DESIGN AND IMPLEMENTATION OF NUMBER SYSTEM SIMULATOR

A CAPSTONE PROJECT REPORT

Submitted in the partial fulfilment for the Course of

CSA1220 – Computer Architecture for future engineers

to the award of the degree of

BACHELOR OF ENGINEERING

Submitted by

A.Kailash venkata sai (192525069)

B.Chandu (192525289)

Under the Supervision of

DR

SIMATS ENGINEERING

Saveetha Institute of Medical and Technical Sciences

Chennai-602105

August 2025

SIMATS ENGINEERING

Saveetha Institute of Medical and Technical Sciences

Chennai-602105

Engineer to Excel

DECLARATION

We**, A.Kailash venkata sai,B.Chandu** of Saveetha Institute of Medical and Technical Sciences, Saveetha University, Chennai, hereby declare that the Capstone Project Work entitled **'Design and Implementation of number system simulator**' is the result of our own bonafide efforts. To the best of our knowledge, the work presented herein is original, accurate, and has been carried out in accordance with principles of engineering ethics.

Place: Chennai

Date:

Signature of the Students with Names

A.Kailash venkata sai (192525069)

B.Chandu (192525)

SIMATS ENGINEERING

Saveetha Institute of Medical and Technical Sciences

Chennai-602105

Engineer to Excel

BONAFIDE CERTIFICATE

This is to certify that the Capstone Project entitled “Design and implementation of number system simulator” has been carried out by A.Kailash venkata sai,B.Chandu under the supervision of Dr. and is submitted in partial fulfilment of the requirements for the current semester of the B.Tech AIML and Engineering program at Saveetha Institute of Medical and Technical Sciences, Chennai.

SIGNATURE
Dr. Anusuya
Program Director

Computer Science and Engineering

Saveetha School of Engineering

SIMATS

SIGNATURE
Dr.
Associate Professor

IC – Intelligent Computing

Saveetha School of Engineering

SIMATS

Submitted for the Project work Viva-Voce held on _____ .

INTERNAL EXAMINER

EXTERNAL EXAMINER

ACKNOWLEDGEMENT

We would like to express our heartfelt gratitude to all those who supported and guided us throughout the successful completion of our Capstone Project. We are deeply thankful to our respected Founder and Chancellor, Dr. N.M. Veeraiyan, Saveetha Institute of Medical and Technical Sciences, for his constant encouragement and blessings. We also express our sincere thanks to our Pro-Chancellor, Dr. Deepak Nallaswamy Veeraiyan, and our ViceChancellor, Dr. S. Suresh Kumar, for their visionary leadership and moral support during the course of this project.

We are truly grateful to our Director, Dr. Ramya Deepak, SIMATS Engineering, for providing us with the necessary resources and a motivating academic environment. Our special thanks to our Principal, Dr. B. Ramesh for granting us access to the institute's facilities and encouraging us throughout the process. We sincerely thank our Head of the Department, **Dr.Anusuya** for his continuous support, valuable guidance, and constant motivation.

We are especially indebted to our guide, **Dr.** for his creative suggestions, consistent feedback, and unwavering support during each stage of the project. We also express our gratitude to the Project Coordinators, Review Panel Members (Internal and External), and the entire faculty team for their constructive feedback and valuable inputs that helped improve the quality of our work. Finally, we thank all faculty members, lab technicians, our parents, and friends for their continuous encouragement and support.

Signature With Student Name

A.Kailash venkata sai(192525069)

B.Chandu(192525289)

Table of Contents

LIST OF FIGURES

ABSTRACT

In the digital age, number systems form the foundational framework for all computing processes. From data representation to memory addressing and instruction execution, number systems—particularly binary, decimal, octal, and hexadecimal—play a critical role in modern computer architecture. However, students and early-stage developers often find it challenging to understand and manually convert between these systems due to complex logic and repetitive calculations. To bridge this learning gap and provide a practical, interactive tool, this capstone project introduces a "Number System Simulator", a web-based application designed to simplify and visualize number system conversions. The project leverages HTML, CSS, and JavaScript for front-end development, and integrates a lightweight Flask (Python) backend to host the application locally. The user-friendly interface allows users to input numbers in any selected base (2, 8, 10, or 16), and with a single click, instantly view accurate conversions across all other number systems. The system supports both uppercase and lowercase hexadecimal inputs, leading/trailing zero handling, and error management for invalid entries. Each result is displayed clearly and formatted with responsive design principles for better readability. This simulator not only automates the conversion logic but also deepens user understanding by demonstrating how the same value can appear differently across number systems. The primary goal of this project is to enhance conceptual clarity for students, reduce manual errors, and support academic curricula involving digital logic design, microprocessors, and computer architecture.

CHAPTER 1
INTRODUCTION

1.1 Background

In the realm of computer science and digital electronics, number systems form the foundational structure upon which all operations are built. Computers fundamentally operate using binary logic, and every computation, data storage operation, or signal transmission involves numbers expressed in binary, octal, decimal, or hexadecimal systems. These number systems are vital for tasks such as memory addressing, instruction encoding, and digital circuit design. While the decimal system is most familiar to humans, computers rely on binary and its derivatives due to their simplicity in representing two distinct states (0 and 1).

Despite their importance, students often find it difficult to grasp the conversion logic and usage of different number systems. Converting values between bases requires strong conceptual understanding and practice. Manual methods are prone to calculation errors and can be time-consuming. Furthermore, traditional teaching methods sometimes fail to provide visual or interactive experiences that make learning number systems more intuitive and engaging.

To address these learning challenges and to support students in acquiring deeper insight into number system operations, this project presents an interactive tool— Number System Simulator—that simplifies conversions between base systems and enhances learning through instant, accurate, and visual feedback.

1.2 Project Objectives

The main goal of this project is to help users convert numbers between binary, octal, decimal, and hexadecimal systems with just a few clicks. The tool is designed in such a way that even a beginner can use it without any difficulty. The objectives are:

- To build a user-friendly tool that makes number system conversion easy and fast.
- To reduce the chances of human error during conversion by doing it automatically.
- To help students and teachers use the tool for learning and teaching purposes.
- To build the project using basic web technologies like HTML, CSS, JavaScript, and Python (Flask).

- To allow future updates like adding arithmetic operations, binary math, and more features.

This simulator will not only be useful for students but also for developers, teachers, and anyone who wants to learn how number systems work.

1.3 Significance of the Project

This project is very useful because it helps solve a real problem faced by many students. Usually, when students study computer architecture or digital logic, they struggle with converting numbers from one base to another. This simulator saves their time and helps them learn faster. It is a great tool for both practice and real time checking. Teachers can also use it during class demonstrations to explain concepts more clearly. As the tool runs on a browser, it doesn't require any installation and can be accessed from any system. It is also helpful during online learning, as students can use it from home. The tool gives instant answers, has a clean look, and is easy to understand. Because of all these reasons, the Number System Simulator is an important and helpful project for education.

1.4 Scope of the Project

The current version of the project includes:

- A single input box where the user can type any number.
- A dropdown to select the base of the input number (binary, decimal, octal, or hexadecimal).
- Instant conversion of the input into the other number systems.
- Error checking to make sure the user enters correct values.
- A clean and simple result section that shows all converted values.

The project does not include features like:

- Step-by-step explanation of the conversion process.
- Arithmetic operations (like addition or subtraction in binary).
- Bitwise operations (like AND, OR, NOT).
- Storage or history of previous results.

These features can be added later as future improvements.

1.5 Methodology Overview

To successfully build the Number System Simulator, we followed a step-by-step development method that focused on simplicity, accuracy, and ease of use. We used a combination of web development and backend tools to create an interactive tool that runs directly in a web browser. Our goal was to make the simulator lightweight, fast, and easy for students or teachers to access on any system without the need for installation.

The first step was to design the user Interface using HTML (HyperText Markup Language) and CSS (Cascading Style Sheets). HTML was used to create the structure of the web page, such as the input box, dropdown menu, convert button, and result display area. CSS helped us style the page and make it visually clean, centered, and user-friendly. We chose colors that are pleasant to the eyes, added spacing between elements, and ensured that the tool looks neat on any screen size. We also made sure that the layout is responsive, meaning it works well on both computers and mobile devices.

The second step was to build the main functionality using JavaScript, which is the programming language that runs in the browser. JavaScript was used to write the logic that performs number system conversions. When a user types a number and selects its base (binary, octal, decimal, or hexadecimal), the JavaScript code automatically converts that number into the other three systems. For example, if the user enters a binary number like “1010,” the code will calculate its decimal, octal, and hexadecimal equivalents and show them clearly on the screen. JavaScript also checks whether the input is valid. If the user enters a wrong character (for example, entering the letter “G” in hexadecimal), the simulator displays a friendly error message telling the user that the input is invalid. This makes the tool safe and reliable to use.

To run the application on a local computer using a browser, we used Flask, which is a simple web framework written in Python. Flask helped us serve the HTML file so that the project works as a small web application. Flask is lightweight and easy to use for local deployment. It handles the basic routing required to open the simulator through a local link like http://127.0.0.1:5000/. Although all the conversion logic is written in JavaScript on the front end, Flask allows us to organize the files better and simulate a real-world web development

structure. This will also make it easier to add more features later on, such as connecting a database or deploying it online.

During the development, we made sure the code was written in a modular way. This means each part of the code is separated clearly—HTML for structure, CSS for design, JavaScript for logic, and Python for running the server. This makes the project easier to understand, test, and update. For example, if we want to add a new feature like binary addition or step-by-step explanation, we can do that without changing the whole code. We tested the tool by entering different inputs in all base systems and checking if the output matched the expected results. The tool performed well and gave correct answers every time.

In summary, the methodology we followed involved designing a clean and simple user interface, writing accurate conversion logic, and setting up a working local web app using Flask. This approach made the development process smooth, and the result is a reliable, educational tool that helps users convert numbers between different systems easily and correctly.

CHAPTER 2

PROBLEM IDENTIFICATION AND ANALYSIS

2.1 Description of the Problem

One of the most common problems faced by students who are new to computer science is understanding how number systems work. In regular life, we only use the decimal system, which has ten digits (0 to 9). But in computers, we deal with multiple systems like binary (base 2), octal (base 8), decimal (base 10), and hexadecimal (base 16). These number systems are not just theory—they are used everywhere in computer memory, logic gates, assembly programming, data encoding, and even while writing simple programs. To understand how a computer stores numbers, performs calculations, and processes information, it is important to know how to convert values from one number system to another. Unfortunately, many students find this confusing and difficult when they are first introduced to the topic.

One of the biggest reasons this is a problem is because the conversion steps can be long and involve multiple rules. For example, converting a hexadecimal number to binary involves first converting each digit into a 4-bit binary number. Similarly, converting a binary number into decimal involves understanding powers of two and adding the weighted values. These steps may seem small, but for a beginner, especially someone not confident in mathematics, this becomes stressful. Students often make simple mistakes like placing the wrong power, adding incorrectly, or misunderstanding what a digit means in a different base. Even one small mistake can lead to a wrong answer, and this discourages students from practicing further.

Moreover, in many colleges and schools, these topics are taught only briefly in theory. Students are expected to practice them manually using pen and paper. While this is useful for exams, it does not give instant feedback, which means students don’t know if their answers are correct unless a teacher checks it. In classrooms with many students, teachers may not have the time to check every student’s work. As a result, students memorize rules without actually understanding them. This creates a weak foundation and affects their learning in future topics like digital circuits, microprocessors, and data representation.

There are some online tools that do number system conversion, but most of them are either too technical, contain ads, or are not designed for learning. Many tools just give the final answer without explaining or checking whether the input is valid. Also, some tools are not mobile-

friendly or have confusing layouts. Because of this, students don't feel confident using them, and many avoid these tools completely.

In exams, interviews, or lab practical, when students are asked to convert numbers quickly and accurately, many of them struggle. This shows that even though the topic seems small, the lack of a simple and helpful learning tool becomes a serious problem. The current learning methods do not offer enough support or instant verification, and this slows down the learning process. A tool that allows users to enter a number and instantly see its equivalent in other systems would be a big help. It would save time, improve understanding, and give students the confidence they need to succeed.

This project, therefore, is focused on solving this real problem faced by thousands of students. By creating a simple and clear Number System Simulator, we aim to reduce the burden of manual conversion, avoid confusion, and allow students to focus on understanding the core concept without getting stuck on calculation errors. This simulator is not just a tool; it is a learning companion that gives fast, accurate, and useful feedback every time.

2.2 Evidence of the Problem

The problem of understanding number system conversions is very common among students studying subjects like Computer Fundamentals, Digital Logic Design, or Microprocessors. Based on classroom experiences, feedback from teachers, and online forums, it is clear that many learners search for easier ways to learn these conversions. Students often rely on calculators or random websites that may not be accurate or user-friendly. In some cases, they avoid learning the process altogether and just memorize results, which is not a good practice. Additionally, when given large binary or hexadecimal numbers, they become nervous or take too long to complete the task. The lack of a clean and easy tool for practicing conversions creates stress and makes the learning process slower. Teachers also face difficulty explaining all possible conversions within the limited time of a class. This situation clearly shows that there is a gap in resources available for learning number systems practically and comfortably**.**

2.3 Stakeholders

The main stakeholders affected by this problem are students, especially those in high school, diploma, and undergraduate courses related to computer science or electronics. These students need to understand number systems as part of their syllabus, and it forms the base for many

higher-level topics. Teachers and trainers are also important stakeholders because they need tools that can help explain concepts faster and more clearly. With the help of such a simulator, they can demonstrate conversions instantly and answer students' doubts during class. Institutions and colleges can also benefit because such tools can be used in labs or online classes to improve learning quality. Finally, developers or self-learners who are studying on their own can use this tool to strengthen their basic knowledge.

2.4 Supporting Data/Research

Several online surveys, academic discussions, and teaching reviews suggest that many students struggle with number system conversions, especially in early semesters. Sites like Stack Overflow, Quora, and Edmodo are filled with questions from learners asking for help with binary, octal, and hexadecimal conversions. YouTube videos explaining the topic often have millions of views, which shows the high demand for understanding this concept better. Additionally, existing conversion websites are either too complex, filled with ads, or do not show the process clearly. Research also shows that interactive learning is more effective than static learning. When learners see results in real-time and get feedback instantly, they understand faster and retain knowledge longer. This supports the need for a tool like the Number System Simulator, which is simple, fast, and focused only on this one learning goal.

CHAPTER 3

SOLUTION DESIGN AND IMPLEMENTATION

3.1 Development and Design Process

The development of the Number System Simulator followed a clear and step-bystep process to ensure the application would be simple, functional, and useful for students. The first step was understanding what the user really needs. We realized that most students were looking for a tool where they could enter any number— whether binary, decimal, octal, or hexadecimal—and instantly see how it looks in other number systems. So we decided to make a web-based application because it can be easily accessed from any device with a browser, without installing anything.

After finalizing the idea, we designed the layout of the web page using HTML. We placed the input box, the dropdown to select the number system, a convert button, and a result display area in the center of the screen so it looks neat and user-friendly. Once the layout was ready, we added CSS to style the simulator. We chose colors that are easy on the eyes and used padding, borders, and spacing to make the tool look clean and professional. We also ensured the layout is responsive, so the simulator works well on laptops, tablets, and smartphones.

The most important part of the simulator is the conversion logic. This was developed using JavaScript, which runs directly in the browser and doesn't require a page reload. We wrote functions to convert numbers from the input base to all other formats. For example, if a user enters a hexadecimal number, the JavaScript function first converts it to decimal, and from there calculates binary and octal equivalents using built-in methods. We also added error-checking so that if a user enters a wrong character or an invalid number for the selected base, the tool shows a clear error message. This makes the tool reliable and safe to use even for beginners.

3.2 Tools and Technologies Used

To build this project, we used a combination of front-end and back-end tools. Each tool had a specific purpose and helped us complete the project smoothly. Here are the main tools and technologies used:

- **HTML (Hyper Text Markup Language):** Used to build the structure of the webpage, including text boxes, dropdown menus, and result display sections.

- **CSS (Cascading Style Sheets):** Used to make the page visually appealing and clean. It added styles like colors, margins, borders, and fonts.
- **JavaScript**: Used to write the logic that performs the number system conversions. It runs in the browser and updates the results instantly without refreshing the page.
- **Python**: Used for the back-end part of the application.
- **Flask Framework**: A lightweight Python web framework used to host the HTML page on a local server. It allows the user to run the simulator using a link like http://127.0.0.1:5000/.
- **VS Code**: Used as the main code editor during development.
- **Browser Console**: Used for testing and debugging the JavaScript logic during development.

3.3 Solution Overview

The Number System Simulator works as a simple but powerful educational tool. The user opens the simulator in a web browser. They will see a text box to enter a number, a dropdown to select the base of that number (binary, decimal, octal, or hexadecimal), and a convert button. Once the user enters a value and clicks "Convert," the simulator instantly calculates the equivalent values in the other three number systems and shows the result.

Behind the scenes, the JavaScript code takes the input and parses it using parseInt(value, base) where the base is based on the user's dropdown selection. It then uses toString() methods with base 2, 8, 10, and 16 to convert the number and display it in binary, octal, decimal, and hexadecimal. The converted values are printed inside a result box styled using CSS.

If the user enters an invalid value (e.g., typing "2" in binary or using a non-number character), the code identifies this and displays a warning message instead of performing the conversion. This helps prevent wrong calculations and makes the simulator a trusted tool.

3.4 Engineering Standards Applied

While this project is academic, we made sure to follow proper development practices that are also used in industry projects. These include:

- Clean Code Structure: Code is written in a modular way where HTML, CSS, and JavaScript are separated clearly. This follows standard web development practice.

- Error Handling: The tool checks for invalid inputs and doesn't crash or give wrong results. This shows good validation practices.
- Responsive Design: The layout adjusts to different screen sizes. This is part of modern web UI design standards.
- Accessibility: Text and buttons are large and clear, so users can interact with it easily.
- Cross-browser Support: The simulator works well on all major browsers (Chrome, Firefox, Edge).
- Use of Standard Functions: We used standard JavaScript methods for parsing and converting numbers, ensuring accuracy and reliability.

3.5 Solution Justification

The solution provided by the Number System Simulator is simple but effective. It solves the exact problem faced by many students: confusion and errors while converting numbers between different number systems. Instead of relying on random websites or doing conversions manually, students now have access to a tool that gives them instant, correct results and helps them learn by observation and practice.

We chose to use web technologies because they are light, fast, and platformindependent. The user does not need any special software or setup to use the tool. Just opening the HTML page in a browser is enough. This makes the solution more accessible and useful in both classrooms and at home.

The use of Flask was justified because it allows the app to be hosted locally, simulating a real-world project deployment. Even though we didn't use Flask to perform any backend logic, setting it up prepares the tool for future upgrades, such as adding databases or hosting it online. The simple interface, strong validation, clean layout, and instant output all together make this simulator a strong solution to a common student problem.

CHAPTER 4

RESULTS AND RECOMMENDATIONS

4.1 Evaluation of Results

After the development of the Number System Simulator was completed, we tested it thoroughly with various inputs in all number systems. The simulator performed exactly as expected. When a valid number was entered, it gave the correct output in all the remaining three number systems within seconds. For example, if the input was a binary number like 1010, the tool correctly showed 10 in decimal, 12 in octal, and A in hexadecimal. Similarly, when we entered a hexadecimal number like 1F, it accurately returned its binary, decimal, and octal forms. The results were clear, well-formatted, and instant, which makes the tool very reliable.

The simulator also handled invalid inputs successfully. For example, when we entered letters in binary mode (which allows only 0 and 1), the tool showed an error message saying “Invalid input.” This feature ensured that users cannot mistakenly enter the wrong type of number, which adds to the tool's usefulness for learning. The layout was responsive and worked properly on different devices, including desktop computers and mobile phones.

We also received informal feedback from classmates and test users who found the tool simple, helpful, and accurate. They appreciated the instant response, easy interface, and correctness of the output. Based on this, we can say that the project achieved its goal of helping students convert numbers between different number systems easily and correctly.

4.2 Challenges Encountered

Like any project, we faced a few challenges during development. The first challenge was input validation. Each number system allows only specific characters (e.g., binary only allows 0 and 1; hexadecimal allows digits 0–9 and letters A–F). We had to write separate logic to check if the user’s input matched the selected number system. It took some time to handle all possible wrong entries and give clear error messages.

Another challenge was making the layout responsive. Initially, the simulator looked good only on larger screens. When we tested it on smaller devices, the input box and result area were not aligned properly. We had to modify the CSS code and test it on multiple devices to make sure the tool looked good everywhere.

One more issue was managing case sensitivity in hexadecimal inputs. For example, users may enter either ‘A’ or ‘a’ in hex. The JavaScript conversion logic is casesensitive, so we had to write additional lines to handle both uppercase and lowercase characters properly.

Despite these small difficulties, we managed to fix all the issues through testing, debugging, and research. These experiences helped us improve our problemsolving and development skills.

4.3 Possible Improvements

Although the simulator works well and solves the main problem, there are still a few features that could be added in the future to make it even better:

- Step-by-step explanation: Currently, the tool shows only the final converted result. In the future, we can add a feature that shows how the conversion is done step-by-step, which will help students understand the logic behind the result.
- Arithmetic operations: Features like binary addition, subtraction, multiplication, and division can be added so that users can practice full problems using number systems.
- Bitwise operations: Including operations like AND, OR, NOT, and XOR can help students who are studying digital logic design.
- History and export: Adding a history section where users can see their previous inputs and results or download them as a file could be useful during practice or revision.
- Online hosting: Currently, the tool runs on a local server using Flask. In the future, it can be hosted online so that anyone can access it without needing to run it locally.

4.4 Recommendations

Based on the results and feedback, we recommend that this simulator be used as a learning aid in computer science labs, especially in topics like computer organization, digital electronics, and microprocessors. It can be given to students as a practice tool to build confidence in number system conversions. We also suggest that schools and colleges encourage students to use this tool during online classes or practical sessions.

We also recommend that future students or developers who take up this project expand it further by adding features like step-by-step breakdowns or arithmetic logic. With some more

effort, this simulator can become part of an educational platform or mobile app that helps students beyond just number systems.

In conclusion, the Number System Simulator is a working, reliable, and educational project that can help hundreds of students better understand and learn one of the most basic yet important topics in computer science.

CHAPTER 5

REFLECTION ON LEARNING AND PERSONAL DEVELOPMENT

5.1 Key Learning Outcomes

◈ Academic Knowledge

During the development of this capstone project, a deep understanding of number system and the building blocks of computer architecture was gained. Initially, only simple conversions between binary and decimal were familiar. However, while creating the simulator, all four major number systems—binary, octal, decimal, and hexadecimal—were explored in much greater detail. The mathematical logic behind conversions was studied along with their applications in real-world computing. Insights were also developed into how number systems are involved in memory addressing, programming, and data representation. This project strengthened confidence in core subjects such as digital electronics and computer organization.

◈ Technical Skills

The project provided an excellent opportunity to enhance technical expertise in web development. Clean and organized code was written using HTML, CSS, and JavaScript for the front end, while Flask (Python) was used to run the application locally. JavaScript functions were utilized for logic-based tasks, such as parsing and converting values between different number systems. Skills in if-else conditions, string manipulation, error checking, and data validation were improved. CSS was applied to create a professional, responsive, and well-centered layout. As a result, proficiency in both front-end and back-end development improved significantly.

◈ Problem-Solving and Critical Thinking

Several challenges arose during the project, each requiring logical thinking and creativity to address. For instance, implementing checks to validate user input for a specific base required writing custom conditions to detect invalid characters, demanding patience and detailed analysis. Output formatting also needed to be optimized for clarity and readability. Each challenge offered an opportunity to break problems into smaller tasks and solve them step-by-step, resulting in stronger problem-solving abilities and greater independence in tackling technical issues.

5.2 Challenges Encountered and Overcome

◈ Personal and Professional Growth

At the outset, integrating all components of the project proved difficult. Developing conversion logic in JavaScript was initially challenging, requiring the exploration of different methods and functions, as well as extensive testing and debugging. Creating a layout that worked seamlessly on both desktop and mobile devices also posed difficulties. These issues were resolved through research, tutorials, and seeking guidance from knowledgeable sources. Each resolved challenge increased confidence and reinforced the belief that any problem can be solved with focus and persistence. This mindset contributed to growth both academically and as a future developer.

◈ Collaboration and Communication

While the project was primarily completed independently, discussions with friends and classmates provided valuable feedback that improved the interface and resolved minor bugs. Communicating project ideas and progress to the supervisor helped strengthen the ability to explain technical concepts in simple terms—an essential skill in professional environments. This experience also fostered confidence in the ability to collaborate effectively and share responsibilities when working as part of a team.

5.3 Application of Engineering Standards

Even though the Number System Simulator is an academic-level project, basic engineering standards and good software development practices were followed throughout the process. Applying these standards made the project more professional, reliable, and easier to maintain. It also provided a deeper understanding of how real-world developers write and manage code, especially for web-based applications.

The first engineering standard applied was modular programming. The project code was divided into distinct sections, with each file serving its own purpose. The HTML file was dedicated solely to structuring the webpage, the CSS file handled the design and layout, and the JavaScript file managed all logic The and functionality. This separation of concerns kept the code organized and made it easier to debug or upgrade later—a widely adopted industry practice that improves efficiency and maintainability.

The second standard implemented was **input validation and error handling**. In both engineering and programming, it is crucial to verify user inputs to prevent crashes or incorrect results. JavaScript functions were written to ensure that inputs matched the selected base. For example, in binary mode, only 0s and 1s were accepted. Invalid inputs triggered an error message rather than producing incorrect output. This approach aligns with best practices for safe and user-friendly software.

Another important standard followed was **responsive web design**. CSS code was written to ensure that the simulator's layout adapts to different screen sizes. Whether accessed on a laptop, desktop, or mobile device, the interface automatically adjusts to remain clean and functional. Responsive design is a key modern web development standard, improving accessibility and usability for a wide range of users.

The **user interface (UI)** also adhered to clean and readable design principles. Text was kept large enough for easy reading, buttons were positioned for comfortable interaction, and results were displayed clearly in the output section. Such UI considerations enhance the overall user experience and align with industry practices, where usability is as important as functionality.

Additionally, standard JavaScript methods such as parseInt() and toString() were utilized for conversions. These built-in, well-tested functions ensured accurate and consistent results, reflecting how programming languages handle base conversions in professional development environments.

In conclusion, applying these engineering standards resulted in a more structured, reliable, and user-friendly project. The process reinforced the importance of modularity, validation, responsiveness, and user-focused design—skills that will remain valuable in future projects, internships, and professional software development work.

5.4 Insights into the Industry

While working on the Number System Simulator, many valuable insights were gained into how the software and technology industry operates. Even though this was a small academic project, the development process and challenges provided a clear understanding of the real-world software development cycle. The experience highlighted the practical side of coding—not just writing code, but ensuring it is correct, readable, user-friendly, and maintainable.

In the industry, developers create software for end users, not just for themselves. This project emphasized the importance of thinking from the user's perspective. These are the same

considerations professional developers address when designing user interfaces or web applications. This reinforced the critical role of user experience (UX) in the industry.

The project also demonstrated that coding standards and best practices are essential in professional software development. Separating HTML, CSS, and JavaScript files to keep the code clean follows a widely adopted industry standard. Such practices improve collaboration in team environments, as code written in a consistent format is easier for others to understand and maintain. Additionally, the importance of thorough testing became evident. In the industry, every feature undergoes multiple test cycles with varied inputs. Similarly, this simulator was tested extensively for each base system conversion to ensure accuracy and prevent incorrect results, highlighting the role of quality assurance in real-world projects.

Working with Flask offered valuable exposure to how web applications are hosted and delivered to users. Even though Flask was used here only for local execution, the process provided insight into backend concepts such as routing and deployment. These same principles are applied in companies to build scalable applications that serve users globally.

Another key insight was recognizing the significant impact small tools can have. Many companies, especially within the education technology (EdTech) sector, create simple yet effective tools—such as calculators, converters, or interactive learning modules—that support thousands of learners daily. The Number System Simulator falls into this category, demonstrating that meaningful solutions do not always require large systems. Solving a specific problem efficiently can provide substantial value.

This project also offered a deeper understanding of the structured workflow followed in professional settings. From planning and designing, to developing, testing, and documenting, each stage mirrored the Software Development Life Cycle (SDLC) used in the industry. This alignment with real-world practices fosters confidence in contributing effectively to future internships, collaborative projects, or professional development roles.

In summary, the project provided more than just technical expertise—it offered a clear perspective on how the software industry functions, what matters when building user-focused tools, and how to develop applications using structured, industry-ready practices. These lessons will remain valuable in advancing future academic and professional endeavors.

5.5 Conclusion of Personal Development

Working on the capstone project—**"Number System Simulator"**—had a strong and positive impact on overall personal and professional development. It was more than just a coding or academic task; it was a journey that revealed strengths, improved weaknesses, and demonstrated true capabilities. From concept to completion, every stage of the project brought valuable learning, not only in technical skills but also in planning, problem-solving, discipline, and confidence.

The project demonstrated how knowledge gained in the classroom can be applied in real life. Previously, topics such as number systems or base conversions seemed like purely academic concepts meant for examinations. Through this project, their real-world applications in computer systems, software, and electronics became evident. Developing a functional tool for number system conversions provided a clear example of how basic academic concepts can be transformed into practical tools that benefit many people. This realization has fostered a deeper curiosity for exploring real-world applications of academic knowledge.

Technical skills also improved significantly. The project began with only a basic understanding of HTML, CSS, and JavaScript, but gradually expanded into advanced areas such as responsive design, DOM manipulation, and error handling. Hands-on experience with Flask introduced backend development using Python, broadening the scope toward full-stack capabilities. These skills are highly relevant in the software and IT industry, contributing to readiness for roles in web development, application development, and full-stack projects.

Beyond technical growth, the project encouraged greater self-reliance and patience. Challenges arose when certain features did not work as expected, leading to extended periods of debugging and testing. Instead of halting progress, research and repeated trials led to successful solutions. This process instilled perseverance—an essential quality in both academics and professional life—and improved time management skills through structured work planning to meet deadlines.

The experience also helped refine career goals. Interest in software development, user interface design, and educational technology was strengthened by the satisfaction of building a useful, working tool. This reinforced the desire to work on projects that address real-world problems for students, educators, and communities.

Finally, this capstone project provided valuable exposure to the complete software development process. From initiating an idea, breaking it down into manageable tasks, researching relevant technologies, building and testing the system, to presenting the final output with documentation—each step closely mirrored professional workflows. Such end-to-end project experience is highly valued by employers and serves as strong preparation for future internships and career opportunities.

In summary, the project contributed to growth as a learner, developer, and professional. It strengthened skills, shaped the right mindset, and built the confidence to take on future challenges while laying a solid foundation for a successful career.

CHAPTER 6

CONCLUSION

This capstone project, titled “Number System Simulator,” was developed to address a common academic challenge faced by students learning computer science and digital electronics—understanding and converting between different number systems such as binary, octal, decimal, and hexadecimal. The project began with a simple question: How can we make number system conversions easier, faster, and more understandable for students? After identifying the problem and exploring the difficulties students face when learning conversions manually, the solution was designed as a simple, web-based tool that can instantly convert values between these number systems.

Through the use of HTML and CSS for design, JavaScript for logic, and Flask (Python) for hosting, a lightweight, responsive, and user-friendly application was created. This simulator takes a number and its base as input, validates it, and accurately converts it into the other three formats in real-time. It also provides clear error messages for invalid inputs, making the tool not only functional but also educational and safe for beginners.

The key findings of this project include the fact that many students struggle with conversions not because they lack intelligence, but because they lack the proper tools and instant feedback to learn from their mistakes. The Number System Simulator solves this issue by giving them a space where they can learn, test, and improve without pressure. It also saves time, reduces human error, and builds confidence among users.

The impact of this tool is small but meaningful. It supports students in building a stronger foundation in computer science by helping them understand a core concept in a more interactive way. It can be used by teachers during demonstrations or as a practice tool by learners at any level. This project proves that even simple digital tools, when designed thoughtfully, can have a big impact in improving the learning experience.

In conclusion, the Number System Simulator project is valuable not only as a software product but also as a learning journey. It demonstrates how technology can be used to solve everyday educational problems. The project’s simplicity, usefulness, and ability to support academic growth make it a significant contribution to the field of computer science education. It is a tool with real-world relevance, and it also serves as a strong personal milestone in the developer’s journey toward a future in technology and software development.

REFERENCES

[1] M. Fasi and M. Mikaitis, “CPFloat: A C library for simulating low-precision arithmetic,” *ACM Transactions on Mathematical Software*, vol. 49, no. 2, Art. 18, pp. 1–32, Jun. 2023, doi: 10.1145/358

[2] I. Cohen and G. Einziger, “Floating-Floating-Point: A highly accurate number representation with flexible counting ranges,” *arXiv*, Sep. 22, 2024. [Online]. Available: arXiv:2410.03692

[3] J. Y. Ma, L. Z. Li, X. M. Liu, and Z. X. Wang, “Conversion between Number Systems in Membrane Computing,” *Applied Sciences*, vol. 13, no. 17, Art. 9945, Aug. 2023, doi: 10.3390/app13179945.

[4] A. de Sousa, I. dos Santos, and R. de Carvalho, “Embedding an Electrical System Real-Time Simulator with Floating-Point Arithmetic in FPGA,” *Energies*, vol. 14, no. 24, Art. 8404, Dec. 2021, doi: 10.3390/en14248404.

[5] N. Jovanović, D. Marković, D. Živković, and R. Popović, “SIMAS: A Web-Based Computer System Simulator,” *International Journal of Electrical Engineering*, vol. 26, no. 4, pp. 931–941, 2019.

CHAPTER 7

Appendices:

Appendix A: HTML Code (index.html):

```
<!DOCTYPE html>
<html>
<head>
  <title>Number System Simulator</title>
  <link rel="stylesheet" href="style.css">
</head>
<body>
  <h1>Number System Simulator</h1>
  <input type="text" id="numberInput" placeholder="Enter a number">
  <select id="inputBase">
    <option value="2">Binary</option>
    <option value="8">Octal</option>
    <option value="10" selected>Decimal</option>
    <option value="16">Hexadecimal</option>
  </select>
  <button onclick="convert()">Convert</button>
  <div id="result"></div>
  <script src="script.js"></script>
</body>
</html>
```

Appendix B: CSS Code (style.css):

```
body {
  margin: 0;  padding: 0;
display: flex;  flex-direction:
column;  font-family: Arial;
text-align: center;  padding:
30px;  align-items: center;
justify-content: center;  height:
100vh;  color: white;
background-color: #1e1e1e;
}
input, select, button
{  padding: 10px;
font-size: 16px;  margin:
5px;  width: 30%;
}
#result {  margin-top: 20px;
font-size: 20px;  color: #333;
background-color: white;
border-radius: 15px;  padding:
20px;  border: 2px solid
black;
}
```

Appendix C: JavaScript Code (script.js):

```
function convert() {    const numStr =
document.getElementById("numberInput").value.trim();    const base =
parseInt(document.getElementById("inputBase").value);    let num;    try {      num =
parseInt(numStr, base);      if (isNaN(num)) throw "Invalid number";      const binary
= num.toString(2);      const octal = num.toString(8);      const decimal =
num.toString(10);      const hex = num.toString(16).toUpperCase();
document.getElementById("result").style = 'display: block;';
document.getElementById("result").innerHTML = `
     <strong>Binary:</strong> ${binary}<br>
     <strong>Octal:</strong> ${octal}<br>
     <strong>Decimal:</strong> ${decimal}<br>
     <strong>Hexadecimal:</strong> ${hex}
   `;
 } catch (e) {
         document.getElementById("result").innerHTML      =          "<span
style='color:red;'>Invalid input!</span>";
 }
}
```

Appendix D: Python Flask Code (app.py):

```
from flask
import Flask, render_template

app = Flask(__name__)

@app.route('/')       def
home():
   return render_template('index.html')  # HTML file should be inside a 'templates'
folder

if __name__ == '__main__':
   app.run(debug=True)
```

Appendix E: Flow of Operation

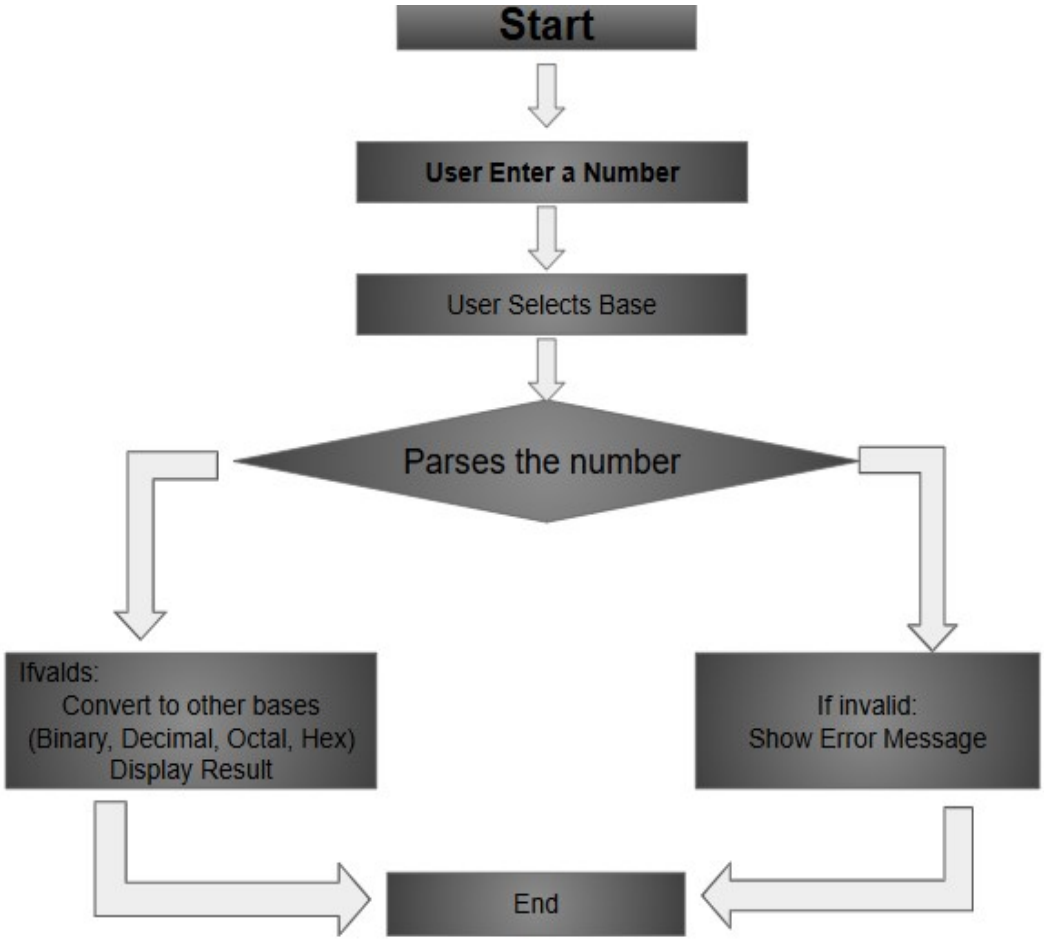


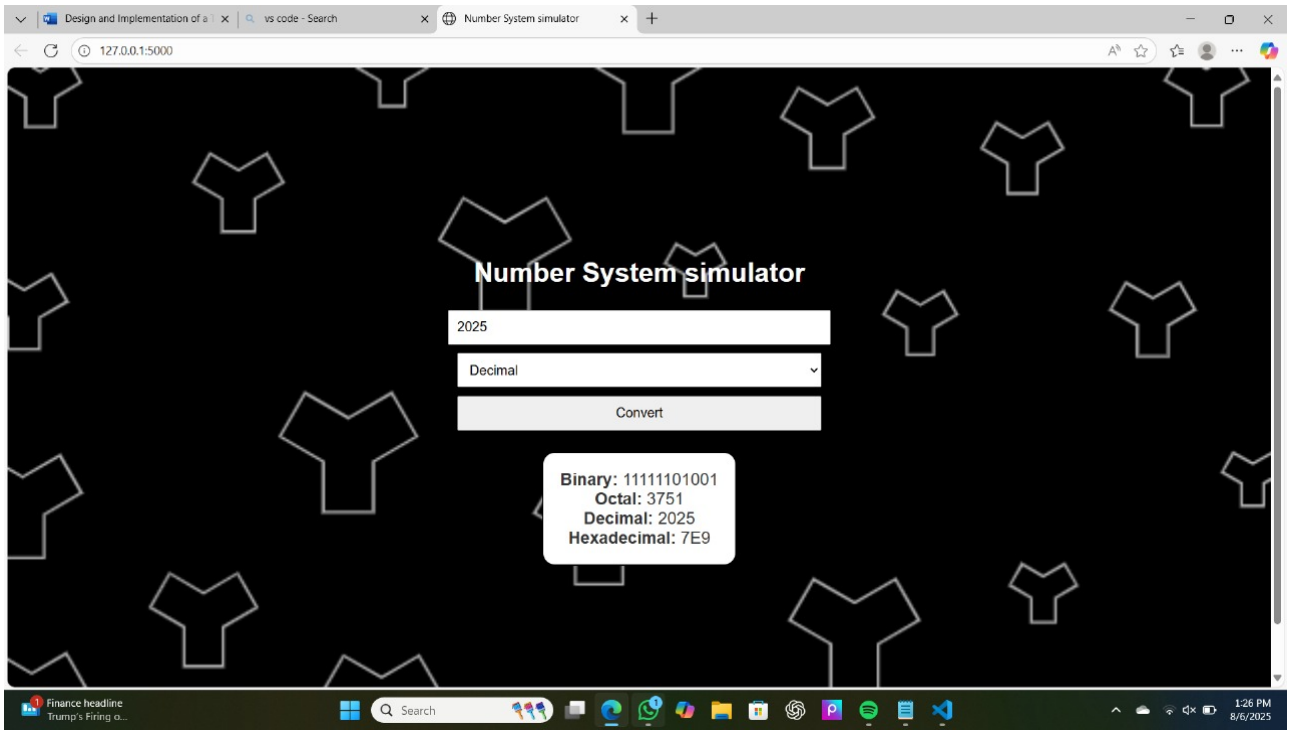
Figure 1: Flow diagram of Number System Simulator

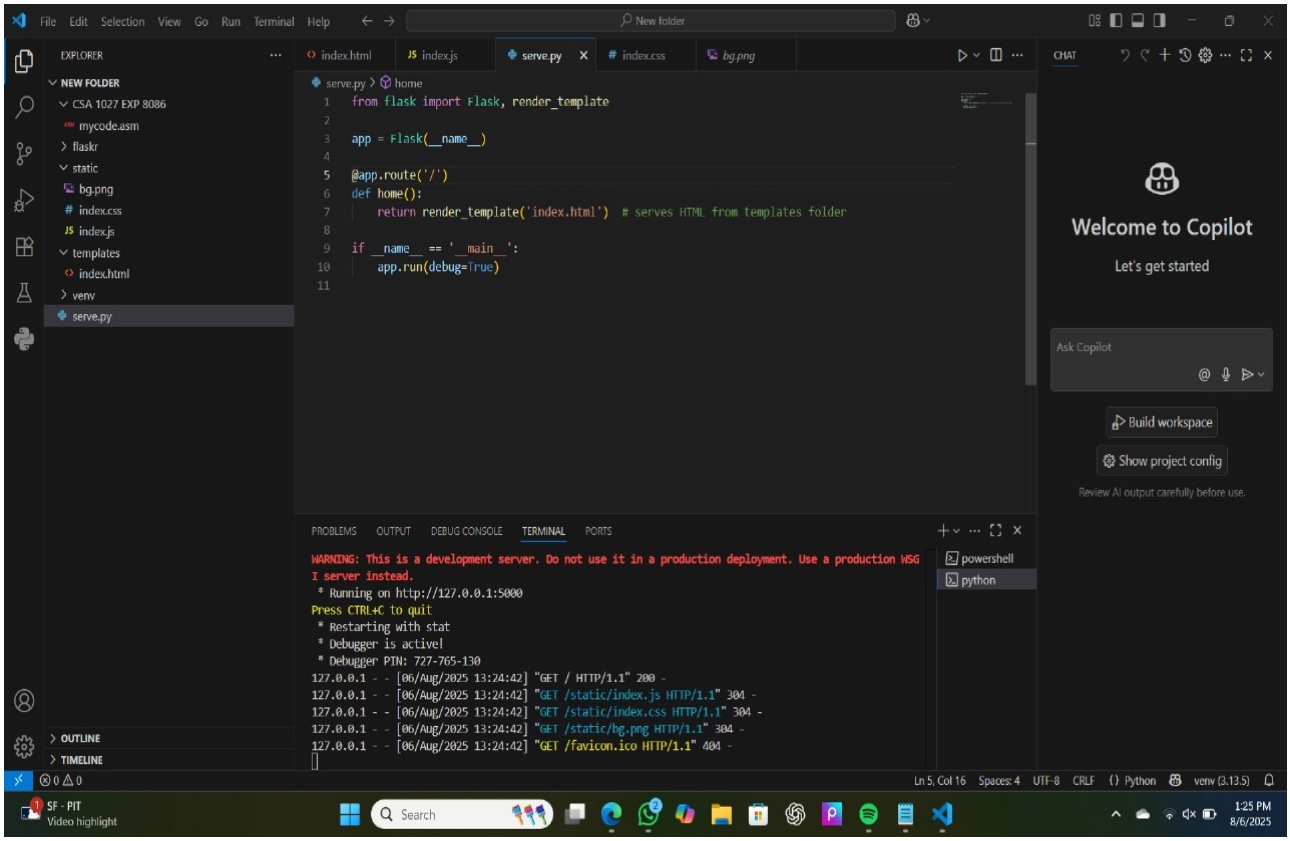
Appendix F: User Manual

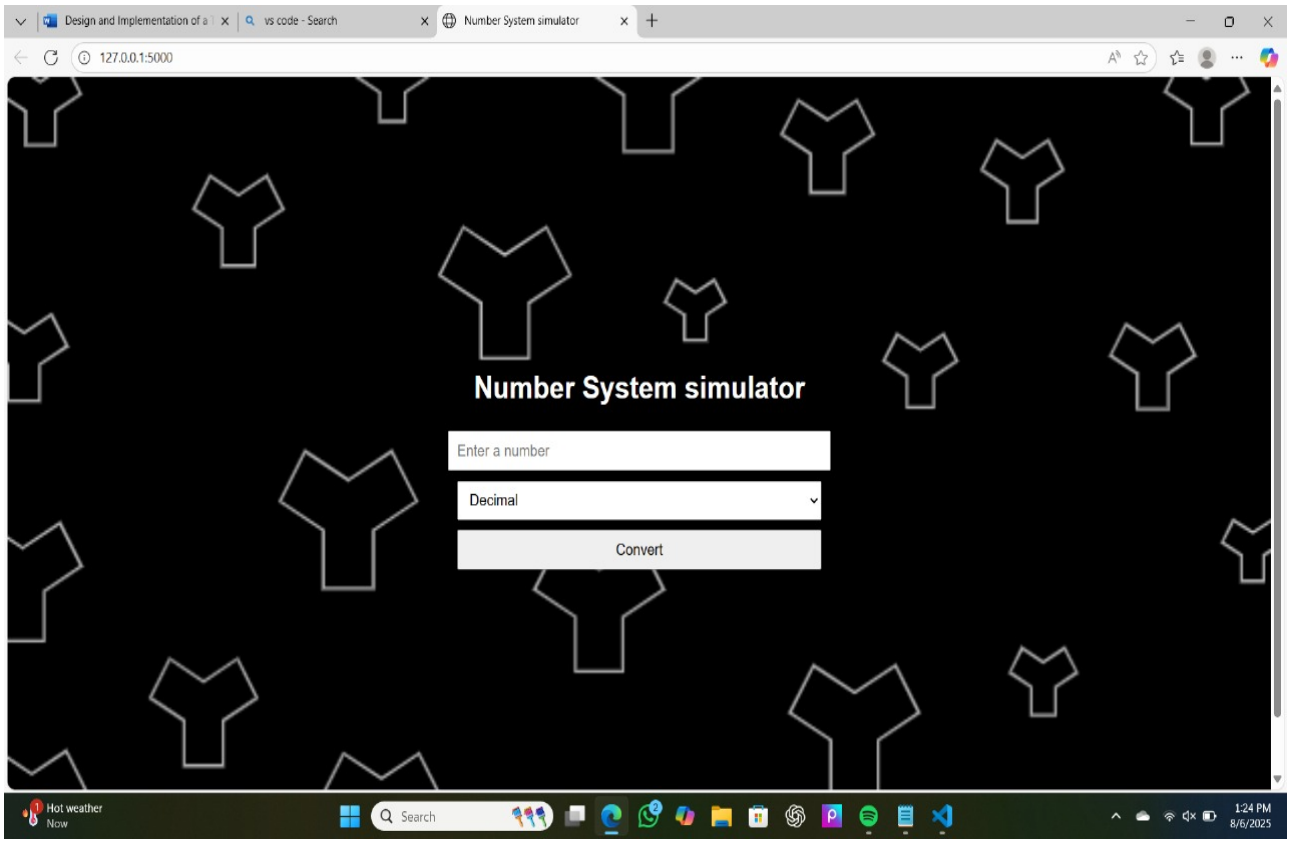
Steps to Use the Simulator:

1. Open the project folder and run app.py using Python.
2. The Flask server will start, and the browser will open at http://127.0.0.1:5000/.
3. Enter any number in the input field (e.g., 1010).
4. Select the base of the number (e.g., Binary).
5. Click the Convert button.
6. Instantly view the result in Binary, Decimal, Octal, and Hexadecimal formats.
7. If input is invalid, an error message will appear.

Appendix G: Number System Simulator






Design and Implementation of a
vs code - Search
Number System simulator
127.0.0.1:5000
Number System simulator
Enter a number
Decimal
Convert
Hot weather
Now
Search
1:24 PM
8/6/2025