy APIs. In the context of airline data, many global distribution systems (GDS) and travel platforms still rely on XML responses to deliver fare information, seat availability, and reservation details.

In relation to the Airline Price Visualization System, XML plays an indirect but important role. If the project were to be expanded to fetch real-time airline prices from external providers, XML could serve as one of the data exchange formats.

CHAPTER 3

REQUIREMENT SPECIFICATION

The Requirement Specification defines what the Airline Price Visualization System must accomplish in order to fulfill its intended purpose. It describes the expected system behavior, performance characteristics, and operational constraints. These requirements ensure that the system is designed, implemented, and evaluated according to measurable standards. The specification is divided into functional requirements, which identify what the system must do, and non-functional requirements, which define the quality attributes and performance expectations of the system.

3.1 HARDWARE REQUIREMENTS

Since the Airline Price Visualization System is a web-based application that runs entirely in a browser, the hardware requirements for this project are minimal. The system does not rely on server-side computation or specialized hardware components, making it accessible on a wide range of devices. However, certain baseline hardware capabilities are necessary to ensure smooth performance, especially when rendering interactive charts and handling data processing tasks on the client side.

1. Minimum Hardware Requirements

These requirements represent the lowest hardware specifications needed for the system to function without significant delays:

- Processor: Dual-core CPU (1.0 GHz or higher)
- Memory (RAM): 2 GB
- Storage: At least 200 MB of free storage for browser cache and temporary files
- Display: Screen resolution of 1024 × 768 or higher
- Graphics: Integrated graphics capable of rendering HTML5 Canvas
- Input Devices: Keyboard and mouse or touch-enabled screen

These specifications allow the system to load, process simulated datasets, and render the Chart.js visualization effectively.

2. Recommended Hardware Requirements

For optimal performance, especially when working with larger datasets or multiple charts, the following hardware is recommended:

- Processor: Quad-core CPU (2.0 GHz or higher)
- Memory (RAM): 4 GB or more
- Storage: At least 1 GB free space
- Display: Full HD resolution (1920 × 1080) for better chart readability
- Graphics: Modern integrated or dedicated GPU supporting advanced Canvas rendering
- Input Devices: Standard keyboard, mouse, or touchscreen with multi-gesture support

A higher-performance device ensures smooth chart updates, faster data processing, and more responsive interactions.

3. Device Compatibility

The system is designed to run on:

- Desktop computers
- Laptops
- Tablets
- Modern smartphones

As long as the device supports a modern browser and HTML5 Canvas, it can run the application without compatibility issues.

4. Hardware Considerations

- Devices with very low RAM may experience slow chart rendering.
- Older processors may delay the generation of PNG exports or CSV files.
- Very small screens may require zooming or responsive layout adjustments.

3.2 SOFTWARE REQUIREMENTS

The Airline Price Visualization System is a browser-based application that relies entirely on client-side technologies. As a result, the software requirements are minimal and focus on ensuring compatibility with modern web standards. The system does not require any installation or backend services, making it highly accessible and easy to deploy.

1. Operating System Requirements

The system is platform-independent and can run on any operating system capable of supporting modern web browsers. Compatible systems include:

- Windows 8 / 10 / 11
- macOS
- Linux distributions (Ubuntu, Fedora, etc.)
- Android and iOS for mobile devices

Because all processing occurs in the browser, no specialized operating system components are needed.

2. Browser Requirements

A modern, HTML5-compliant browser is required to render the interface and execute JavaScript functionalities. Supported browsers include:

- Google Chrome (recommended)
- Mozilla Firefox
- Microsoft Edge
- Safari
- Opera

The browser must support HTML5 Canvas, ES6 JavaScript features, and external libraries loaded via CDN. Using the latest browser version ensures optimal performance.

3. Development Tools

Although the system can run without installation, certain tools assist in development and testing:

- Text editors or IDEs such as Visual Studio Code, Sublime Text, or Notepad++
- Built-in browser developer tools for debugging and layout inspection
- Optional version control tools like Git and GitHub for collaboration

These tools enhance maintainability but are not required for running the system.

4. Front-End Technologies

The system uses standard web technologies:

- HTML5 for structuring the interface
- CSS3 and Tailwind CSS for styling and responsive layout
- JavaScript (ES6) for interactivity and data manipulation
- Chart.js for rendering the dynamic price visualization

All libraries are included via CDN links, eliminating installation requirements.

5. Optional Software

A lightweight local server, such as the Live Server extension in Visual Studio Code, may be used during development to test the project more efficiently, though the system can run directly by opening the HTML file.

CHAPTER 4

DESIGN

4.1 DESIGN GOALS

The design goals of the Airline Price Visualization System guide the development of a user-friendly, efficient, and flexible interface capable of presenting airline price trends in a meaningful and interactive way. These goals ensure that the system is not only functional but also easy to maintain, visually appealing, and adaptable to future enhancements. By establishing clear design objectives, the project maintains a structured approach that supports usability, reliability, and long-term scalability.

The first design goal is **simplicity**. The interface should be straightforward and easy for users with varying levels of technical experience. Essential features such as airline selection, date filtering, and export options are placed in clearly visible sections of the layout. The system avoids unnecessary complexity so that users can interact with it intuitively without requiring prior training.

The second goal is **interactivity**, which is crucial for a data visualization application. Users must be able to modify filters and instantly observe changes in the displayed chart. This real-time responsiveness is achieved through a combination of JavaScript logic and the Chart.js library. Interactive elements such as hover tooltips, smooth transitions, and clickable controls help users gain deeper insight into airline price behavior.

Another important design goal is **responsiveness**. The system must function smoothly across various devices, including laptops, tablets, and smartphones. By adopting CSS and Tailwind CSS utility classes, the layout automatically adjusts to different screen sizes without compromising usability. Responsive design ensures that users can analyze data conveniently, regardless of their device.

CHAPTER 4

Visual clarity is also a key objective. Because data visualization requires users to interpret trends quickly, the chart and interface elements must be visually clean and well organized. Consistent color schemes, readable typography, appropriate spacing, and well-defined panel sections contribute to a professional and readable presentation.

The design also emphasizes **extensibility**. The system should be easy to modify or expand, allowing future integration with real airline APIs, additional chart types, predictive analytics, or enhanced visual features. Modular code structure and the use of standardized libraries support this long-term flexibility.

Finally, **performance efficiency** is a crucial design goal. The system must load quickly, update charts smoothly, and handle dataset operations without causing noticeable delays. This ensures a positive user experience and prevents frustration during data exploration.

In summary, the design goals of the Airline Price Visualization System focus on delivering a simple, interactive, responsive, visually clear, scalable, and efficient application. These goals collectively ensure that the system meets user expectations while providing a solid foundation for future enhancements.


Airline Price Explorer
Interactive HTML + Chart.js demo — select airline and date range.
Airline
BlueHorizon
Price (USD)
Date
Date range
13-11-2025
12-12-2025
Default: last 30 days
Actions
Reset to 30 days
Download chart PNG
Export CSV
Tip: Hover the chart to see exact price at a date. Use the date inputs to zoom in.
Demo generated sample data — replace with your API data by setting rawData in the script.

Fig 4.1

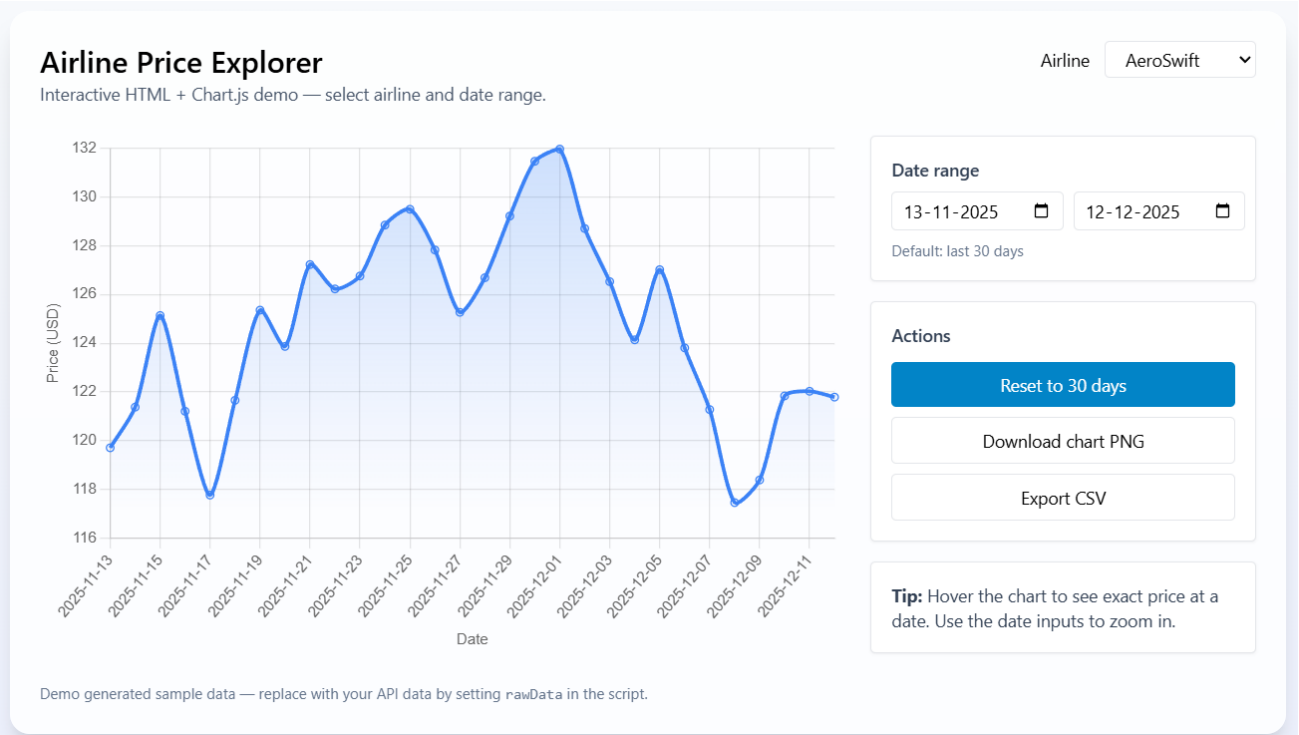
Airline Price Explorer
Interactive HTML + Chart.js demo — select airline and date range.
Airline
AeroSwift
Price (USD)
Date
Date range
13-11-2025
12-12-2025
Default: last 30 days
Actions
Reset to 30 days
Download chart PNG
Export CSV
Tip: Hover the chart to see exact price at a date. Use the date inputs to zoom in.
Demo generated sample data — replace with your API data by setting rawData in the script.

Fig 4.2

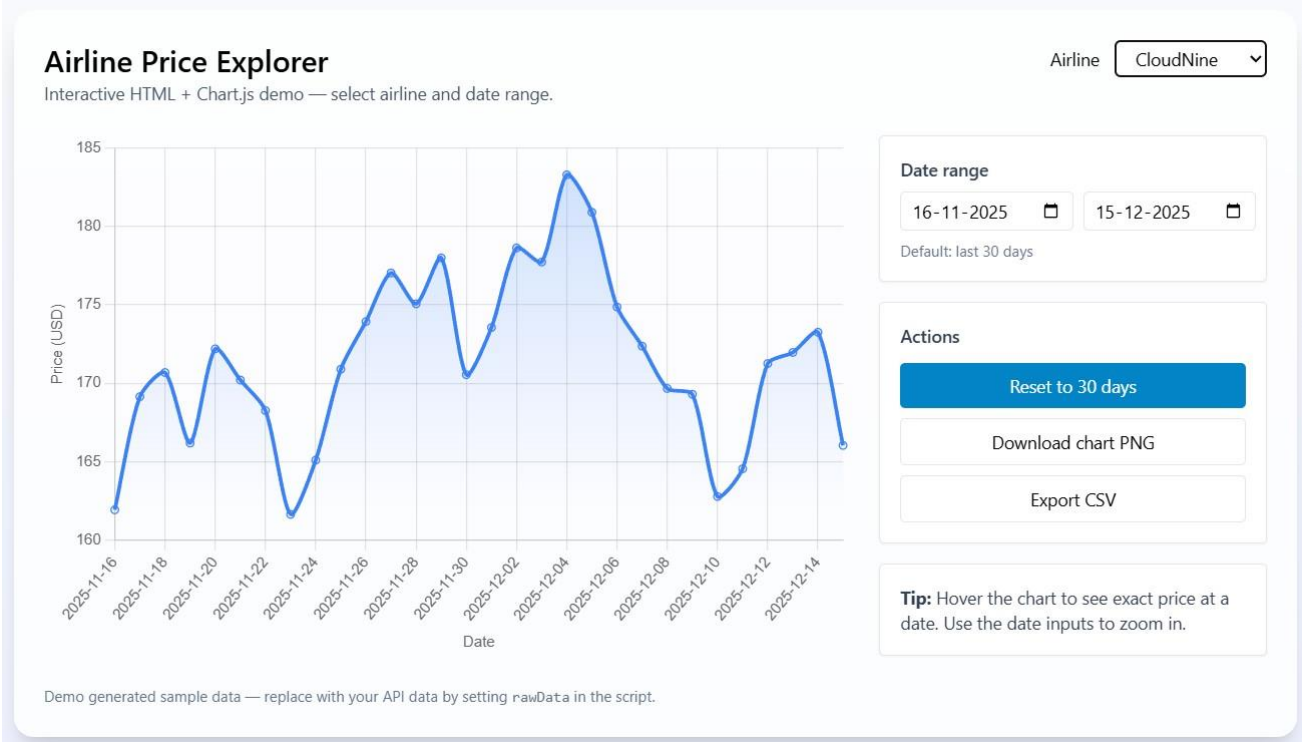
Airline Price Explorer
Interactive HTML + Chart.js demo — select airline and date range.
Airline
CloudNine
Price (USD)
185
180
175
170
165
160
2025-11-16
2025-11-18
2025-11-20
2025-11-22
2025-11-24
2025-11-26
2025-11-28
2025-11-30
2025-12-02
2025-12-04
2025-12-06
2025-12-08
2025-12-10
2025-12-12
2025-12-14
Date
Date range
16-11-2025
15-12-2025
Default: last 30 days
Actions
Reset to 30 days
Download chart PNG
Export CSV
Tip: Hover the chart to see exact price at a date. Use the date inputs to zoom in.
Demo generated sample data — replace with your API data by setting rawData in the script.

Fig4.3

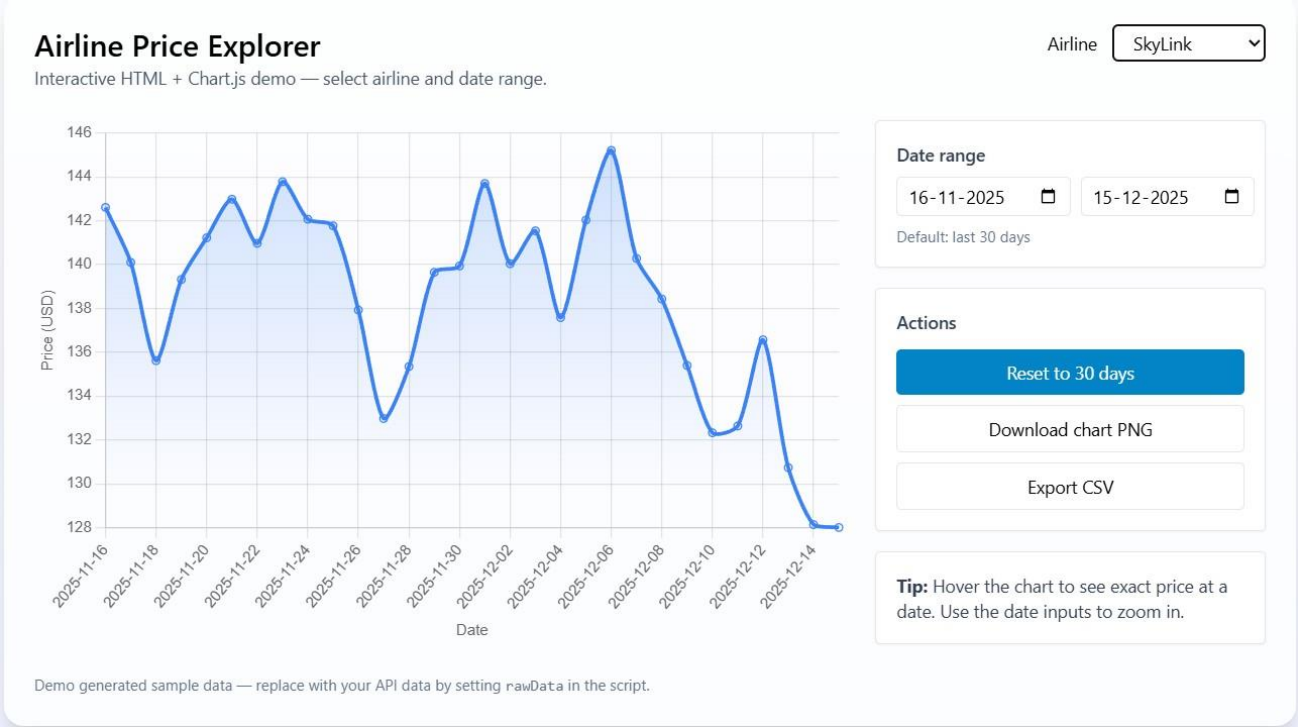
Airline Price Explorer
Interactive HTML + Chart.js demo — select airline and date range.
Airline
SkyLink
Price (USD)
146
144
142
140
138
136
134
132
130
128
2025-11-16
2025-11-18
2025-11-20
2025-11-22
2025-11-24
2025-11-26
2025-11-28
2025-11-30
2025-12-02
2025-12-04
2025-12-06
2025-12-08
2025-12-10
2025-12-12
2025-12-14
Date
Date range
16-11-2025
15-12-2025
Default: last 30 days
Actions
Reset to 30 days
Download chart PNG
Export CSV
Tip: Hover the chart to see exact price at a date. Use the date inputs to zoom in.
Demo generated sample data — replace with your API data by setting rawData in the script.

Fig 4.4

CHAPTER 5

IMPLEMENTATION

The implementation phase of the Airline Price Visualization System focuses on transforming the design specifications into a functioning application that effectively visualizes airline pricing trends. The system is entirely web-based and relies exclusively on client-side technologies, which allows it to run without requiring installation or backend support. This phase integrates structural HTML, styling through CSS and Tailwind CSS, dynamic functionality using JavaScript, and powerful chart rendering through the Chart.js library. The implementation emphasizes modular coding practices, performance, and maintainability to ensure that the system is both efficient and scalable.

5.1 HTML Implementation

The implementation begins with constructing the foundational structure of the application using HTML. A semantic layout is created to ensure clarity, organization, and accessibility. The primary sections of the page include the header, main content area, control panel, chart container, and footer. These sections are arranged using semantic tags such as <header>, <main>, <section>, and <footer>, which improve readability and support modern web accessibility standards.

Within the HTML structure, the most crucial element is the <canvas> tag, which serves as the drawing area for the Chart.js library. The canvas is assigned a unique id so that it can be easily accessed and manipulated via JavaScript. In addition, the HTML file contains user interface elements such as dropdown menus for selecting airlines, date input fields for selecting start and end dates, and buttons for resetting the interface or exporting data. These elements are interconnected with JavaScript functions through unique identifiers (id attributes), ensuring smooth interaction between the user interface and the underlying application logic.

External libraries and frameworks, including Tailwind CSS and Chart.js, are linked through <script> and <link> tags. This approach eliminates the need for installation, reduces system complexity, and ensures that updates to these libraries

can be easily applied.

5.2 CSS and Tailwind CSS Implementation

Styling is implemented using a combination of custom CSS and utility classes provided by Tailwind CSS. This hybrid approach offers both flexibility and efficiency. Custom CSS is used to refine specific components such as the chart container, control panel, and buttons. For example, rules governing box shadows, rounded borders, background transparency, and layout spacing help distinguish different parts of the interface and improve the user experience.

Tailwind CSS plays a major role in ensuring responsive design. Its utility-first classes allow quick application of consistent margins, padding, color schemes, alignment properties, and fonts. For instance, classes such as flex, grid, text-sm, rounded-lg, shadow, and p-4 are used extensively to structure the layout and improve aesthetics. This approach speeds up the development process and ensures that the interface scales gracefully across multiple screen sizes.

Responsiveness is a major implementation goal, and Tailwind CSS addresses this by providing device-specific breakpoints. For smaller devices like tablets and smartphones, layout components automatically stack vertically rather than horizontally. This ensures that the chart and control inputs remain accessible and readable on all devices, enhancing the system's usability.

5.3 JavaScript Implementation

JavaScript forms the core logic of the system and governs most of its dynamic behavior. The initial step in JavaScript implementation is generating simulated airline price data using a random-walk algorithm. This algorithm creates realistic fluctuations by introducing small, controlled variations in price values. The generated dataset includes multiple airlines, each represented by an array of prices corresponding to specific dates. These datasets are stored in a rawData object, allowing the system to easily retrieve and filter values based on user selections.

Interactive functionality is implemented through event listeners that respond to user actions. For example, when a user selects an airline from the dropdown menu or adjusts the date inputs, JavaScript captures the event and triggers an update

function. This function filters the dataset, adjusts the displayed date range, and refreshes the chart accordingly. The interactive nature of these updates ensures that the system responds immediately to user inputs without requiring page reloads.

JavaScript also includes methods for validating user inputs. If the selected date range exceeds the available dataset, JavaScript automatically clamps the values to prevent errors. This ensures that the system remains stable and reliable, even when user input is imprecise or incomplete.

5.4 Chart.js Implementation

The Chart.js library handles the visual rendering of airline price trends. The implementation begins with initializing a line chart and configuring its properties, including datasets, gradient colors, line tension, tooltips, axis labels, and fill behavior. The chart uses smooth transitions and interactive tooltips to enhance the visualization experience.

One of the key implementation features is the real-time update capability. When the dataset or date range changes, JavaScript modifies the chart's data object and calls the .update() method. This efficiently redraws the chart without reinitializing the entire visualization. Gradient fills are applied to emphasize the pricing curve, making the data easier to interpret. Tooltips provide detailed information when the user hovers over specific data points, adding depth to the visualization.

Chart.js also integrates seamlessly with export functions. For example, the chart can be converted into a PNG image by invoking the toBase64Image() method. This enables the user to download visualizations for use in reports or presentations.

5.5 Export and Download Functionality

The system provides two export features:

1. Download Chart as PNG
2. Export Data as CSV

The PNG export uses Chart.js's built-in image conversion capabilities. JavaScript creates a hidden download link, assigns the base64 image string as the link source, and triggers a click event to initiate download automatically.

The CSV export feature uses JavaScript's Blob API to construct a text file containing date and price values. A dynamic URL is generated using URL.createObjectURL(), and a temporary <a> tag triggers the download. This enables users to analyze the data further in spreadsheet applications such as Microsoft Excel or Google Sheets.

5.6 Summary of Implementation

The implementation of the Airline Price Visualization System effectively combines markup, styling, scripting, and visualization technologies into a cohesive application. HTML structures the interface, CSS and Tailwind CSS enhance visual quality and responsiveness, JavaScript manages interactivity and data processing, and Chart.js produces a dynamic, visually appealing chart. The modular, client-side design ensures maintainability, scalability, and compatibility across multiple devices and platforms. The system is implemented in a way that supports future enhancements such as API integration or predictive

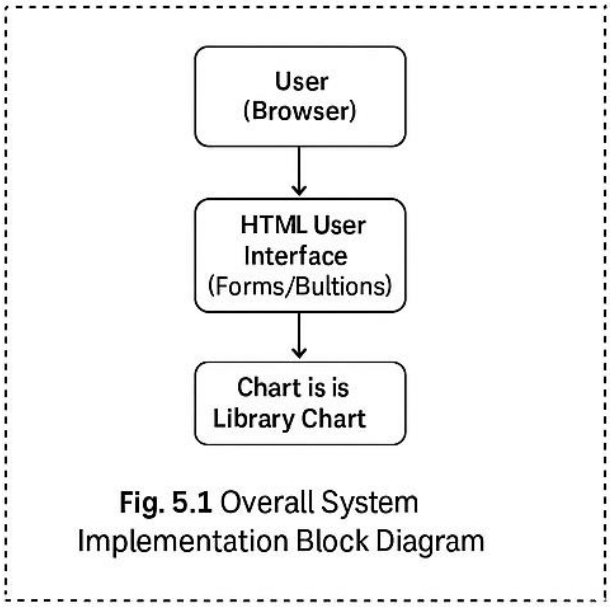

Fig. 5.1 Overall System Implementation Block Diagram

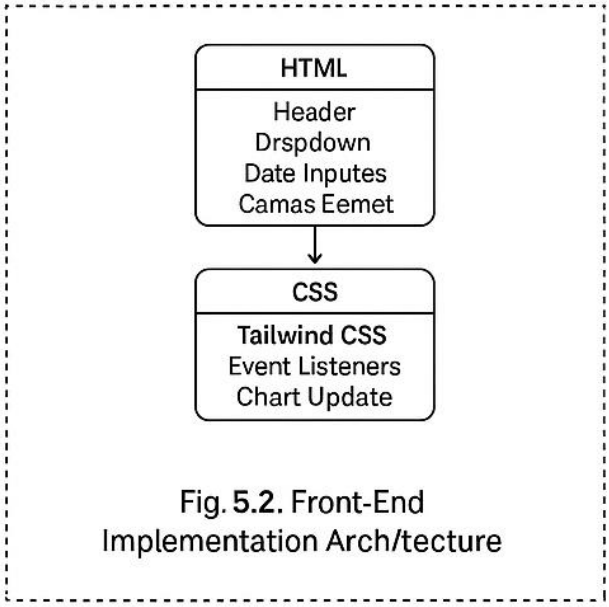

Fig. 5.2. Front-End Implementation Arch/tecture

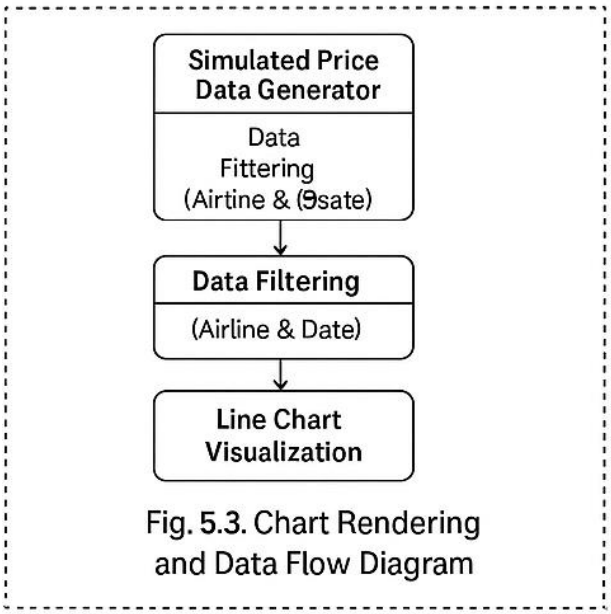

Fig. 5.3. Chart Rendering and Data Flow Diagram

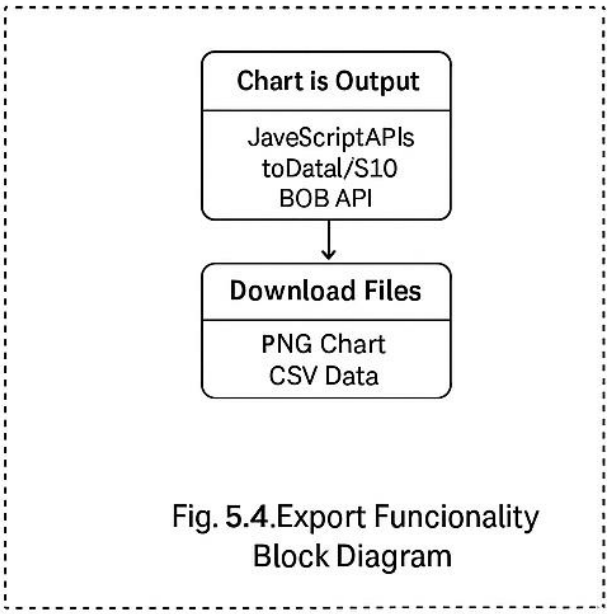

Fig. 5.4.Export Funcionality Block Diagram

CHAPTER 6

RESULTS

The Airline Price Visualization System was successfully implemented and tested using modern web technologies including HTML, CSS, JavaScript, Tailwind CSS, and Chart.js. The results demonstrate that the system effectively meets the objectives defined during the design and implementation phases by providing a responsive, interactive, and user-friendly platform for analyzing airline ticket price trends.

Upon execution, the application loads quickly in a standard web browser without requiring any installation or backend services. The user interface is displayed clearly with well-organized components such as airline selection dropdowns, date range inputs, and action buttons. The responsive design adapts smoothly to different screen sizes, ensuring usability on desktops, laptops, tablets, and mobile devices.

The interactive line chart generated using Chart.js accurately displays simulated airline price data over the selected time period. Price fluctuations are clearly visible, allowing users to identify trends, peaks, and variations with ease. The hover-based tooltips provide precise price values for specific dates, enhancing data readability and analytical accuracy. When users change the selected airline or adjust the date range, the chart updates instantly without page reloads, confirming the effectiveness of JavaScript event handling and real-time data processing.

The simulated data generation using a random-walk algorithm produces realistic airline pricing behavior, which is sufficient for demonstration and testing purposes. This confirms that the system can function independently of external data sources while remaining ready for future integration with real-time airline APIs.

The export functionality performed as expected. Users were able to download the displayed chart as a PNG image, making it suitable for inclusion in reports and presentations. Additionally, the CSV export feature successfully generated downloadable files containing date-wise price data, which could be opened and analyzed in spreadsheet applications such as Microsoft Excel and Google Sheets.

CHAPTER 7

CONCLUSION

The Airline Price Visualization System developed in this project successfully demonstrates the effective use of modern web technologies for interactive data analysis and visualization. By integrating HTML, CSS, JavaScript, Tailwind CSS, and Chart.js, the system provides a responsive and user-friendly platform that enables users to explore and understand airline ticket price fluctuations over time.

The project achieved its primary objectives by transforming simulated airline price data into an intuitive graphical format. Users can select different airlines, adjust date ranges, and instantly view updated price trends through an interactive line chart. The real-time responsiveness of the system, combined with features such as hover-based tooltips and smooth chart transitions, enhances data clarity and supports informed analysis without requiring technical expertise.

The implementation of export features further increases the practical value of the system. The ability to download the chart as a PNG image and export the underlying dataset as a CSV file allows users to utilize the results for academic studies, reporting, and further statistical analysis. The use of a random-walk algorithm for data simulation effectively demonstrates system functionality while maintaining flexibility for future integration with real-time airline data sources.

Overall, the project validates that browser-based visualization tools can efficiently present complex pricing information in an accessible and meaningful way. The modular design, responsive interface, and reliance on widely supported web standards make the system scalable and easy to enhance. Future improvements may include integration with live airline APIs, predictive price modeling, advanced comparison features, and enhanced user analytics.

In conclusion, the Airline Price Visualization System serves as a practical and extensible solution for analyzing airline price trends and highlights the potential of web-based technologies in supporting data-driven decision-making in the aviation domain**.**

REFERENCES

1. Chart.js, JavaScript Charting Library Documentation.
2. Mozilla Developer Network, HTML (HyperText Markup Language) Documentation.
3. Mozilla Developer Network, CSS (Cascading Style Sheets) Documentation.
4. Mozilla Developer Network, JavaScript Language Guide.
5. Tailwind CSS, Utility-First CSS Framework Documentation.
6. W3Schools, Web Development Tutorials for HTML, CSS, and JavaScript.
7. Flanagan, David, JavaScript: The Definitive Guide, O'Reilly Media, 7th Edition, 2020.
8. Sommerville, Ian, Software Engineering, Pearson Education, 10th Edition, 2016.
9. Duckett, Jon, HTML and CSS: Design and Build Websites, John Wiley & Sons, 2011.
10. Duckett, Jon, JavaScript and jQuery: Interactive Front-End Web Development, John Wiley & Sons, 2014.
11. Pressman, Roger S., Software Engineering: A Practitioner's Approach, McGraw-Hill Education, 8th Edition, 2015.
12. Few, Stephen, Information Dashboard Design: Displaying Data for At-a-Glance Monitoring, Analytics Press, 2nd Edition, 2013.