HCI & Computer Graphics LAB Report

Name : Afaq Ahmed Sap id : 55241

Table of Contents:

Lab 1

Activity:

Task 1:

Introduction to HCI

1. Discussion: - Define HCI and its importance in designing user-friendly systems.
    - Discuss the key components of HCI: User, Computer, and Interaction.
    - Explore examples of good and bad HCI designs (e.g., websites, mobile apps).
2. Discover list of wireframing tools (e.g., Figma or Adobe XD)

What is HCI and Its Importance

HCI (Human-Computer Interaction) is the study and design of how people interact with computers and digital systems. It focuses on creating systems that are efficient, intuitive, and enjoyable for users to interact with.
HCI is important because:

- It ensures **user-centered design** for better usability.
- Reduces user errors and enhances productivity.
- Increases accessibility and user satisfaction.
- Helps bridge the gap between technology and real-world user needs.

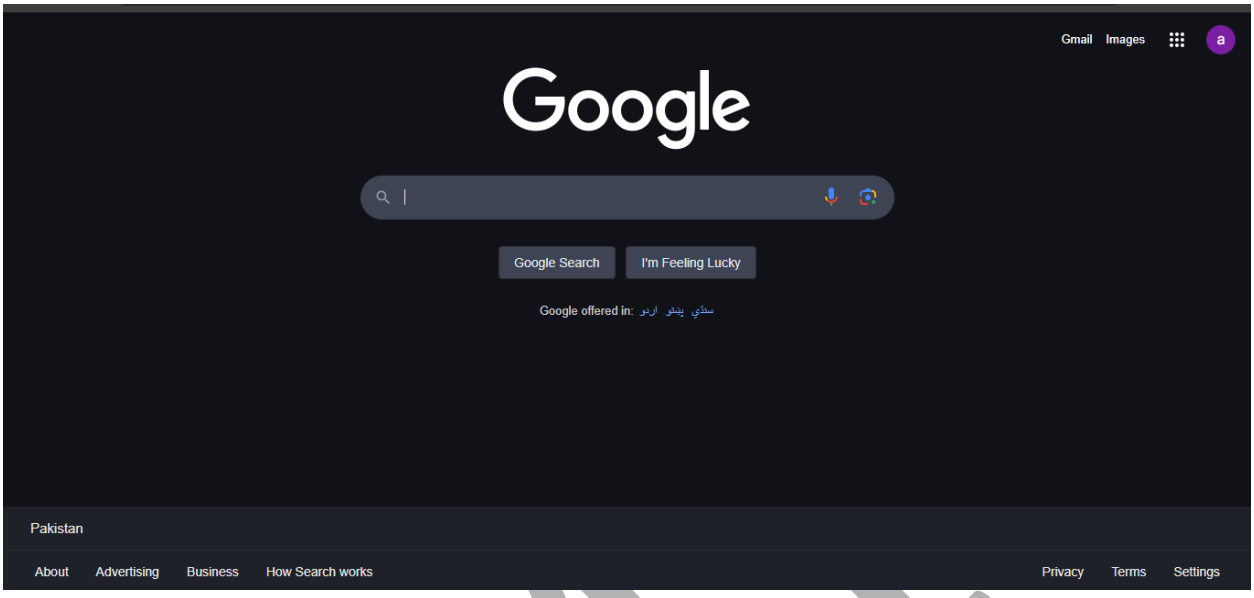
Key Components of HCI

1. **User**
    - The individual or group interacting with the computer.
    - Users may have different skill levels, needs, and goals.
2. **Computer**
    - Any digital system or device (e.g., laptops, smartphones, ATMs) that the user interacts with.
3. **Interaction**
    - The communication between the user and the computer, including inputs (clicks, gestures, voice) and outputs (visuals, audio, feedback).

Examples of Good and Bad HCI Designs

Good HCI Designs

- **Google Search Engine** – Simple, fast, and minimal design.



- **Apple iOS Interface** – Intuitive navigation, consistent icons, and good feedback.

2. Discover: List of Wireframing Tools

Tool	Description
Figma	Cloud-based tool with real-time collaboration, great for UI/UX design.
Adobe XD	Powerful design and prototyping tool with integration into Adobe Creative Suite.
Balsamiq	Easy-to-use, low-fidelity wireframing tool, ideal for quick sketches.
Sketch	Popular with macOS users, great for interface and icon design.
Axure RP	Advanced wireframing and prototyping, with interaction and logic features.
InVision	Focuses on prototyping and collaboration with interactive mockups.
Marvel	Simple wireframing and prototyping tool, beginner-friendly.
Wireframe.cc	A minimal and straightforward wireframing tool for quick drafts.

Task 2:

Integration of HCI and CG Discussion: -

- Discuss how HCI and CG work together in applications like virtual reality, augmented reality, and video games.
- Explore the role of user interaction in CG applications.

1. How HCI and CG Work Together

HCI (Human-Computer Interaction) and CG (Computer Graphics) are closely linked in modern digital applications, especially in Virtual Reality (VR), Augmented Reality (AR), and Video Games.

Virtual Reality (VR)

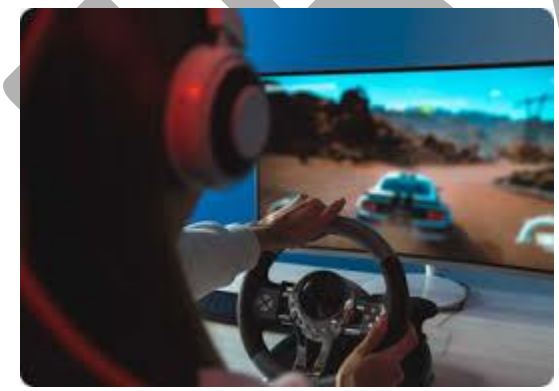
- HCI focuses on how users navigate and interact with virtual environments.
- CG generates realistic 3D graphics to simulate immersive worlds.
- Example: VR games like *Beat Saber* use intuitive hand controls (HCI) and immersive visuals (CG) to engage players.

Augmented Reality (AR)

- HCI helps design user interfaces and interactions with real-world and virtual overlays.
- CG is used to render virtual objects that blend with the real environment.
- Example: *Pokémon GO* uses HCI to interact with the game via touch and movement, and CG to place Pokémon in the real world.

Video Games

- HCI plays a role in designing game controls, menus, and feedback systems.
- CG is responsible for creating the visual world, characters, animations, and effects.
- Example: In a racing game, HCI ensures the player has intuitive steering, while CG provides lifelike tracks and cars.



2. Role of User Interaction in CG Applications

User interaction is essential in CG-based systems, as it determines how effectively a user can control, manipulate, and respond to the visual environment.

Key Roles:

- Navigation: Moving through 3D environments using input devices like keyboards, controllers, or VR handsets.

- Manipulation: Interacting with objects—rotating, resizing, or dragging in 3D space.
- Feedback: Receiving visual/auditory responses (e.g., button animation, sound effects) that confirm actions.
- Realism & Immersion: Good interaction design makes CG environments feel more lifelike and responsive.

Examples:

- 3D modeling software like Blender uses HCI principles for intuitive tool placement and manipulation.

- Medical simulations use HCI+CG for training, allowing users to interact with virtual organs or tools.

- Flight simulators combine physical controls (HCI) with realistic 3D visuals (CG) for training pilots.

Post-Lab Questions

1. What are the key differences between HCI and CG? How do they complement each other?

HCI (Human-Computer Interaction)	**CG (Computer Graphics)**
Focuses on the **design and usability** of user interfaces	Focuses on **creating and rendering visuals**
Involves user behavior, psychology, ergonomics	Involves math, physics, and algorithms
Examples: UI design, input systems, feedback	Examples: 3D modeling, animation, shading

How They Complement Each Other:

- CG provides the visual elements, while HCI ensures users can interact with them effectively.
- In VR/AR, HCI makes the controls intuitive and immersive, while CG brings the virtual world to life.

2. Why is usability important in HCI? Provide examples of good and bad usability.

Usability ensures that users can achieve their goals efficiently, effectively, and satisfactorily.

- Good Usability
  Example:
  Google homepage: Simple, fast, minimal distractions.
- Bad Usability
  Example:
  Complex software with too many hidden menus and inconsistent navigation, like old-school enterprise applications.
  Usability directly impacts user satisfaction, productivity, and adoption of a system.

3. What are the challenges of creating realistic computer graphics? How do these challenges impact user experience?

Challenges:

- Achieving photorealism requires heavy computation and rendering power.
- Lighting, shadows, textures, and physics must mimic the real world.
- Latency in real-time rendering can break immersion.
- Requires specialized hardware and software.

Impact on User Experience:

- Poor graphics reduce realism and engagement.
- High-quality CG enhances immersion, believability, and overall user enjoyment in games, training simulations, and VR/AR.

4. Reflect on your experience with the tools used in the lab. What did you find easy or difficult?

   Tool Used: Figma

   Easy Part: Drag-and-drop interface, ready-made components made wireframing quick and simple.

   Difficult Part: Learning how to prototype interactions and manage layers effectively.

   Overall, the tool was user-friendly, but mastering all features would require more practice.

Lab Report Guidelines

1. Purpose of the Lab and Concepts Covered

The purpose of the lab was to introduce the concepts of Human-Computer Interaction (HCI) and its integration with Computer Graphics (CG).

We explored:

- Importance of usability and user-centered design.
- The relationship between HCI and CG in real-world applications (VR, AR, gaming).
- Hands-on use of wireframing tools (e.g., Figma or Adobe XD).

2. Key Takeaways

- HCI and CG must work hand-in-hand for creating immersive and interactive experiences.
- Usability is a critical factor in user satisfaction and system success.
- Wireframing tools like Figma simplify the UI/UX design process, allowing for rapid prototyping and collaboration.
- Realistic CG requires complex processes but can transform the way users interact with digital systems.

Lab 2

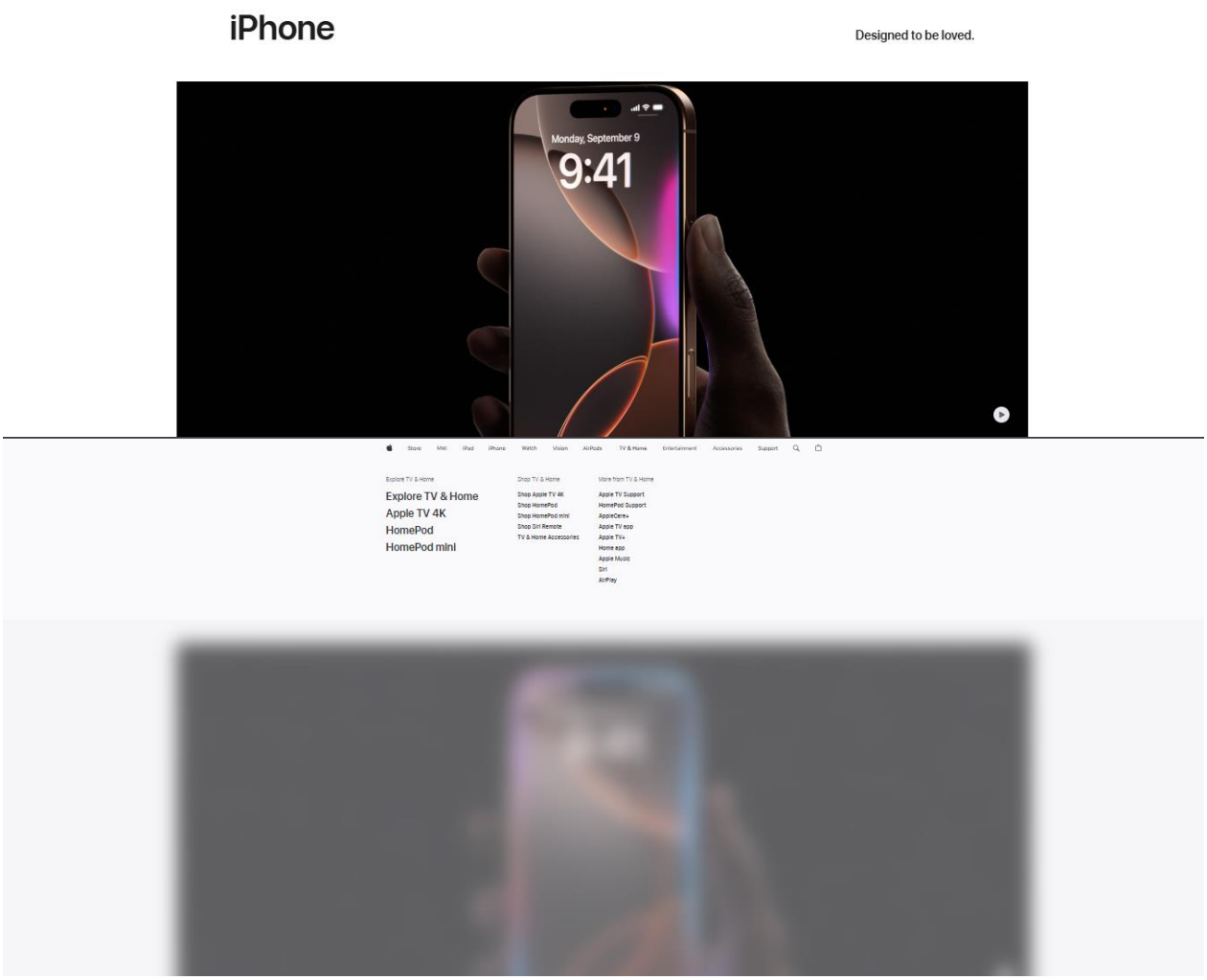
Activity:

1: Observing Real-World Designs

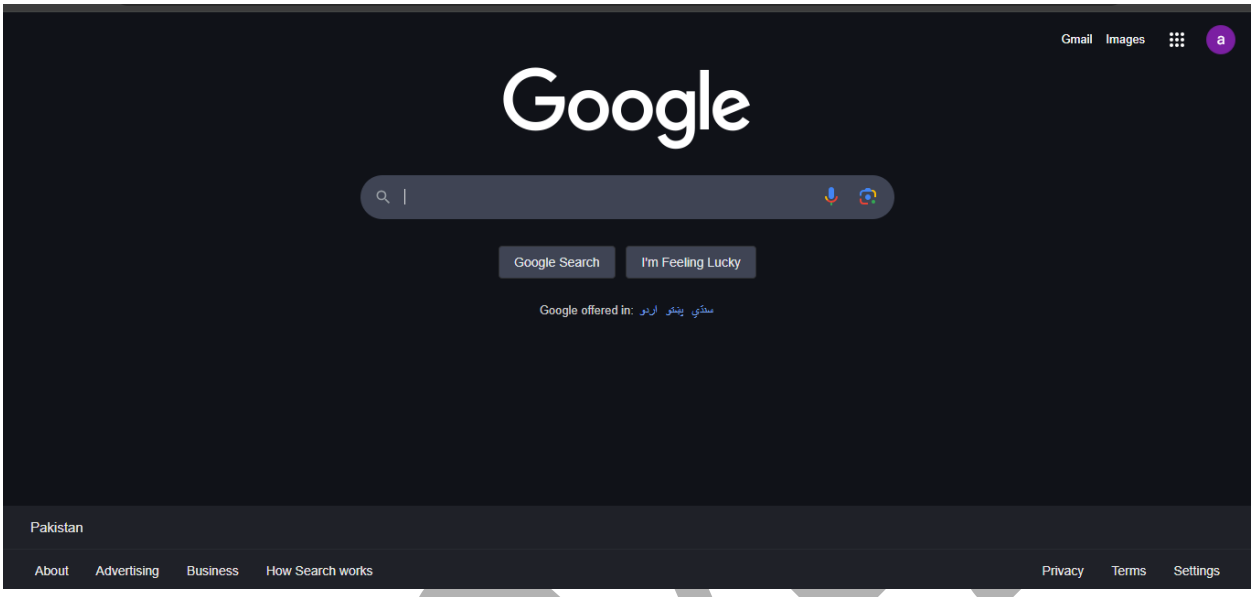
1. Task: Observe and document examples of good and bad designs in your environment (e.g., websites, mobile apps, physical devices, public interfaces like ATMs or ticket machines).

- Good Design

1 Apple's iPhone home screen (simple, consistent, intuitive).

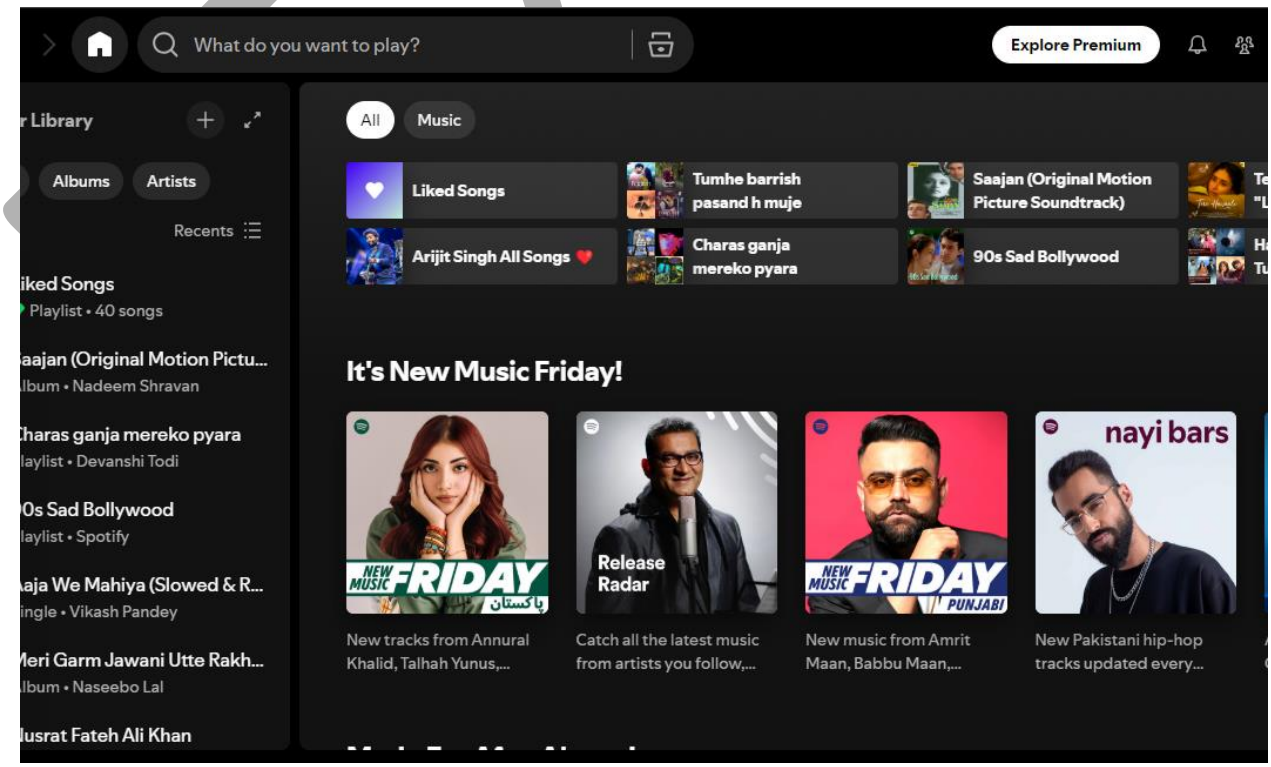


2 Google Search (minimalist, clear feedback, fast results).



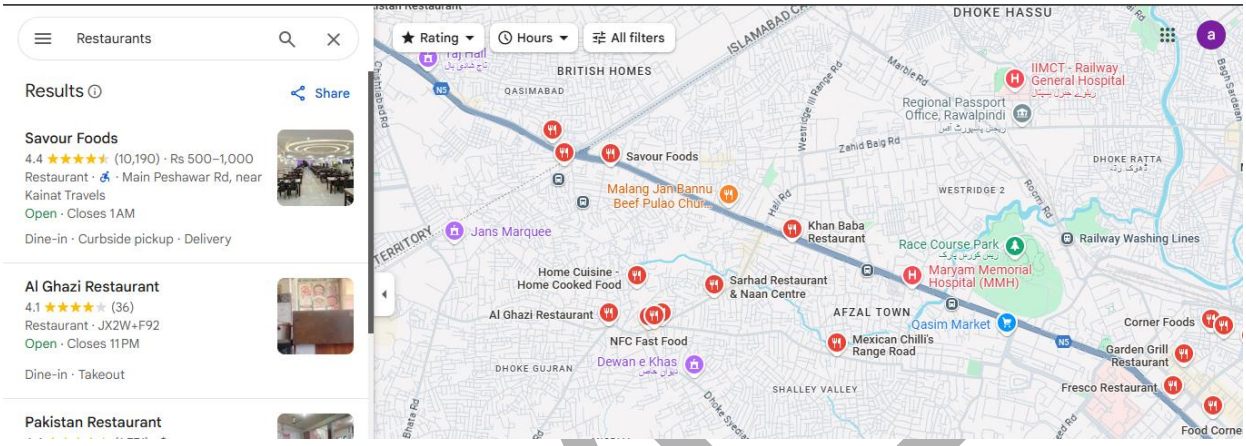
3 **Spotify Mobile App**

- Why it's good: clean interface, easy navigation, smart use of icons and text. The search and recommendations are personalized and easily accessible



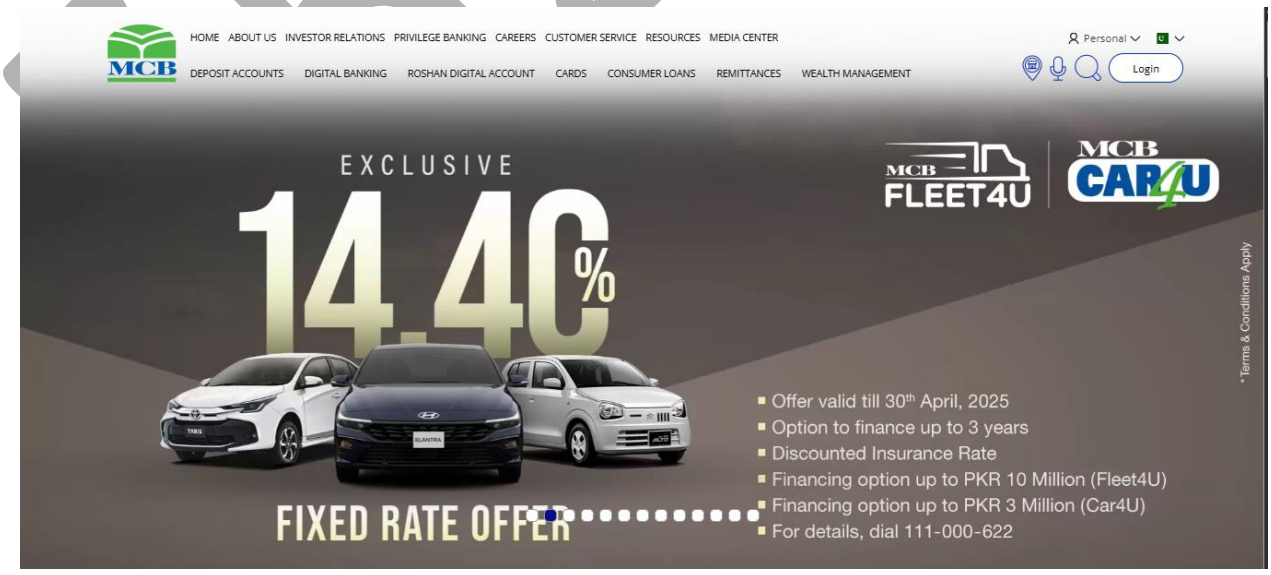
4 **Google Maps**

- Why it's good: Provides clear directions, real-time traffic updates, and intuitive user interaction. The interface is consistent across devices.



5 **ATM Interface by MCB Bank (Pakistan)**

- Why it's good: Easy language selection, large buttons, simple instructions, and fast transaction flow.

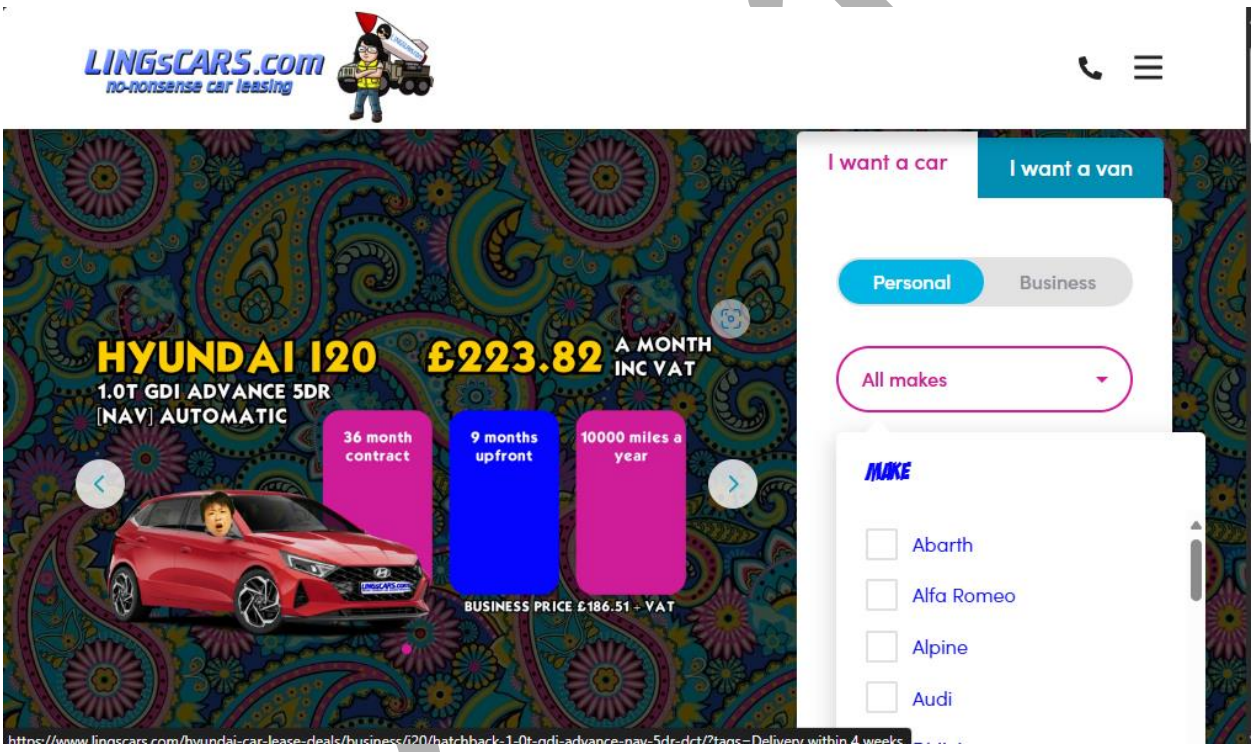


- Bad Design

1. LingsCars.com

Why it's bad:

- Extremely cluttered and chaotic design.
- Overuse of colors, fonts, and animations.
- Hard to focus or find important information.
- Sensory overload—poor visual hierarchy.



2. Arngren.net

Why it's bad:

- No layout or organization; everything is just dumped on the page.
- No clear navigation.
- Overloaded with images and text with no spacing or alignment.
- Confusing for any first-time user.

3. Yale School of Art (Old Version)

Why it's bad:

- Unconventional design that breaks standard usability principles.
- Hard-to-read fonts, inconsistent layout.
- Too experimental for general users—confusing UX.

Activity 2: Analyzing Website Usability

1 Websites Chosen:

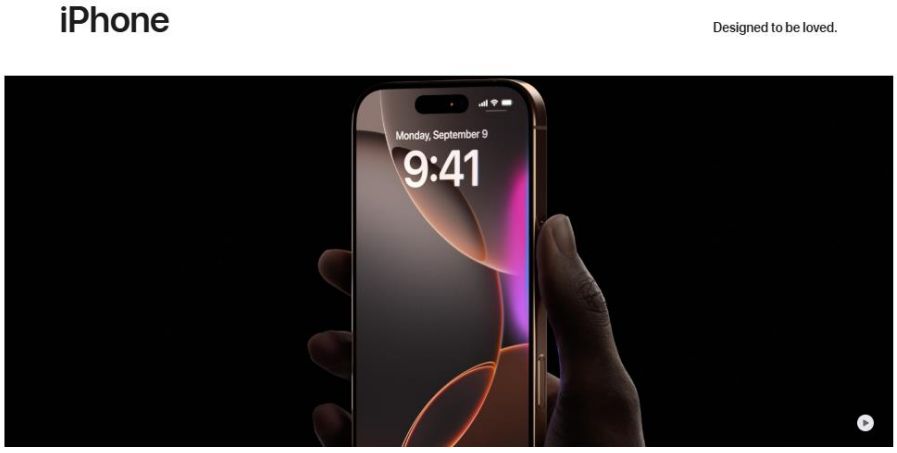
- Good Design: Apple clean layout, clear navigation, responsive design.
- Bad Design: Arngren cluttered layout, inconsistent design, poor readability.

2 **Eva**luation Criteria (Based on HCI Principles):

Principle	**Apple Website**	**Arngren Website**
1. Visibility	Clear menus, icons, product images	Cluttered screen, no visual hierarchy
2. Feedback	Instant hover effects, smooth animations	No user feedback, everything static
3. Consistency	Uniform colors, fonts, layouts	Inconsistent layout, font sizes, colors
4. Affordance	Buttons and links look clickable	No visual indication for clickable items
5. Learnability	Intuitive navigation, minimal learning curve	Extremely hard to understand
6. Accessibility	Responsive and mobile-friendly	Not responsive; bad readability

3. Apple Website – Good Design

- Homepage Screenshot:



Strengths:

- Clean and modern layout with high-quality visuals.
- Clearly labeled sections like "Mac", "iPhone", "Support".
- Smooth animations give feedback to user actions.

- Excellent use of whitespace, making content easy to scan.
- Mobile responsive and fast loading.

4. Arngren Website – Bad Design

Weaknesses:

- No clear structure – everything is placed randomly on one page.
- Inconsistent and outdated font styles and colors.
- No navigation menu or search bar.
- Extremely poor readability and cluttered layout.
- Not mobile-friendly, not accessible for users with disabilities.

5. Conclusion

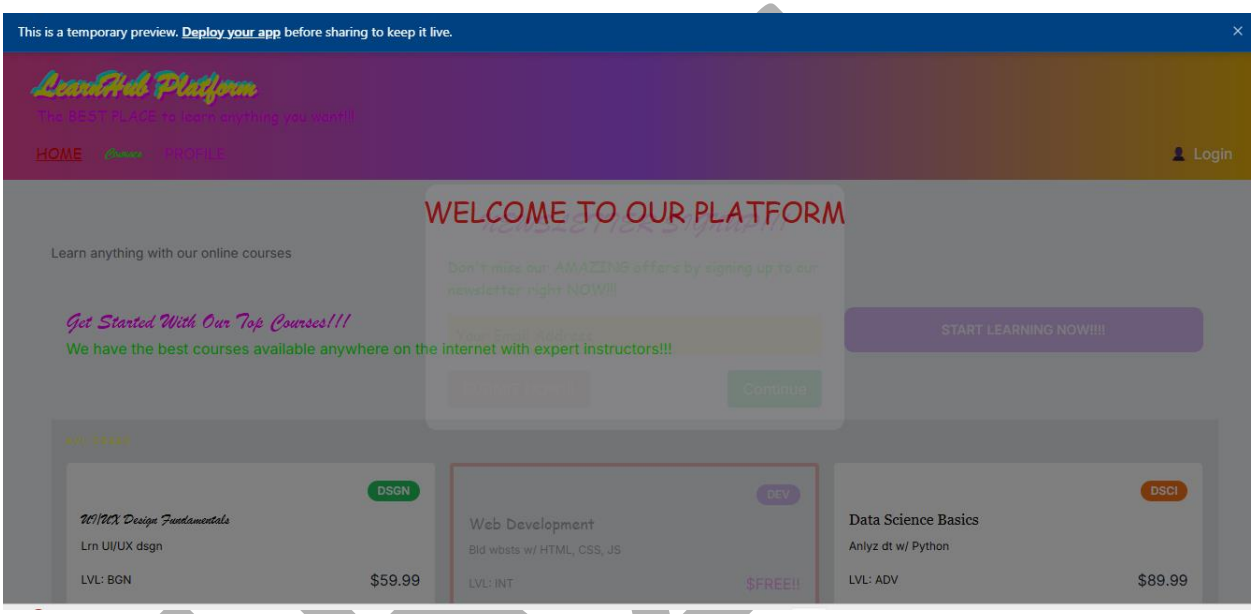
- The **Apple website** is an excellent example of applying HCI principles effectively. It offers a great user experience, is easy to use, visually appealing, and responsive.
  In contrast, the **Arngren website** fails in nearly every aspect of HCI—it's difficult to navigate, confusing, and not visually pleasing. This comparison clearly shows the importance of good interface design in enhancing user satisfaction and usability.

Activity 3: Redesign Exercise

1. Task: Identify a poorly designed interface (e.g., a confusing mobile app, a cluttered website, or a physical device) and propose a redesign.

Learn Hub

2. Steps: - Take a screenshot or photo of the bad design

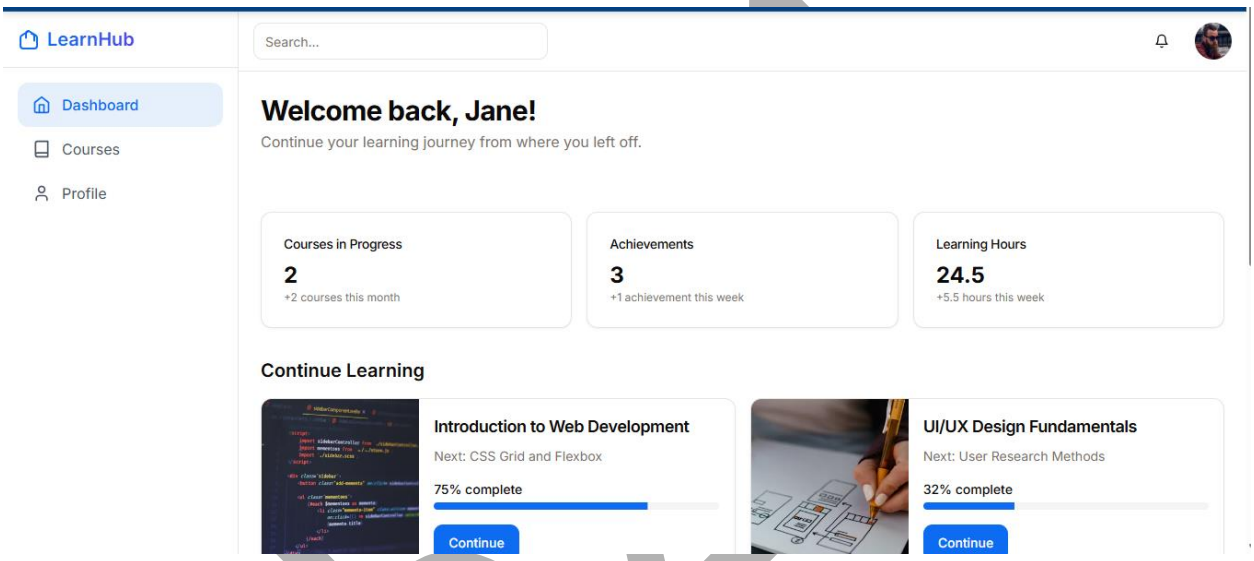


- Identify the specific HCI principles it violates

The application already has a comprehensive set of HCI principles that it violates. These are documented in the HCI Violations and include:

- Visual Hierarchy Violation - The header uses neon colors and poor text alignment, making it difficult to identify important information.
- Consistency Principal Violation - Different fonts, colors, and styles are used inconsistently throughout the interface, especially in form elements.
- Visibility & Feedback Violation - The slow input field doesn't provide immediate feedback, and hidden interactive elements don't indicate they're clickable.
- Affordance Violation - Items that look clickable aren't, while actual buttons are visually de-emphasized. The submit button moves away when hovered.
- Cognitive Load Violation - The interface forces users to remember information rather than recognize it, requiring mental translation.
- Accessibility Violation - Poor color contrast makes text difficult to read, and small interactive elements are hard to click.

- User Control Violation - The modal appears without user action and has a tiny dismiss button. The reset button is misleadingly labeled as 'continue'.
- Recognition vs. Recall Violation - Form labels and fields are mismatched in size and position, and the table hierarchy is confusing.
- Error Prevention Violation - No form validation is present, and the moving submit button makes errors more likely.
- Flexibility & Efficiency Violation - No shortcuts are provided, and the interface forces unnecessary steps.



The new Learn Hub educational platform design significantly improves usability by adhering to core Human-Computer Interaction (HCI) principles:

1. Visibility of System Status

- **Dashboard metrics**: Users can clearly see their progress (courses completed, enrolled courses, learning hours)
- **Course progress bars**: Visual indicators show completion percentage for each course
- **Active navigation items**: The current section is highlighted in the sidebar

2. Match Between System and Real World

- **Intuitive categorization**: Courses are organized by familiar categories (Development, Design, Data Science)
- **Realistic learning timeline**: Course durations and lesson structures match real-world educational experiences

- **Common educational terminology**: Using language like "enroll," "certificate," and "instructor"

3. User Control and Freedom

- **Clear navigation options**: Sidebar menu allows users to move between sections easily
- **Filter and search functionality**: Users can narrow down course listings by category, level, or search term
- **Clear cancel buttons**: Ability to exit modal dialogs and clear form inputs

4. Consistency and Standards

- **Unified design language**: Consistent color scheme, typography, and component usage throughout the application
- **Predictable interactions**: Elements that look clickable are clickable
- **Standard UI patterns**: Using familiar tab interfaces, cards, and navigation patterns

5. Error Prevention

- **Form validation**: Proper input validation in forms prevents errors before submission
- **Visible system status**: Users are informed about the state of their actions
- **Confirmation messages**: Important actions require confirmation

6. Recognition Rather Than Recall

- **Visual course cards**: Course information presented with images and clear labels
- **Badge indicators**: Visual cues for course categories, featured status, and completion
- **Profile dashboard**: User progress and statistics readily visible

7. Flexibility and Efficiency of Use

- **Responsive design**: Adapts to different screen sizes (mobile, tablet, desktop)
- **Efficient navigation**: Quick access to important sections
- **Search functionality**: Fast way to find specific content

8. Aesthetic and Minimalist Design

- **Clean interface**: Focused content with proper whitespace and visual hierarchy
- **Card-based layouts**: Information presented in digestible chunks
- **Gradient accents**: Subtle visual interest without overwhelming the user

9. Help Users Recognize, Diagnose, and Recover from Errors

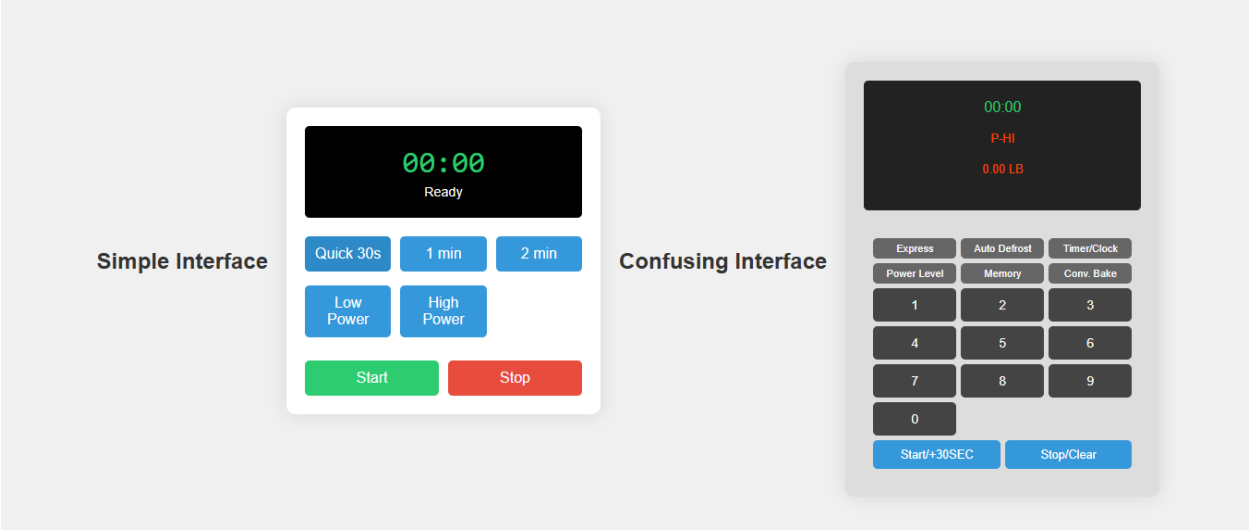
- **Clear error states**: Explicit messages when searches yield no results
- **Empty states**: Appropriate messaging when lists are empty

- **Loading indicators**: Visual feedback during data loading

10. Accessibility Considerations

- **Proper contrast ratios**: Text is legible against backgrounds
- **Keyboard navigation**: Interactive elements can be accessed via keyboard
- **Screen reader support**: Semantic markup and ARIA attributes for assistive technologies
- **Clear focus states**: Visible indicators when elements are focused

3. Example: Redesign a confusing microwave interface by simplifying the buttons and adding clear labels.



Activity 4: Group Discussion and Presentation

Based on the microwave interface example we've created; I can explain the key points about interface design:

1. Common mistakes in bad designs:

- Overwhelming number of buttons (like in our "confusing" interface)
- Unclear labels and cryptic feedback (e.g., "P-HI" instead of "High Power")
- Inconsistent behavior (random error messages)
- Poor organization (scattered controls without logical grouping)
- Hidden features requiring memorization of sequences

2. Good design improves satisfaction and efficiency through:

- Clear, descriptive labels (like "Quick 30s", "1 min" in the simple interface)
- Logical grouping (presets, power controls, main controls)

- Immediate feedback (status display showing "Ready", "Cooking...")
- Most common functions easily accessible (quick time presets)
- Consistent behavior and visual hierarchy

3. Trade-offs in simple vs. feature-rich interfaces:

- Simplicity vs. Functionality: The simple interface sacrifices advanced features (like defrost by weight) for clarity
- Learning curve vs. Power: The complex interface offers more features but requires learning
- Speed vs. Precision: Quick presets are faster but less precise than manual time entry
- Space vs. Features: More features require more space, making the interface busier
- Accessibility vs. Complexity: Simpler interfaces are more accessible but may limit advanced users

The example demonstrates these principles by contrasting:

- Simple interface: 7 clear buttons, logical groups, immediate feedback
- Confusing interface: 16+ buttons, cryptic labels, unclear feedback, requiring memorization

This shows how good design focuses on common use cases while bad design tries to expose every possible feature at once.

Lab 3:

1. Introduction to Interaction Styles Interaction styles refer to the ways users interact with a system. The main interaction styles in HCI are:

- Command-Line Interface (CLI)
- Menu-Driven Interface
- Form-Based Interface
- Graphical User Interface (GUI)
- Natural Language Interface
- Designing WIMP and Direct Manipulation Interfaces

2. Command-Line Interface (CLI)

Overview:

- Users interact with the system by typing commands.
- Requires memorization of commands and syntax.
- Efficient for advanced users but not user-friendly for beginners.

A simple CLI calculator in Python.

Exercise:

- Add more operations (e.g., exponentiation, modulus).

Solution:

```
def factorial(n):
    if n < 0:
        return "Error: Factorial not defined for negative numbers"
    if n == 0 or n == 1:
        return 1
    return n * factorial(n - 1)


def cli_calculator():
    print("Welcome to the CLI Calculator!")
    while True:
        print("\nAvailable operations:")
        print("add, subtract, multiply, divide")
        print("power, modulus, sqrt, factorial, quit")
```

```
command = input("Enter a command: ").strip().lower()

if command == "quit":
    print("Exiting the calculator. Goodbye!")
    break

try:
    if command == "sqrt":
        num = float(input("Enter a number: "))
        if num < 0:
            print("Error: Cannot calculate square root of negative number")
        else:
            print(f"Result: {num ** 0.5}")
        continue

    if command == "factorial":
        num = int(input("Enter a number: "))
        result = factorial(num)
        print(f"Result: {result}")
        continue

    num1 = float(input("Enter first number: "))
    num2 = float(input("Enter second number: "))

    if command == "add":
        print(f"Result: {num1 + num2}")
    elif command == "subtract":
        print(f"Result: {num1 - num2}")
    elif command == "multiply":
```

```
            print(f"Result: {num1 * num2}")
        elif command == "divide":
            if num2 == 0:
                raise ZeroDivisionError
            print(f"Result: {num1 / num2}")
        elif command == "power":
            print(f"Result: {num1 ** num2}")
        elif command == "modulus":
            print(f"Result: {num1 % num2}")
        else:
            print("Invalid command. Try again.")

    except ValueError:
        print("Invalid input. Please enter numbers.")
    except ZeroDivisionError:
        print("Error: Division by zero.")

if __name__ == "__main__":
    cli_calculator()
```

```
Welcome to the CLI Calculator!

Available operations:
add, subtract, multiply, divide
power, modulus, sqrt, factorial, quit
Enter a command: add
Enter first number: 4
Enter second number: 5
Result: 9.0

Available operations:
add, subtract, multiply, divide
power, modulus, sqrt, factorial, quit
Enter a command: quit
Exiting the calculator. Goodbye!
```

3. Menu-Driven Interface

Overview:

- Users interact with the system by selecting options from a menu.
- Easier to use than CLI but can be slower for advanced users.
  A menu-driven program for a simple to-do list.

Exercise:

- Add a feature to mark tasks as completed

```
def menu_driven_todo():
    tasks = []  # Each task will be a dictionary with 'name' and 'completed' status
    while True:
        print("\n--- To-Do List Menu ---")
        print("1. Add Task")
        print("2. View Tasks")
        print("3. Delete Task")
        print("4. Mark Task as Completed")
        print("5. Quit")

        choice = input("Enter your choice: ").strip()

        if choice == "1":
            task = input("Enter the task: ")
            tasks.append({"name": task, "completed": False})
            print(f"Task '{task}' added.")

        elif choice == "2":
            if tasks:
                print("\nTasks:")
                for i, task in enumerate(tasks, 1):
                    status = "✓" if task["completed"] else " "
                    print(f"{i}. [{status}] {task['name']}")
```

```
        else:
            print("No tasks found.")

    elif choice == "3":
        if tasks:
            task_num = int(input("Enter the task number to delete: "))
            if 1 <= task_num <= len(tasks):
                removed_task = tasks.pop(task_num - 1)
                print(f"Task '{removed_task['name']}' deleted.")
            else:
                print("Invalid task number.")
        else:
            print("No tasks to delete.")

    elif choice == "4":
        if tasks:
            task_num = int(input("Enter the task number to mark as completed: "))

            if 1 <= task_num <= len(tasks):
                task = tasks[task_num - 1]
                task["completed"] = True
                print(f"Task '{task['name']}' marked as completed.")
            else:
                print("Invalid task number.")
        else:
            print("No tasks to mark as completed.")

    elif choice == "5":
        print("Exiting the to-do list. Goodbye!")
        break
```

```
        else:
            print("Invalid choice. Try again.")

if __name__ == "__main__":
    menu_driven_todo()
```

4. Form-Based Interface
   Overview:

- Users fill out forms with fields to input data.
- Commonly used in web applications and databases.

A simple form-based program for user registration.

Exercise:

- Add validation for email and password (e.g., password must be at least 8 characters)
- Save the registration details to a file.

LINK FOR CODE:

https://github.com/mrafaqahmed/Userformbasedreg.git

```
--- User Registration Form ---
Enter your name: afaq
Enter your email: mrafaq@gmail.com
Enter your password (min 8 characters): dilkiAWAZ
Enter your age: 19

--- Registration Details ---
Name: afaq
Email: mrafaq@gmail.com
Password: *********
Age: 19
Registration successful! Details saved to registrations.json
```

5. Graphical User Interface (GUI)
   Overview:
   - Users interact with visual elements like buttons, menus, and windows.
   - Most user-friendly but requires more resources to develop.

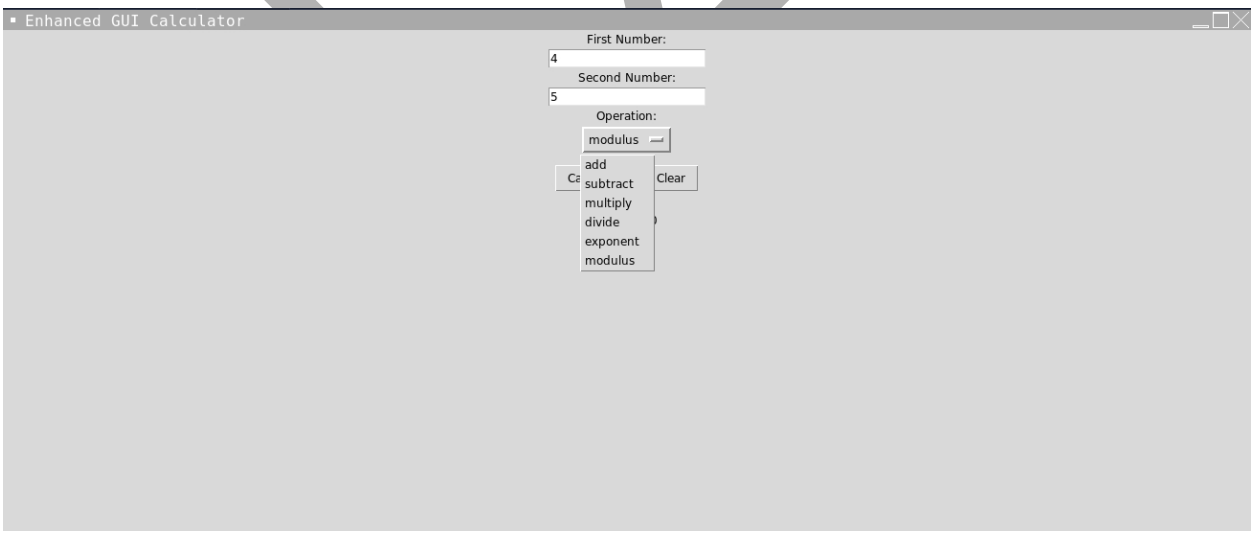
   A simple GUI calculator using tkinter in Python.

   Exercise:

- Add more operations (e.g., exponentiation, modulus).
- Add a clear button to reset the input fields.

LINK FOR CODE:

https://github.com/mrafaqahmed/AsimpleGUIcalculatorusingtkinterinPython.git



6. Natural Language Interface :
   Overview:
   - Users interact with the system using natural language (e.g., voice or text).
   - Requires natural language processing (NLP) techniques.

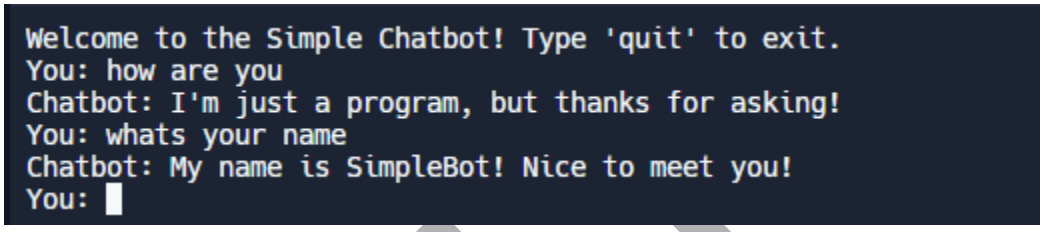
A simple chatbot using Python.

Exercise:

- Add more responses based on user input.

LINK FOR CODE:

https://github.com/mrafaqahmed/ASimplechatbotusingPython.git



7. WIMP and Direct Manipulation Interfaces
   Overview:
   - WIMP stands for Windows, Icons, Menus, Pointer.
   - Users interact with graphical elements using a pointer (e.g., mouse).
   - Direct Manipulation allows users to directly interact with objects on the screen (e.g., drag-and-drop, resizing).

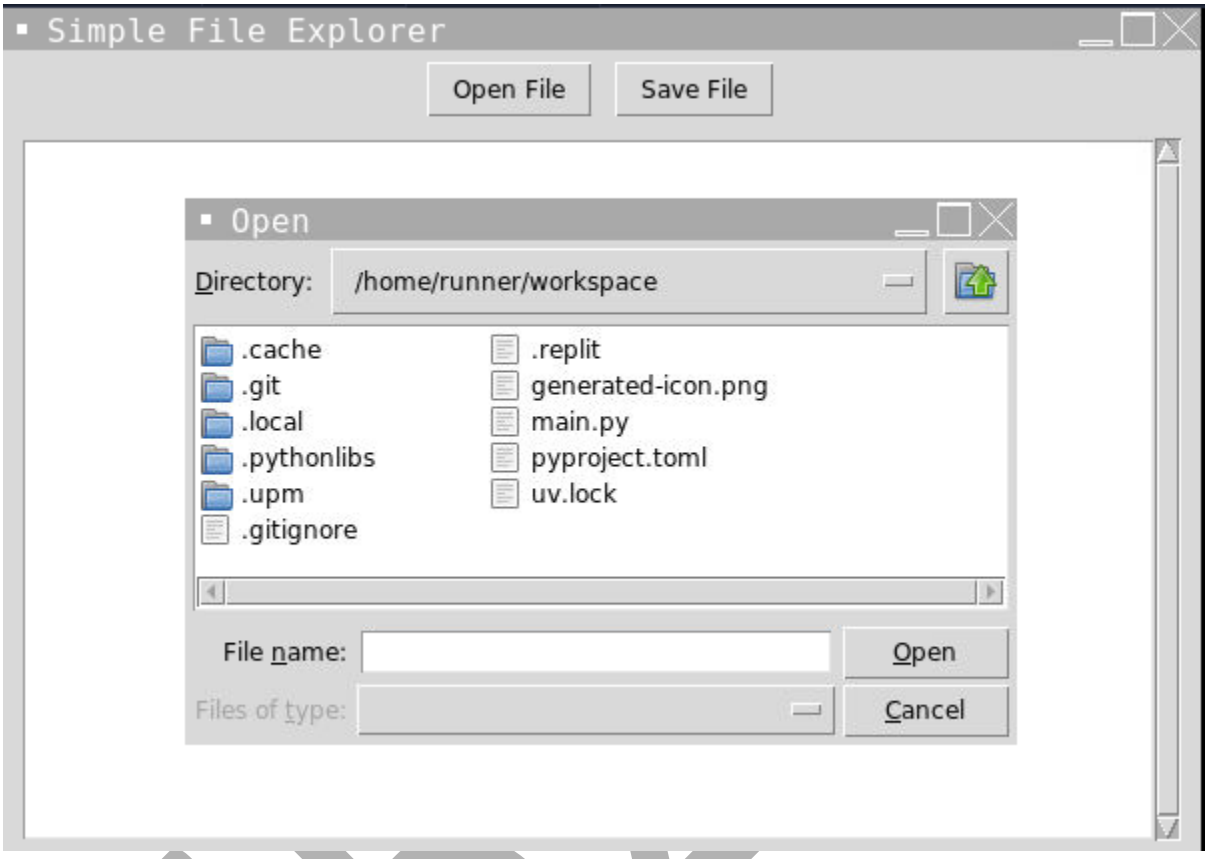
   A simple file explorer using tkinter in Python.

   Exercise:

   - Add a text area to display the contents of the selected file.

LINK FOR CODE:

https://github.com/mrafaqahmed/SimpleFileExplorer.git



7. Lab Exercises CLI and Menu-Based Interface

1. Command-Line Interface (CLI):

Create a simple CLI program in a programming language of choice (Python, Java, etc.) for a library management system. Features to include:

- Adding a book (title, author, ISBN).
- Searching for a book by title or author.
- Exiting the application.

LINK FOR CODE:

https://github.com/mrafaqahmed/librarymanagementsystem.git

```
ollapse
=== Library Management System ===
1. Add a book
2. Search by title
3. Search by author
4. Exit
Enter your choice (1-4): 1

Enter book title: Charlie & the willie woka(BY AFAQ AHMED)
Enter author name: MR AFAQ AHMED
Enter ISBN: 4422

Book added successfully:
Title: Charlie & the willie woka(BY AFAQ AHMED)
Author: MR AFAQ AHMED
ISBN: 4422

=== Library Management System ===
1. Add a book
2. Search by title
3. Search by author
4. Exit
Enter your choice (1-4): 3

Enter author to search: MR AFAQ AHMED

Found books:

Title: Charlie & the willie woka(BY AFAQ AHMED)
Author: MR AFAQ AHMED
```

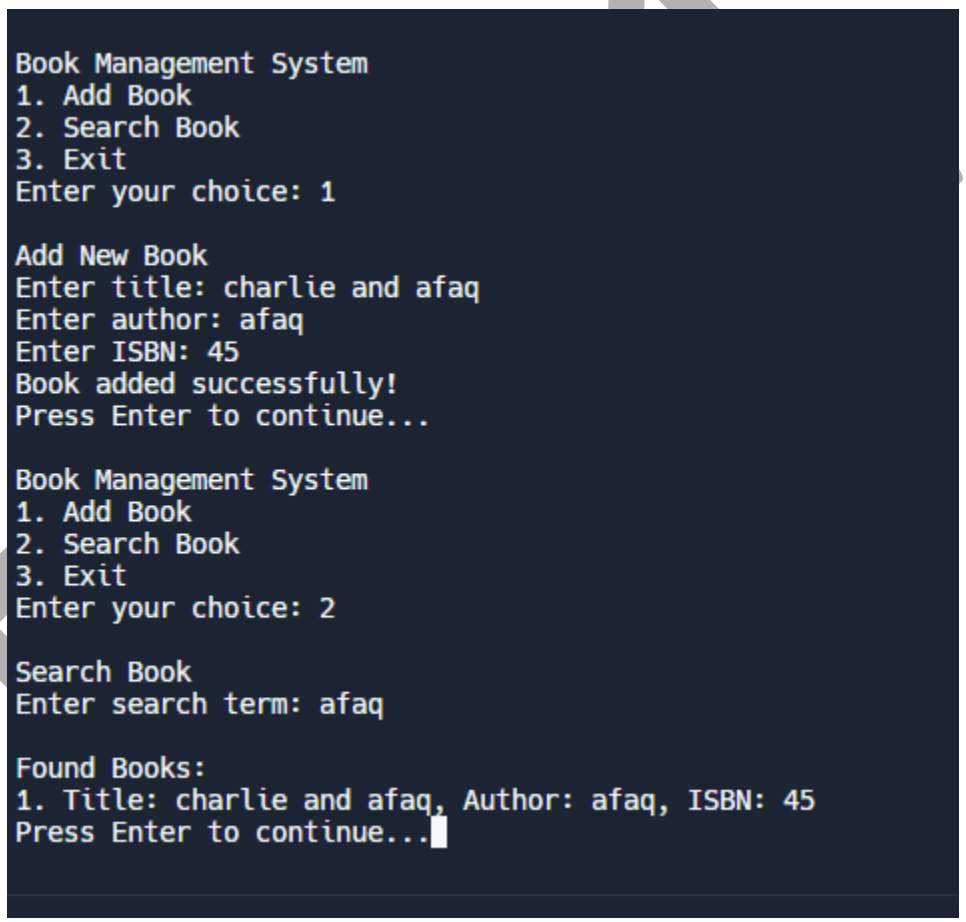
```
=== Library Management System ===
1. Add a book
2. Search by title
3. Search by author
4. Exit
Enter your choice (1-4): 3

Enter author to search: MR AFAQ AHMED

Found books:

Title: Charlie & the willie woka(BY AFAQ AHMED)
Author: MR AFAQ AHMED
ISBN: 4422

=== Library Management System ===
1. Add a book
2. Search by title
3. Search by author
4. Exit
Enter your choice (1-4): ^[[2~
```

2. Menu-Based Interface:
   Convert the CLI application into a Menu-Based Interface using a terminalbased library (e.g., Python's curses, Java's Swing, or console menu frameworks).

   Features to include:
   - A main menu with options like "Add Book," "Search Book," and "Exit."
   - Sub-menus for input and feedback.

LINK FOR CODE:

https://github.com/mrafaqahmed/librarymanagementsystem.git



Designing WIMP and Direct Manipulation Interfaces

WIMP Interface:

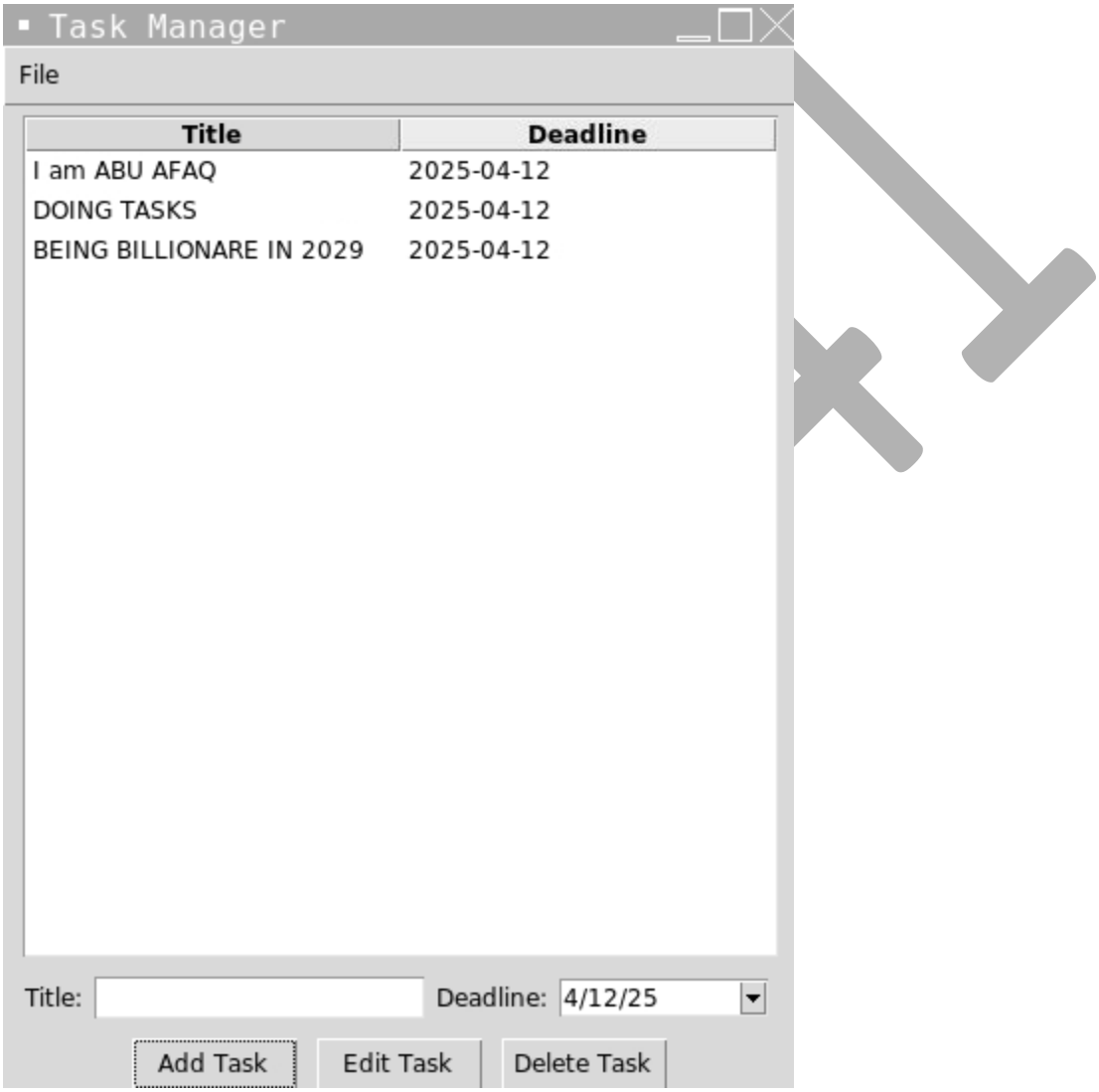
- Using GUI development tools (e.g., Python's Tkinter, JavaFX, or C# Windows Forms), create a WIMP-based application for a task manager.

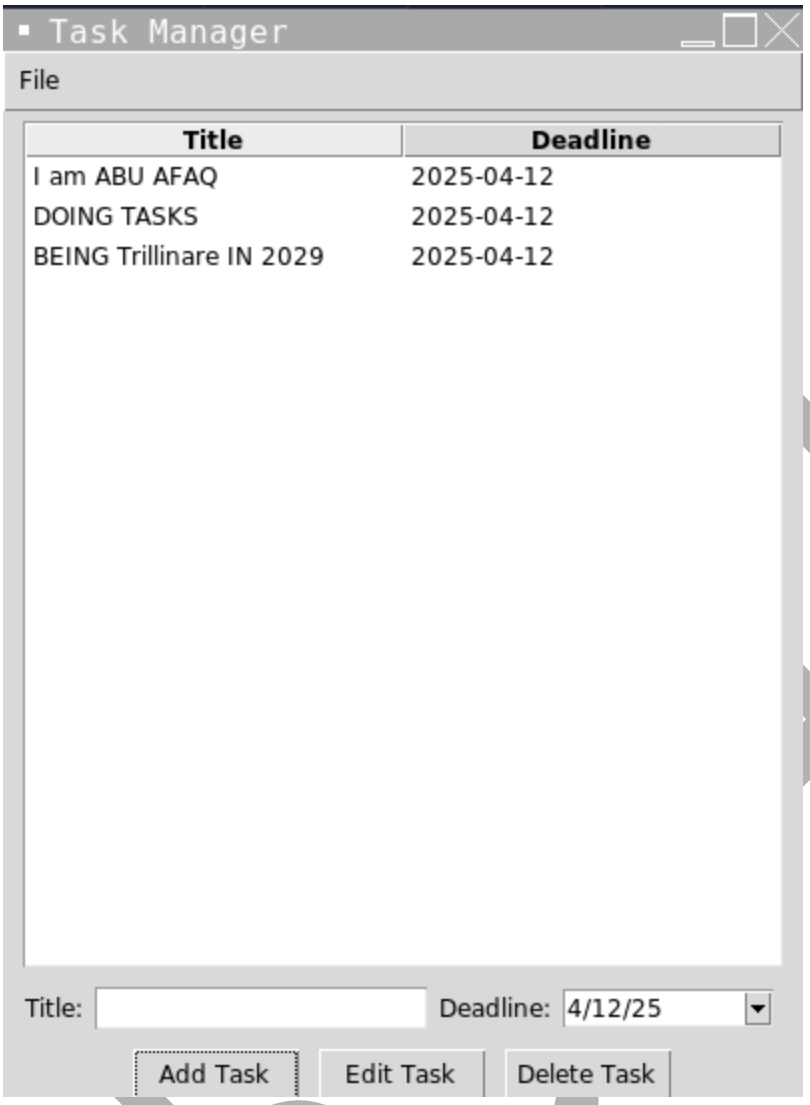
Features to include:

- A window with a list of tasks (title, deadline).
- Buttons for adding, editing, and deleting tasks.
- A menu bar with options like "Save" and "Load."

LINK FOR CODE:

https://github.com/mrafaqahmed/task-manager.git



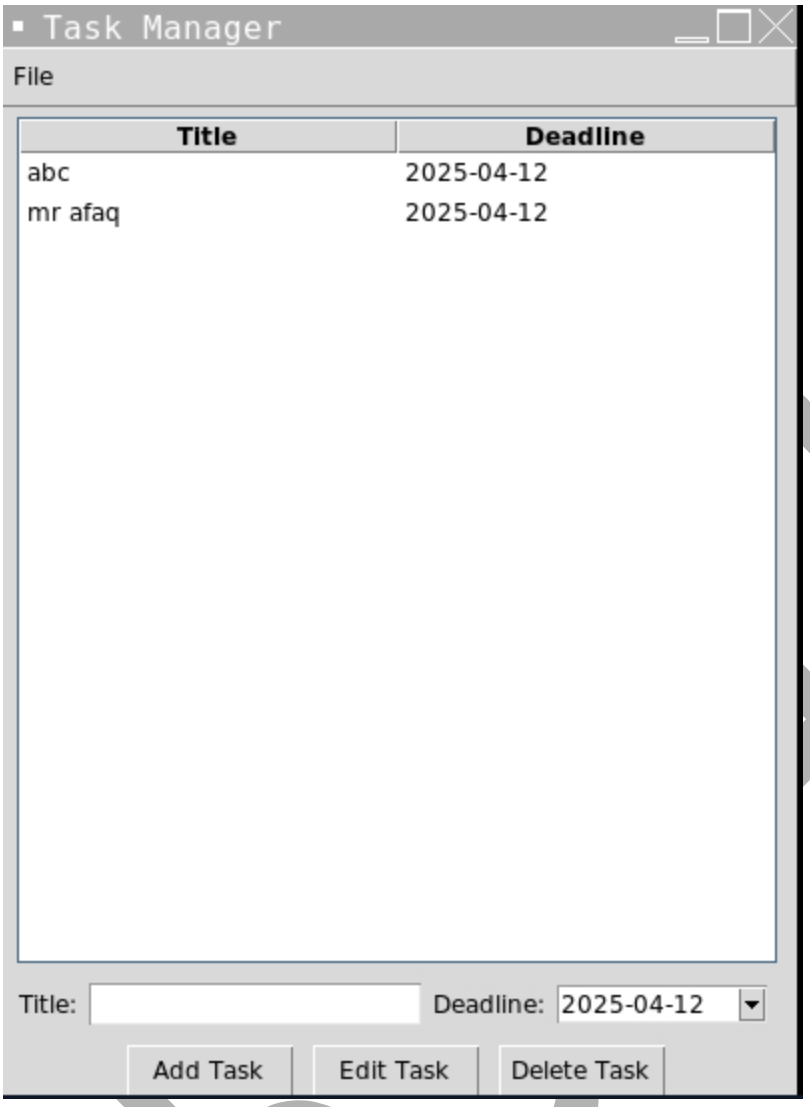


Direct Manipulation Interface:

- Enhance the above interface by enabling drag-and-drop functionality for rearranging tasks or setting deadlines using a graphical calendar.

LINK FOR CODE:

https://github.com/mrafaqahmed/Taskmanager2.git


Task Manager
File
Title
Deadline
abc
2025-04-12
mr afaq
2025-04-12
Title:
Deadline:
2025-04-12
Add Task
Edit Task
Delete Task

2. Design Report:

- Overview of the design process, challenges faced, and usability analysis.

Interface Analysis:

1. GUI (WIMP Interface):

- Windows: Main application window with task list and control panels
- Icons: Basic button representations
- Menus: File menu with save/load functionality
- Pointers: Mouse interaction for selection and manipulation

2. Form-Based Interface:

- Input fields for task title

- DateEntry widget for deadline selection
- Structured data entry approach

3. Direct Manipulation Interface:

- Drag-and-drop task reordering
- Interactive calendar for date selection
- Double-click deadline modification
- Visual feedback during interactions

Design Process Overview:

Core Structure:

- Implemented basic WIMP components using Tkinter
- Organized UI elements in logical frames
- Added scrollable task list for better data management

Enhanced Interaction:

- Integrated direct manipulation features
- Added interactive calendar functionality
- Implemented drag-and-drop capabilities
- Provided immediate visual feedback

Challenges Faced:

Technical:

- Coordinating multiple event bindings
- Managing state during drag operations
- Ensuring data consistency
- Handling date formatting

Design:

- Balancing functionality with simplicity
- Maintaining consistent visual hierarchy
- Managing screen real estate effectively
- Providing clear feedback mechanisms

Usability Analysis:

1. Strengths:

- Intuitive task management
- Direct manipulation capabilities
- Clear visual organization
- Responsive interaction feedback

Areas for Improvement:

- Additional keyboard shortcuts
- Task categorization features
- Search functionality
- Enhanced status indicators

The implementation successfully combines traditional WIMP elements with modern direct manipulation features, creating an efficient and user-friendly task management interface.

Lab 4:

Activity 1: Attention and Distraction in User Interfaces

Expected Outcome:

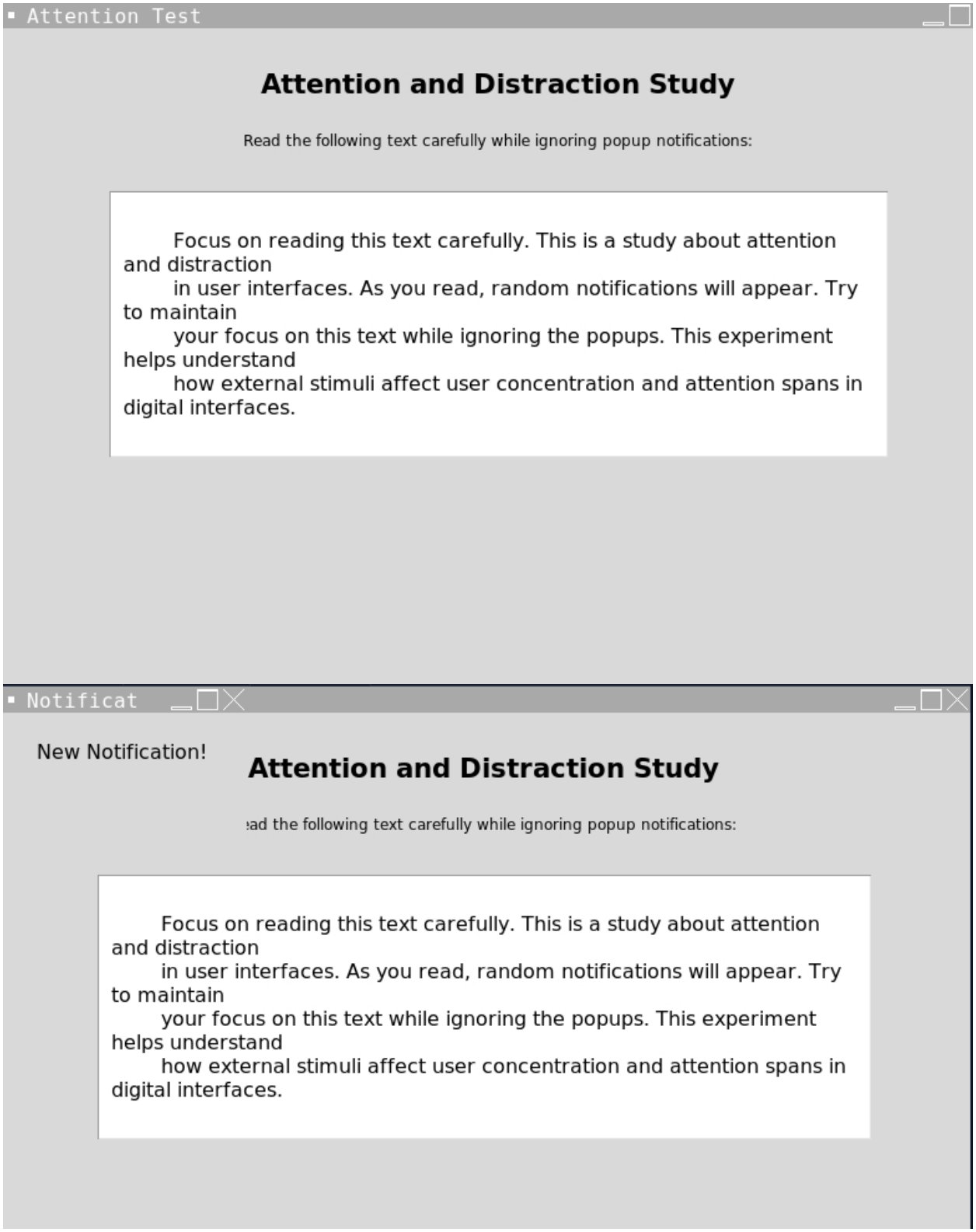
Students will understand how distractions impact attention and propose design solutions to reduce cognitive overload.

This implementation creates a GUI application that:

1. Displays a text passage for users to read
2. Shows random popup notifications every 3-8 seconds
3. Uses proper threading to handle async notifications
4. Has a clean, readable interface

LINK FOR CODE:

https://github.com/mrafaqahmed/Distractioninterface.git


Attention Test
Attention and Distraction Study
Read the following text carefully while ignoring popup notifications:
Focus on reading this text carefully. This is a study about attention and distraction
in user interfaces. As you read, random notifications will appear. Try to maintain
your focus on this text while ignoring the popups. This experiment helps understand
how external stimuli affect user concentration and attention spans in digital interfaces.
Notificat
New Notification!
Attention and Distraction Study
ad the following text carefully while ignoring popup notifications:
Focus on reading this text carefully. This is a study about attention and distraction
in user interfaces. As you read, random notifications will appear. Try to maintain
your focus on this text while ignoring the popups. This experiment helps understand
how external stimuli affect user concentration and attention spans in digital interfaces.

Activity 2: Perception and Recognition in UI Design

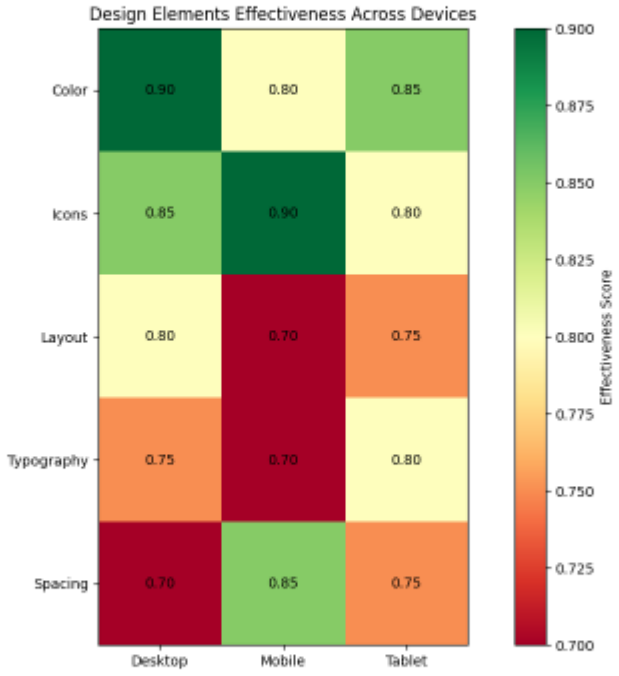
Expected Outcome: Students will understand how colors, icons, and design elements improve usability and user experience.

Understanding UX Design Elements

Learning Outcomes

Students will understand how colors, icons, and design elements improve usability and user experience across different devices.

Design Elements Effectiveness Visualization

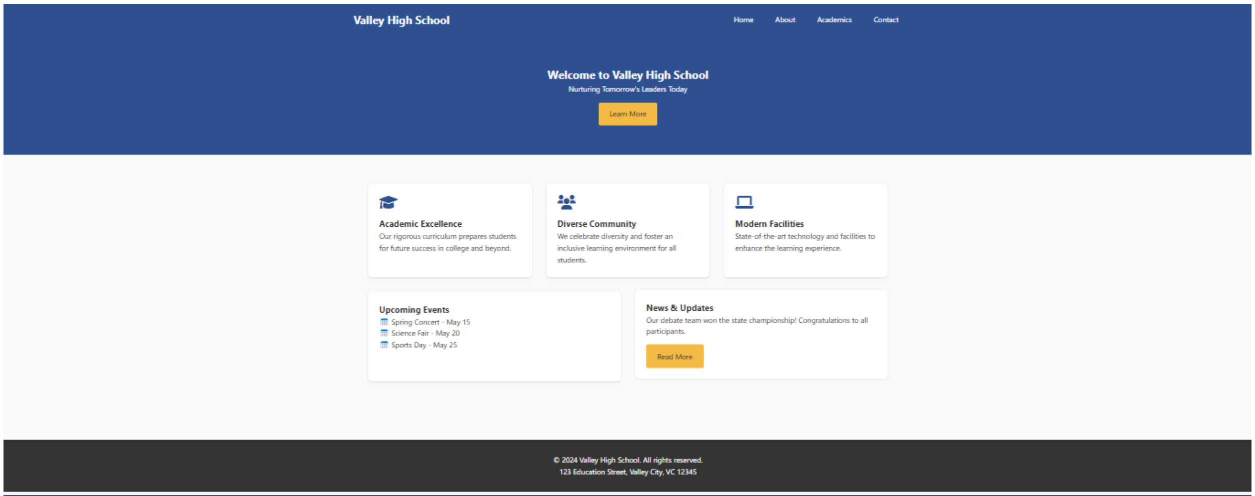


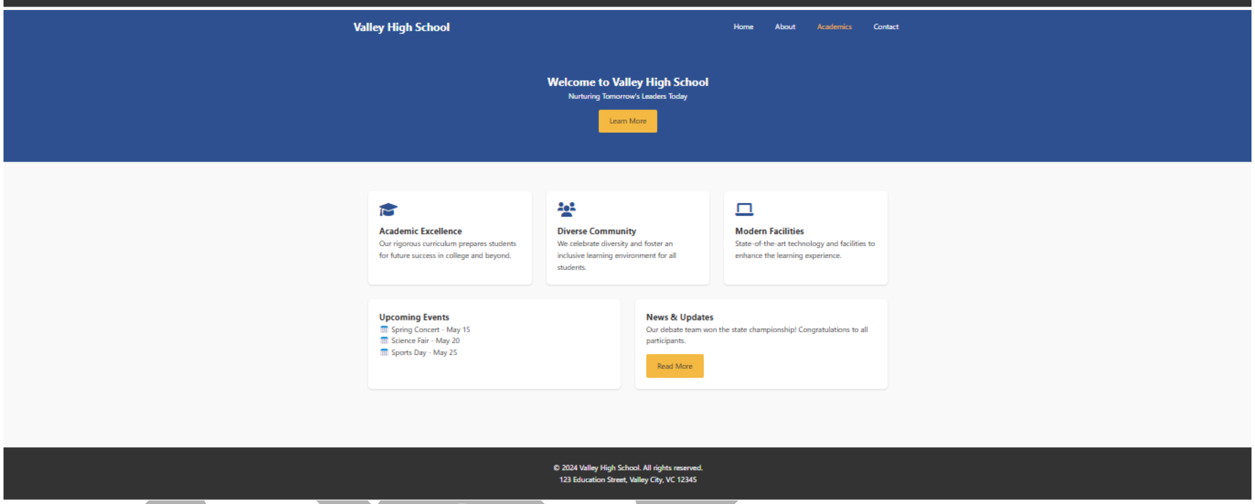
Key Insights

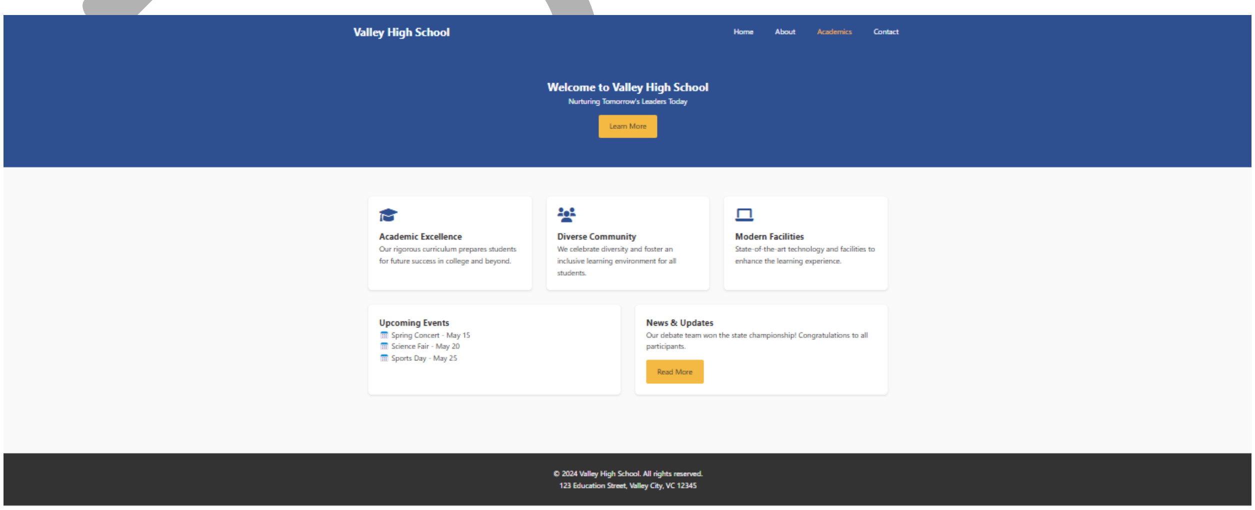
- **Colors:** High effectiveness across all devices, particularly on desktop
- **Icons:** Most effective on mobile devices where space is limited
- **Layout:** Requires careful consideration for different screen sizes
- **Typography:** Consistent effectiveness across devices
- **Spacing:** Particularly important for mobile usability

LINK FOR CODE:

https://github.com/mrafaqahmed/colordesign.git


Valley High School
Home
About
Academics
Contact
Welcome to Valley High School
Nurturing Tomorrow's Leaders Today
Learn More
Academic Excellence
Our rigorous curriculum prepares students for future success in college and beyond.
Diverse Community
We celebrate diversity and foster an inclusive learning environment for all students.
Modern Facilities
State-of-the-art technology and facilities to enhance the learning experience.
Upcoming Events
Spring Concert - May 15
Science Fair - May 20
Sports Day - May 25
News & Updates
Our debate team won the state championship! Congratulations to all participants.
Read More
© 2024 Valley High School. All rights reserved.
123 Education Street, Valley City, VC 12345


Valley High School
Home
About
Academics
Contact
Welcome to Valley High School
Nurturing Tomorrow's Leaders Today
Learn More
Academic Excellence
Our rigorous curriculum prepares students for future success in college and beyond.
Diverse Community
We celebrate diversity and foster an inclusive learning environment for all students.
Modern Facilities
State-of-the-art technology and facilities to enhance the learning experience.
Upcoming Events
Spring Concert - May 15
Science Fair - May 20
Sports Day - May 25
News & Updates
Our debate team won the state championship! Congratulations to all participants.
Read More
© 2024 Valley High School. All rights reserved.
123 Education Street, Valley City, VC 12345


Valley High School
Home
About
Academics
Contact
Welcome to Valley High School
Nurturing Tomorrow's Leaders Today
Learn More
Academic Excellence
Our rigorous curriculum prepares students for future success in college and beyond.
Diverse Community
We celebrate diversity and foster an inclusive learning environment for all students.
Modern Facilities
State-of-the-art technology and facilities to enhance the learning experience.
Upcoming Events
Spring Concert - May 15
Science Fair - May 20
Sports Day - May 25
News & Updates
Our debate team won the state championship! Congratulations to all participants.
Read More
© 2024 Valley High School. All rights reserved.
123 Education Street, Valley City, VC 12345

Activity 3: Memory and Automation in User Interfaces

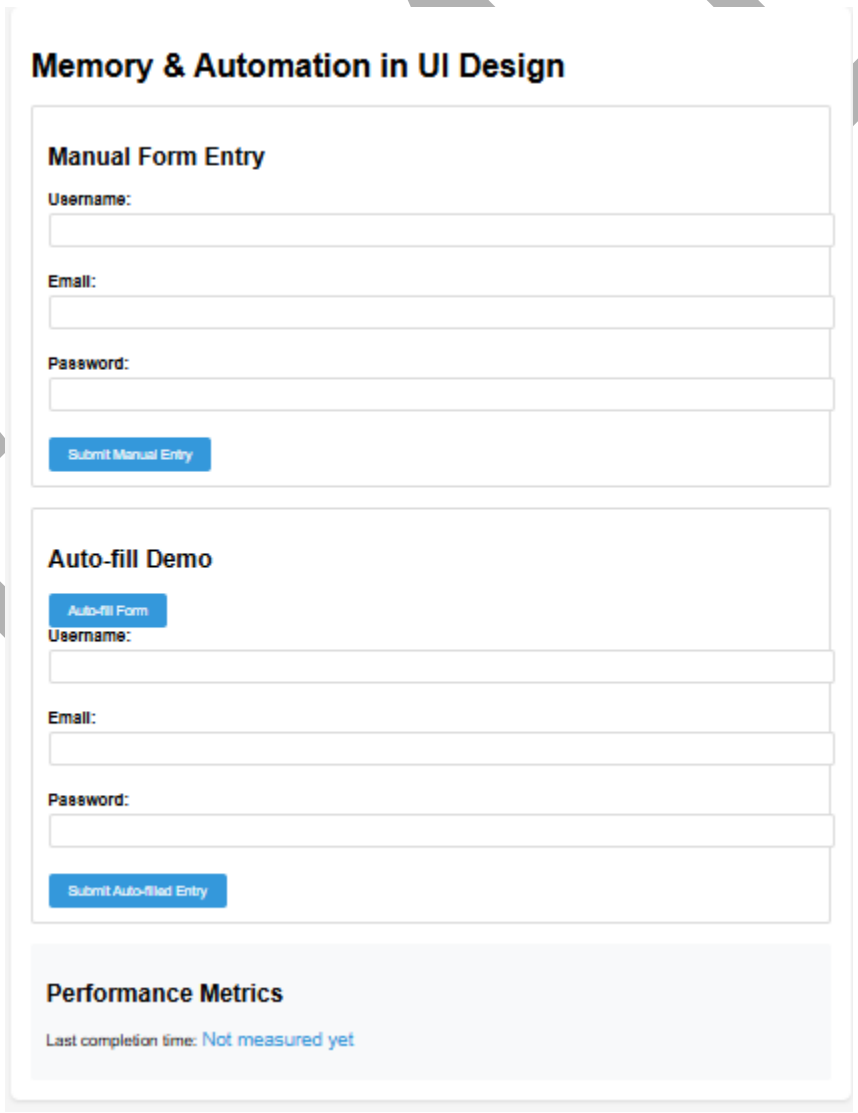
Expected Outcome:

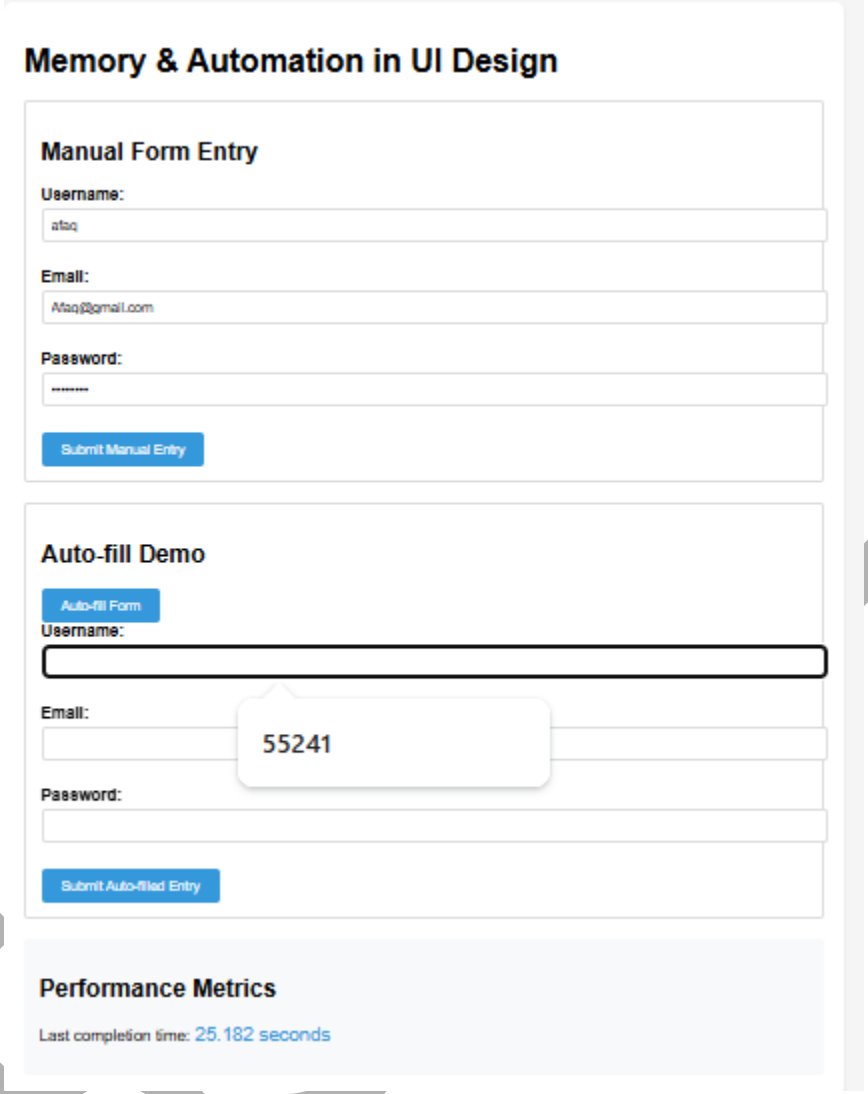
Students will understand how automation assists memory retention and enhances user efficiency in repetitive tasks.

1. Demonstrates manual vs automated form filling
2. Measures completion time for both methods
3. Uses proper UI patterns to reduce cognitive load
4. Implements autocomplete functionality

LINK FOR CODE:

https://github.com/mrafaqahmed/Memory-AutomationinUIDesign.git





Activity 4: Learning and Skill Acquisition in Software Use

Expected Outcome: Students will understand how users learn software tools and how UI design impacts learning curves.

This implementation creates a web-based drawing tool that:

1. Tracks user interaction time with different drawing tools
2. Visualizes learning progress through an interactive graph
3. Provides basic drawing tools (pencil, rectangle, circle)
4. Logs user performance data

LINK FOR CODE:

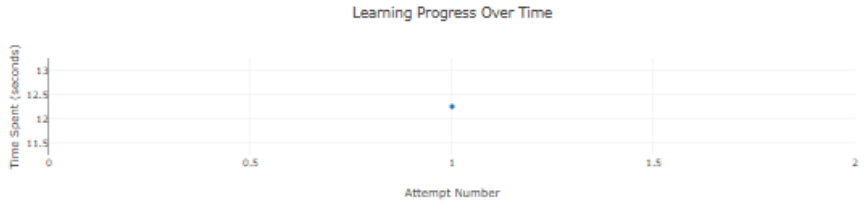
https://github.com/mrafaqahmed/Drawing-Tool-Learning-Assessment.git

50%

Drawing Tool Learning Assessment

Pencil | Rectangle | Circle | Clear | Complete Task

Learning Progress



Drawing Tool Learning Assessment

Pencil | Rectangle | Circle | Clear | Complete Task

Learning Progress

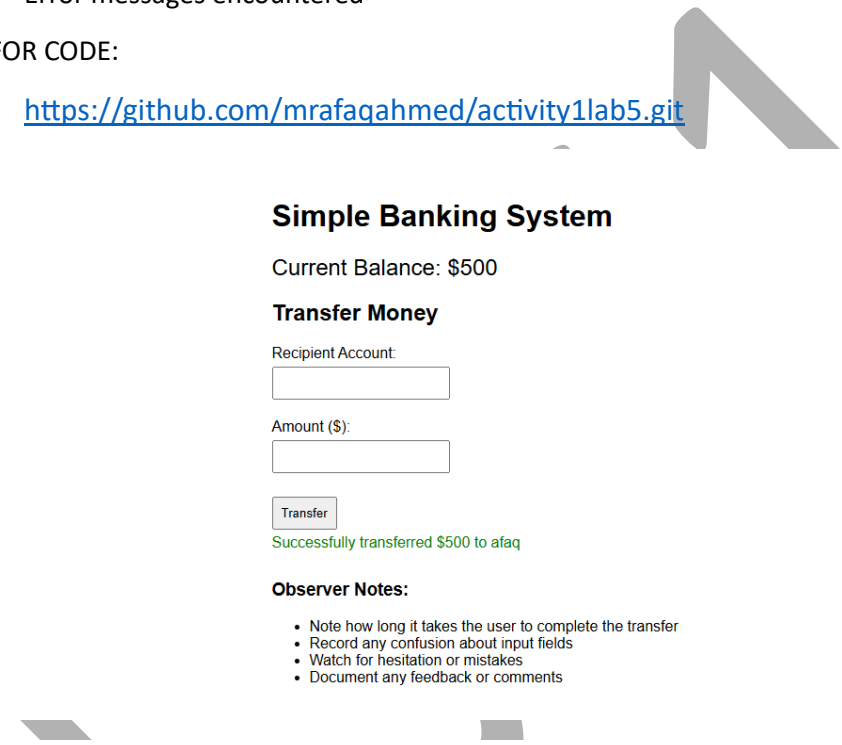
Lab 5:

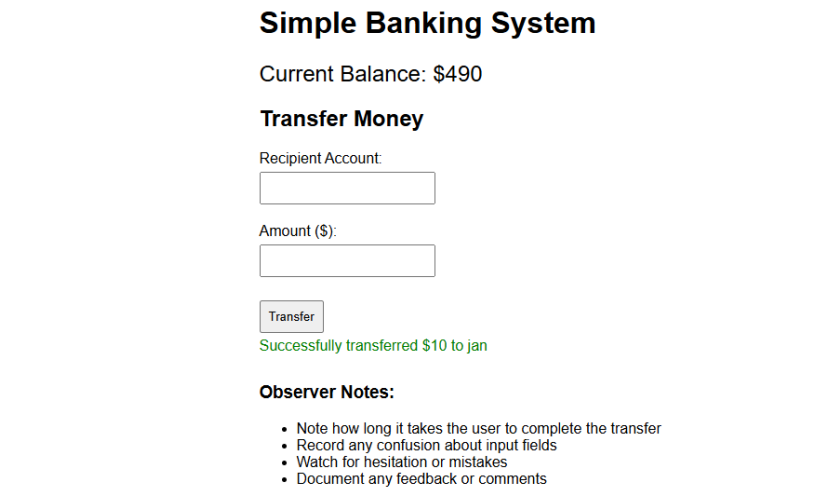
Activity 1 : Usability Testing

- Time taken to complete tasks
- Any confusion with the interface
- Hesitation points
- Error messages encountered

LINK FOR CODE:

https://github.com/mrafaqahmed/activity1lab5.git





Activity 2 : Mental Mode l Mapping

This creates an interactive mental model mapping tool where students can:

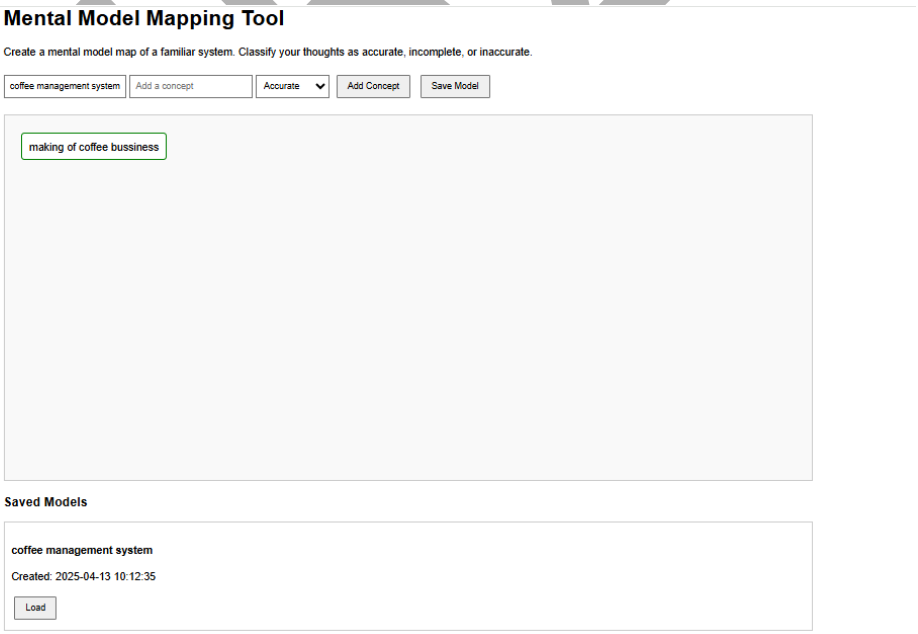
1. Enter a topic (e.g., coffee shop, library)
2. Add concepts/nodes and classify them as:
    - Accurate (green border)
    - Incomplete (orange border)
    - Inaccurate (red border)
3. Drag nodes around to organize their thoughts
4. Save their models and load previous ones
5. Share and discuss their maps with others

To use:

1. Click the Run button
2. Open the web app
3. Start creating mental models by entering concepts and classifying them
4. Save your models and view others' saved models

LINK FOR CODE:

https://github.com/mrafaqahmed/activity2.git



Activity 3 : Norman's Model Role-Play

This creates an interactive simulation where students can:

1. Select a scenario (ATM, Food Ordering, Flight Booking)
2. Work through Norman's Model stages:
    - Planning: Define their goal
    - Execution: Describe their actions
    - Evaluation: Reflect on the outcome
3. Log all interactions for later discussion

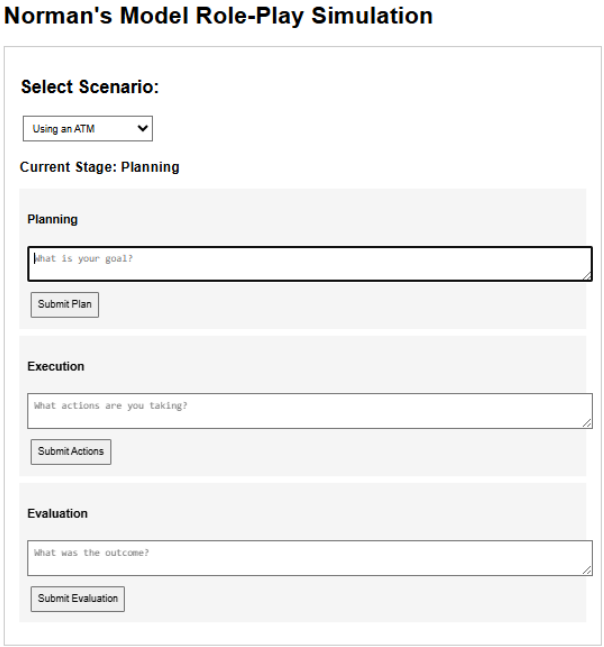
To use:

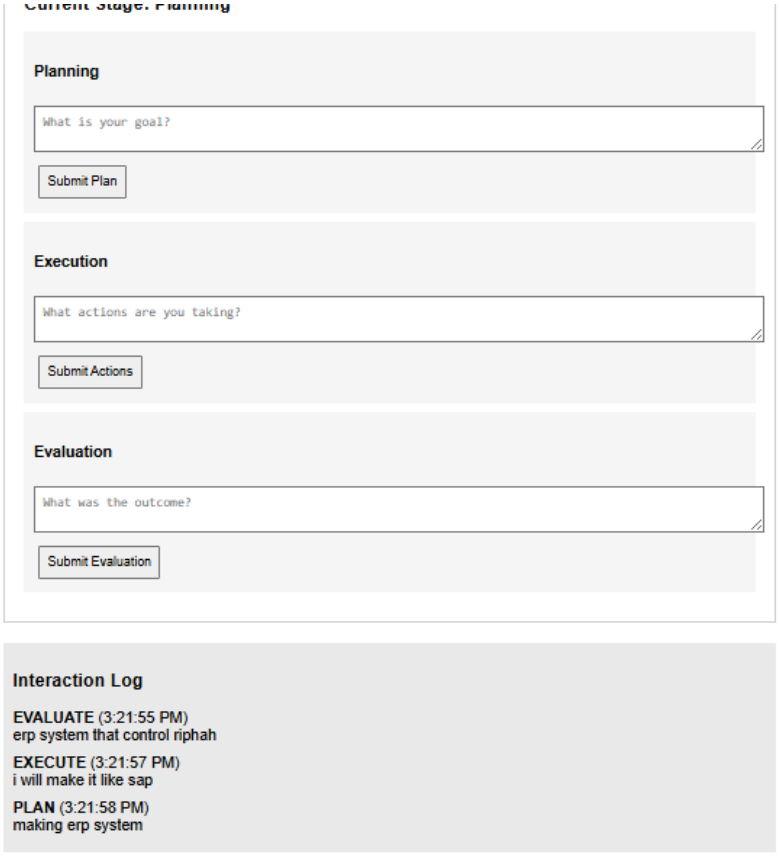
1. Run the application
2. Groups can take turns role-playing different scenarios
3. One student acts as the user entering their thoughts/actions
4. Others observe and discuss the interaction stages
5. Use the logged interactions for class discussion

The simulation helps visualize how users plan, execute, and evaluate their interactions with systems.

LINK FOR CODE:

https://github.com/mrafaqahmed/activity3lab5.git





Activity 4 : User Interface Redesign:

This creates an interactive interface where students can:

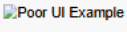
1. Analyze a poor UI design example
2. Consider Norman's Model principles
3. Submit their redesign proposals
4. View and discuss other students' submissions

LINK FOR CODE:

https://github.com/mrafaqahmed/activity4.git

UI Redesign Activity

Case Study: Confusing Settings Menu



This settings menu has several usability issues:

- Inconsistent navigation patterns
- Unclear labeling of options
- Poor information hierarchy
- Confusing grouping of related items

Design Principles to Consider:

- Visibility of system status
- Match between system and real world
- User control and freedom
- Consistency and standards

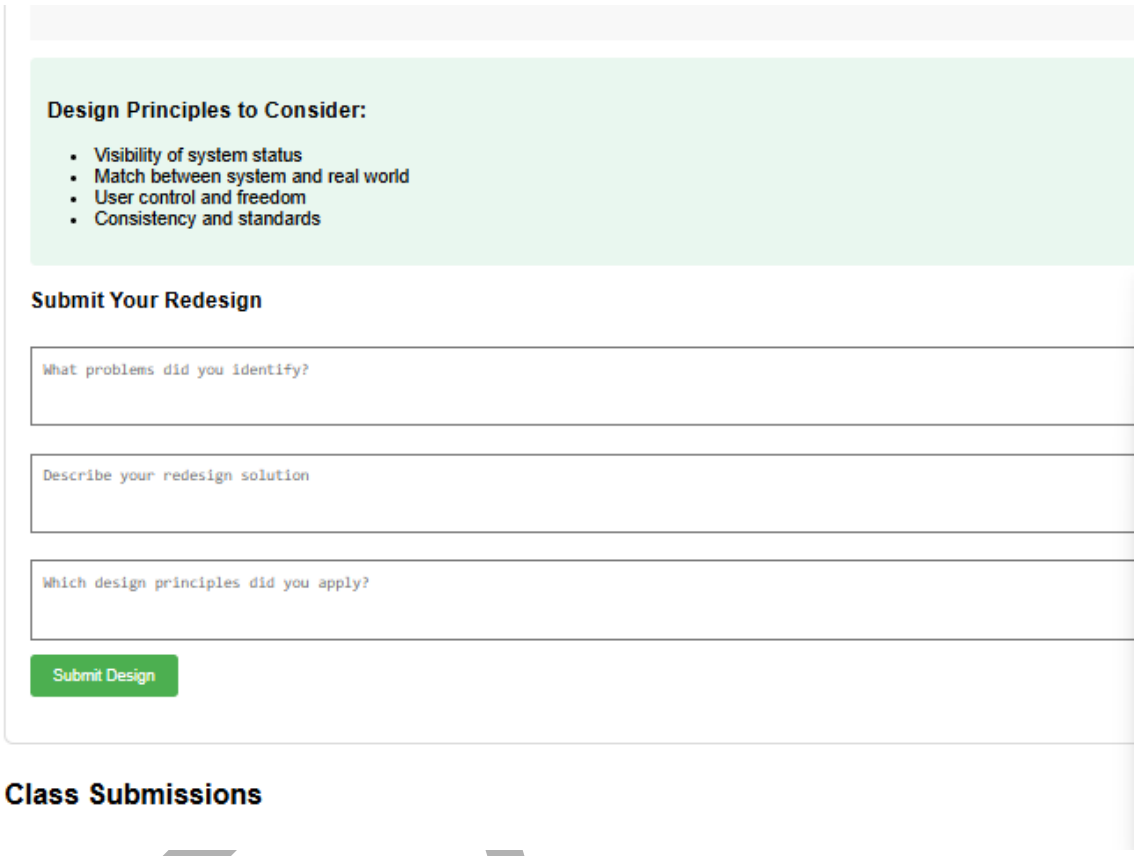
Submit Your Redesign

What problems did you identify?

Describe your redesign solution

Which design principles did you apply?

Submit Design



Deliverables:

1. Norman's Model Diagram: An SVG visualization of the action cycle
2. Mental Model Evolution: A section for students to reflect on their mental model development
3. Interface Design Improvements: A structured form for submitting and viewing UI design improvements

The application is organized into tabs for easy navigation between the different sections. Users can:

- View the Norman's Model diagram
- Submit and view reflections on mental model evolution
- Document and share interface design improvements

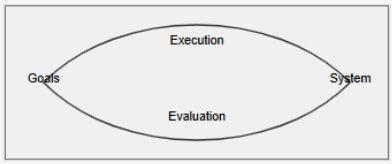
LINK FOR CODE:

https://github.com/mrafaqahmed/LAB.git

UI/UX Design Analysis

Norman's Model | Mental Model | Interface Design

Norman's Model



Norman's Action Cycle: Goals → Execution → System → Evaluation

UI/UX Design Analysis

Norman's Model | Mental Model | Interface Design

Mental Model Evolution

Share your thoughts on how your mental model of the system has evolved...

Submit Reflection

making app like chatgpt

UI/UX Design Analysis

Norman's Model | Mental Model | Interface Design

Interface Design Improvements

What usability issues did you identify?

Describe your improved design solution

Which design principles did you apply?

Submit Design

Lab 6:

Activity 1: This following paragraph is about the process of compiling usability evaluation findings after multiple evaluators have independently assessed a system. It explains how evaluators come together to merge their findings into a single report, ensuring that only distinct usability issues are included to avoid redundancy. Additionally, Figure7.3 introduces the Issues-Evaluators Matrix, a tool that helps designers analyze the effectiveness of each evaluator. In this matrix, rows represent evaluators and columns represent the identified usability issues (Problems). But there is confusion when looking at the figure. It is difficult to understand what the graph actually represents. “Once every evaluator “Once every evaluator has performed the evaluation individually and independently, next step is to get together, aggregate the findings and compile a report, of all distinct usability issues, that can be presented and discussed with the design team. Since multiple evaluators may find same problems, so rather than presenting a redundant set of identified problems, it would be better to include them once in the report. As a designer, we should ask evaluators to include in this report is the Issues-Evaluators matrix (Figure-3), where rows represent evaluators and columns represent the issues identified. This matrix would help designers in differentiating between a very good evaluator and a not very good evaluator.

Understanding the Matrix:

- **Rows = Evaluators:** Each horizontal row represents a different evaluator who independently assessed the system.
- **Columns = Usability Problems:** Each vertical column represents a distinct usability issue or problem found in the system.
- **Black Squares:** A black square at the intersection of a row and column means that the evaluator (row) identified that particular usability problem (column).

Purpose of the Matrix:

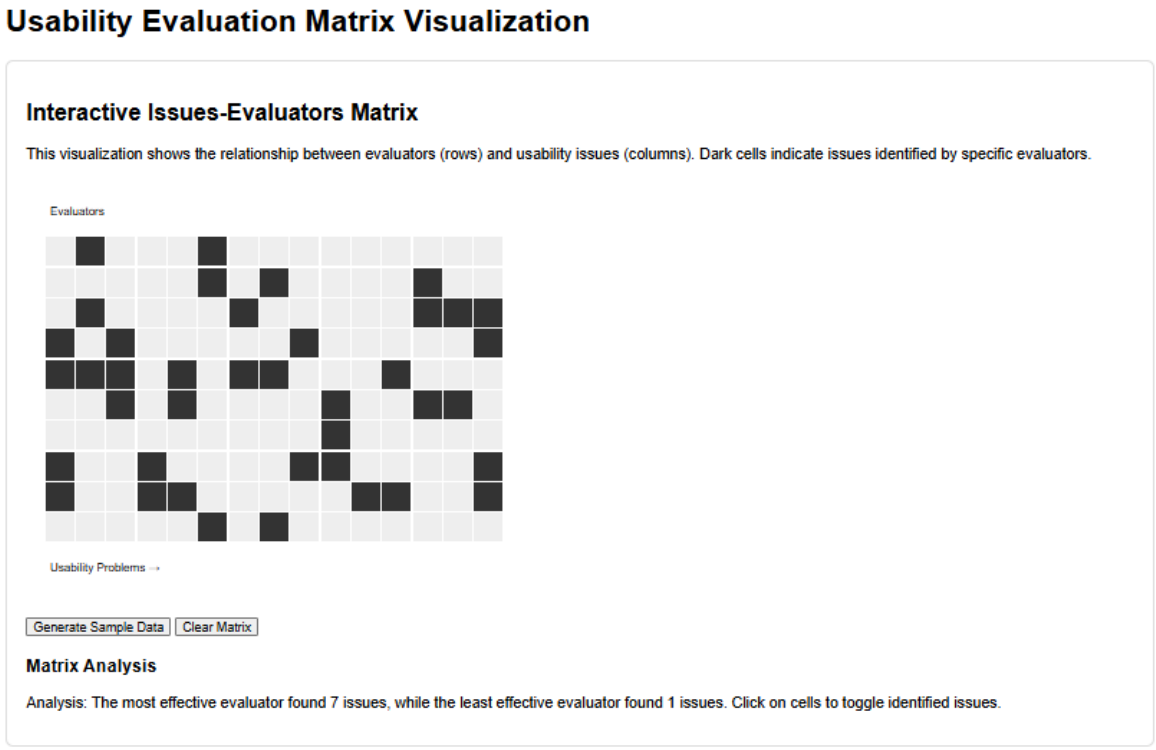
1. **Redundancy Handling:** Since multiple evaluators may find the same problems, this matrix helps to identify overlapping findings and avoid listing the same issue multiple times in the final report.
2. **Evaluator Effectiveness:** It visually shows which evaluator found more issues (rows with more black squares), and which problems were most commonly found (columns with more black squares).
3. **Insight for Designers:** Designers can use this to:
    - Identify **strong evaluators** (those who find more issues).
    - Spot **commonly agreed issues** (columns with many black squares).
    - Detect **rarely found problems** (columns with only one or two black squares).

Common Confusion:

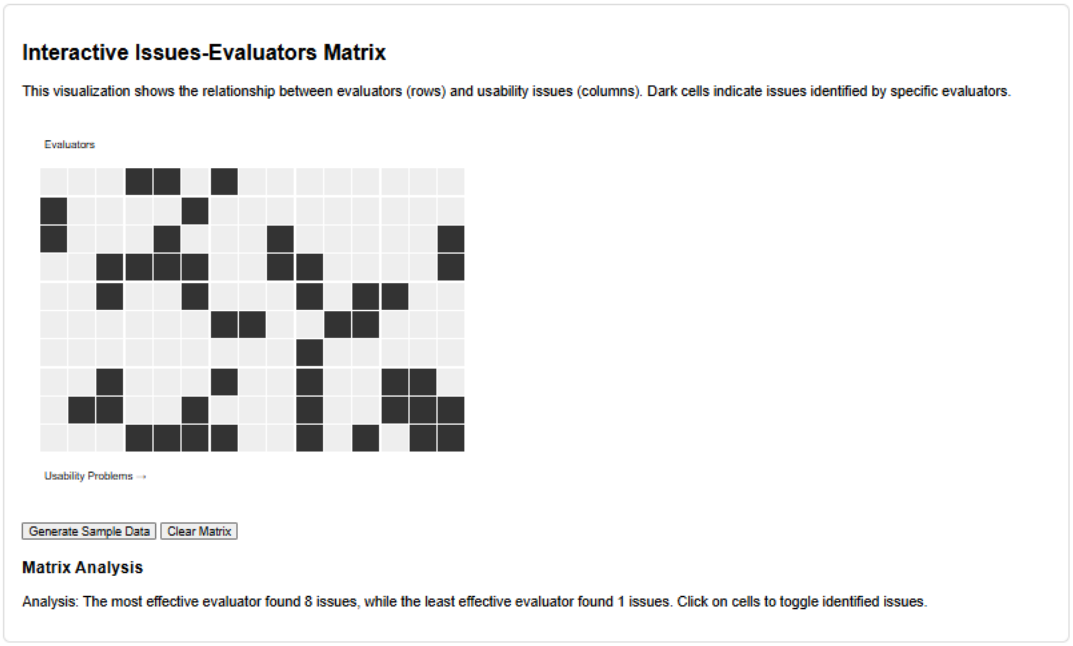
People might find the matrix confusing at first because it looks like a scatter plot. However, it's not about plotting points—it's a **binary matrix** where presence or absence of an issue detection is marked for each evaluator.

CODE OF THIS:

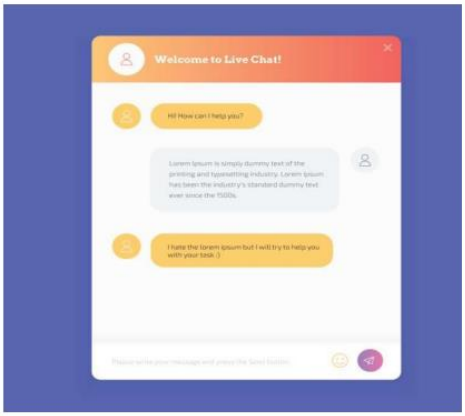
https://github.com/mrafaqahmed/lab61task.git



Usability Evaluation Matrix Visualization



Activity 2: Redesigning the following Chat window using alternative design approaches Or take a simple chat window of your choice and redesign it using alternative design interaction approaches from HCI. Your goal is to make it more aesthetic, user-friendly, and interactive. What to Do: 1. Examine Existing Chat Window Designs – Identify usability issues, limitations, and areas for improvement. 2. Apply Alternative Design Approaches – Think about how you can enhance usability, accessibility, and visual appeal. Consider features like dark mode, custom themes, emoji reactions, gesture controls, or voice input. 3. Create a Prototype – Use a tool like Figma, Adobe XD, Sketch, or a Python GUI framework (Tkinter, PyQt) to build your redesigned version. 4. Explain Your Design Choices – Write a short explanation describing the improvements you made and why they enhance user experience. What to Submit: • A prototype or wireframe of your redesigned chat window.

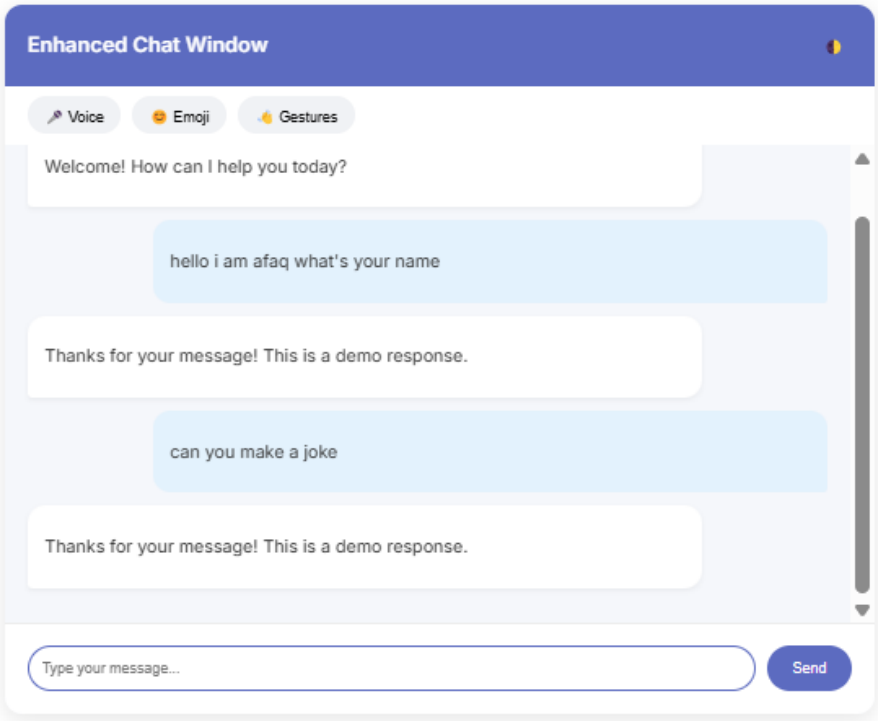


The enhanced chat window includes:

1. Modern UI with smooth animations and transitions
2. Dark/light theme toggle
3. Responsive design for mobile devices
4. Feature buttons for voice input, emoji picker, and gestures
5. Improved message bubbles with visual hierarchy
6. Interactive elements with hover effects
7. Clean and accessible typography using Inter font
8. Placeholder implementations for advanced features

LINK FOR CODE:

https://github.com/mrafaqahmed/lab62task.git



LAB 7:

Lab Activities

Activity 1:

Heuristic Evaluation

1. Task: Choose a website or application (e.g., an e-commerce site, a mobile app, or a university portal).
2. Steps:

- Evaluate the interface using Nielsen's Ten Heuristics.

- Identify at least three usability issues based on the heuristics.

- Suggest improvements for each issue.

3. Deliverable: A report summarizing your findings, including screenshots of the interface, the issues identified, and proposed solutions.

Heuristic Evaluation Report

1. Introduction

- Application/Website Evaluated: Amazon (www.amazon.com)
- Purpose of Evaluation: Assess the usability of the interface using Nielsen's Ten Heuristics.

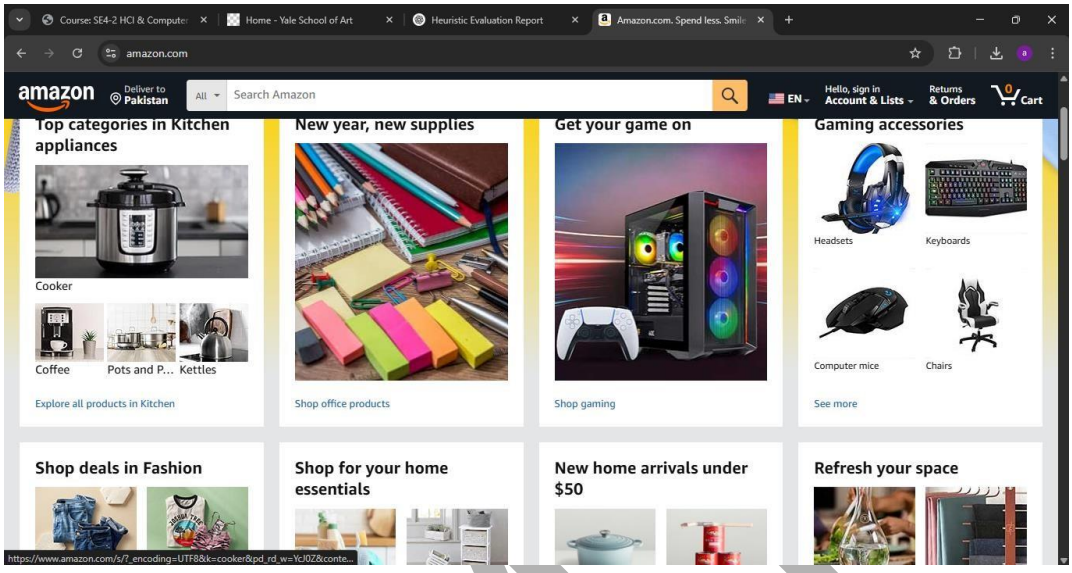
2. Methodology

- Nielsen's Ten Heuristics were used to assess the usability of the selected interface.
- The evaluation was conducted by navigating through key features and documenting usability issues.
- Screenshots were taken to highlight the identified issues.

3. Identified Usability Issues Issue

1: Cluttered Homepage

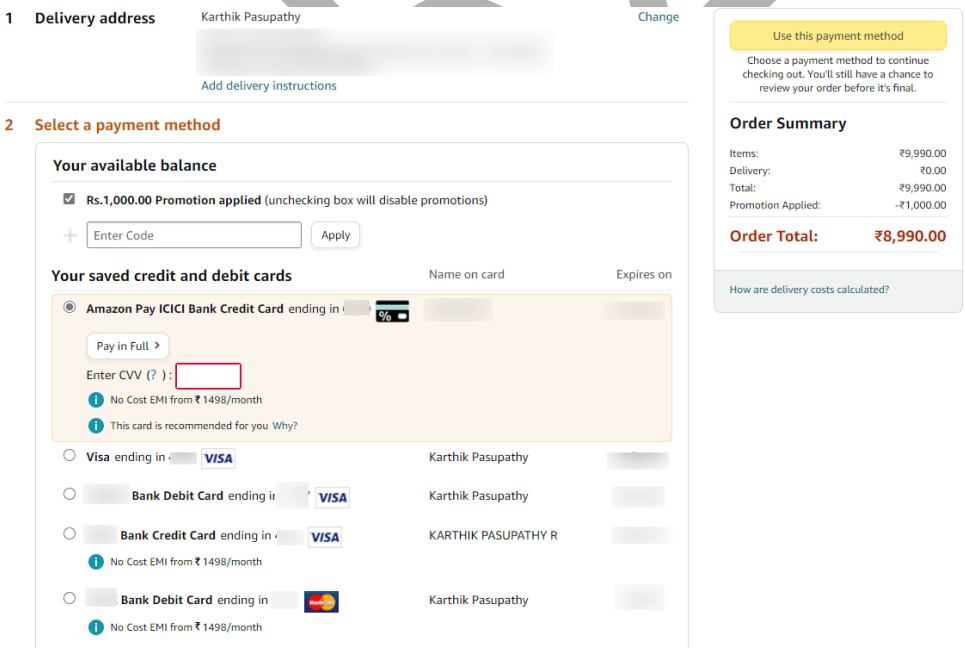
- Heuristic Violated: Aesthetic and Minimalist Design
- Description: The Amazon homepage contains an overwhelming amount of information, including banners, product recommendations, and advertisements, which can make it difficult for users to focus on their intended task.
- Screenshot:



- Suggested Improvement: Reduce visual clutter by prioritizing essential content and allowing users to customize their homepage view.

Issue 2: Confusing Checkout Process

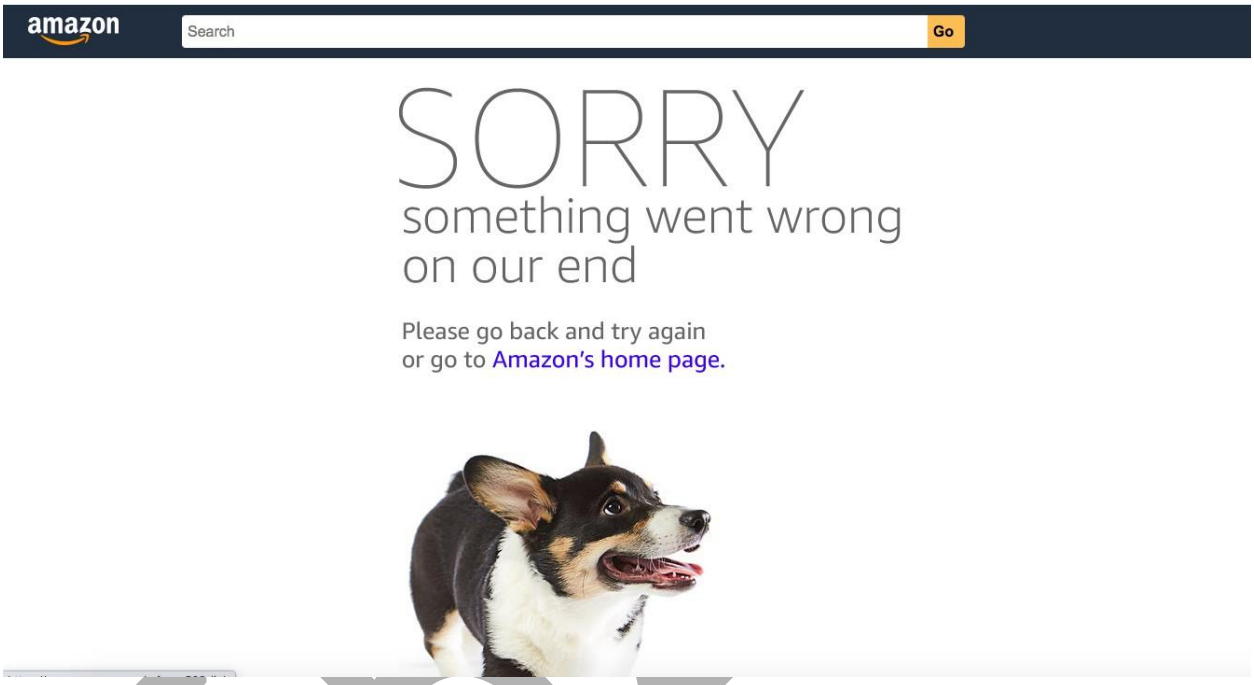
- Heuristic Violated: User Control and Freedom
- Description: During checkout, users may find it difficult to edit their cart or change shipping details without going back multiple steps, which can be frustrating.
- Screenshot:



- Suggested Improvement: Implement a clear and easily accessible navigation path within the checkout process to allow seamless modifications.

Issue 3: Poor Error Handling in Search Function

- Heuristic Violated: Error Prevention
- Description: When a user enters a slightly misspelled product name, the search function may not always provide relevant results, leading to frustration.
- Screenshot:



Suggested Improvement: Improve search error handling by offering better autocorrect suggestions and displaying relevant alternative results.

4. Conclusion: Amazon is a robust e-commerce platform with many useful features, but some usability issues can impact the user experience. Addressing these problems can enhance usability and customer satisfaction.

Activity 2: Redesign a UI Component

1. Task: Select a poorly designed UI component (e.g., a confusing form, a cluttered dashboard, or an unclear navigation menu).

2. Steps: - Analyze the component using the relevant heuristics.
- Redesign the component to improve usability and UX.
- Justify your redesign choices using Nielsen's heuristics.

3. Deliverable: A sketch or wireframe of the redesigned component along with an explanation of how it addresses the usability issues. A poorly designed UI component from Amazon that often causes usability issues is the "Filter & Sort" menu on mobile. Users frequently struggle with:

- Overloaded filters: Too many options without clear hierarchy.
- Hidden or unclear sorting options: Users must tap through multiple menus to find basic sorting (e.g., price low to high).
- Lack of visibility: Applied filters are not always clearly visible.
- Poor feedback: No immediate response when a filter is applied. Redesign Plan

1. Issues & Analysis (Using Nielsen's Heuristics)

- Visibility of system status: Users don't get immediate feedback when applying filters.
- Match between system & real world: The filter categories aren't always intuitive.
- User control & freedom: Users struggle to reset filters easily.
- Consistency & standards: Sorting and filtering are mixed inconsistently.
- Recognition rather than recall: Applied filters are hidden, requiring users to recall their selections.

2. Redesigned Wireframe & Explanation I'll create a wireframe showing a simplified, clearer filter and sort menu with:

- Sticky applied filters bar: Always visible for easy modification.
- Grouped sorting & filtering options: More intuitive layout.
- Real-time feedback: Live preview of applied changes.
- One-tap reset: Easily clear all filters.

How This Redesign Improves Usability (Using Nielsen's Heuristics)

- Visibility of System Status – Applied filters are shown in a sticky bar, so users always see their selections.
- Match Between System & Real World – Filters are grouped in a logical, user-friendly way.
- User Control & Freedom – A "Reset All" button allows quick removal of filters.
- Recognition Rather Than Recall – Selected filters remain visible, eliminating guesswork.
- Flexibility & Efficiency – Live preview updates results in real-time.

Activity 3: Group Discussion

1. Task: Discuss the following questions in groups: - Which of Nielsen's heuristics do you think is the most important?
2. Why? - Can you think of a real-world example where a system violates one or more of these heuristics? How does it impact the user experience?

3. - How can Nielsen's heuristics be applied to emerging technologies like voice assistants or AR/VR interfaces?

Deliverable: A summary of your group's discussion, including key points and examples.

Most Important Heuristic and Why Key Point: The most important heuristic varies based on context, but "Visibility of System Status" is often considered crucial.

Reason: Users need clear feedback to understand what is happening, reducing confusion and frustration. Example: In e-commerce, a loading spinner or confirmation message when adding an item to the cart reassures users that their action was successful.

Real-World Example of a Violation & Its Impact

Example: The Windows 8 Start Menu violated the "Match Between System & Real World" heuristic by removing the traditional Start button. Impact:

- Users struggled to find applications, increasing frustration.
- Productivity dropped as users had to relearn basic navigation.
- Microsoft later reintroduced the Start button in Windows 10, acknowledging user feedback.

3. Applying Nielsen's Heuristics to Emerging Technologies

- Voice Assistants (e.g., Alexa, Siri) o "Recognition Rather Than Recall": Users should be able to phrase commands naturally without memorizing exact phrases. o "Visibility of System Status": Providing audible or visual confirmation (e.g., "Playing your song now") helps users understand responses.
- AR/VR Interfaces o "User Control & Freedom": Users should easily undo actions, such as exiting a virtual space without confusion. o "Consistency & Standards": Gestures and interactions should follow common patterns across different applications to reduce cognitive load

Activity Questions

1. Reflection Questions: - How do Nielsen's heuristics contribute to a positive user experience? - What challenges might designers face when applying these heuristics to complex systems?

2. Application Questions: - Evaluate the login page of a website using Nielsen's heuristics. What improvements would you suggest? - How can the heuristic "Recognition Rather Than Recall" be applied to a mobile banking app?

3. Critical Thinking Questions: - Are there any limitations to Nielsen's heuristics? Can you think of situations where following these heuristics might not lead to the best design? - How do cultural differences impact the application of these heuristics in global products?

1. Reflection Questions

How do Nielsen's heuristics contribute to a positive user experience? Nielsen's heuristics ensure usability by making interfaces intuitive, responsive, and user friendly. They help minimize errors, reduce cognitive load, and improve efficiency, leading to a seamless and enjoyable experience.

What challenges might designers face when applying these heuristics to complex systems? o Balancing simplicity with functionality in feature-rich applications.

- Ensuring consistency across different platforms and devices.
- Handling edge cases where heuristics may conflict (e.g., security vs. user freedom).
- Adapting heuristics to new technologies like AI-driven interfaces.

2. Application Questions

Evaluate the login page of a website using Nielsen's heuristics. What improvements would you suggest? Example: A typical e-commerce login page

- Visibility of System Status: Show real-time validation (e.g., "Password must be 8+ characters").
- Error Prevention: Use clear error messages (e.g., instead of "Invalid login," specify "Incorrect password").

User Control & Freedom: Offer a "Forgot Password?" option.

- Consistency & Standards: Use familiar UI patterns (e.g., password eye icon for visibility). o Recognition Rather Than Recall: Allow autofill for returning users.

How can the heuristic "Recognition Rather Than Recall" be applied to a mobile banking app?

- Pre-filled fields: Display frequently used account numbers or recent transactions.
- Biometric authentication: Face ID or fingerprint login instead of requiring manual input.
- Quick actions: Show suggested transactions (e.g., "Pay last month's bill again").
- Intuitive icons & labels: Instead of text-heavy menus, use familiar banking symbols.

3. Critical Thinking Questions

Are there any limitations to Nielsen's heuristics? Can you think of situations where following these heuristics might not lead to the best design?

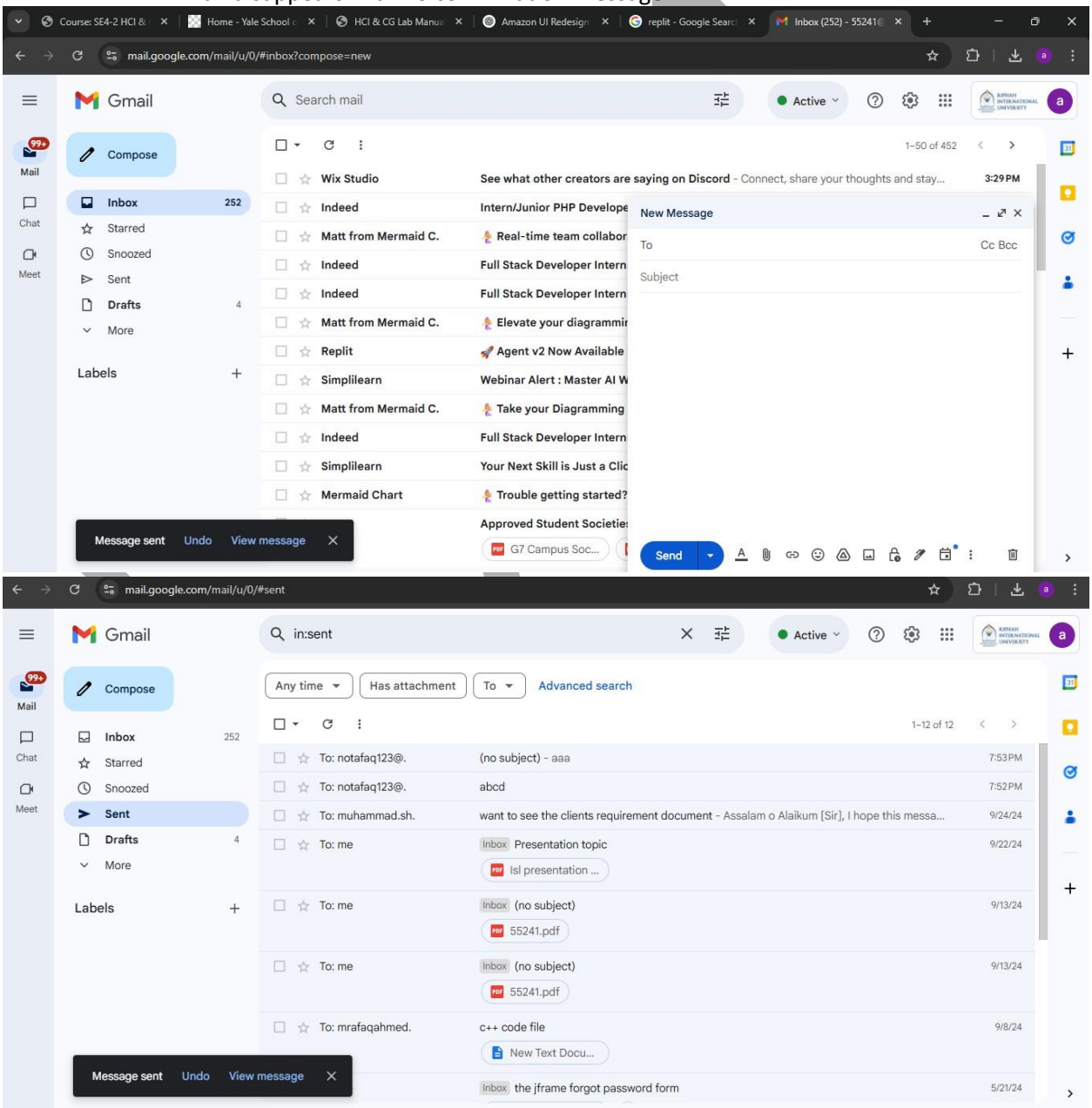
- Over-reliance on heuristics may not account for specific user needs or contexts.
- Some interfaces (e.g., gaming UIs) prioritize engagement over strict usability.
- Security-focused applications might need to prioritize error prevention over user freedom (e.g., limited password reset attempts).

How do cultural differences impact the application of these heuristics in global products?

- Language & symbols: Some icons may have different meanings across cultures.
- Reading patterns: Western users read left to right, while others (e.g., Arabic speakers) read right to left.
- Error messaging: Direct vs. indirect communication styles affect how errors should be presented.
- Color psychology: Red may signal danger in one culture but luck in another (e.g., China).

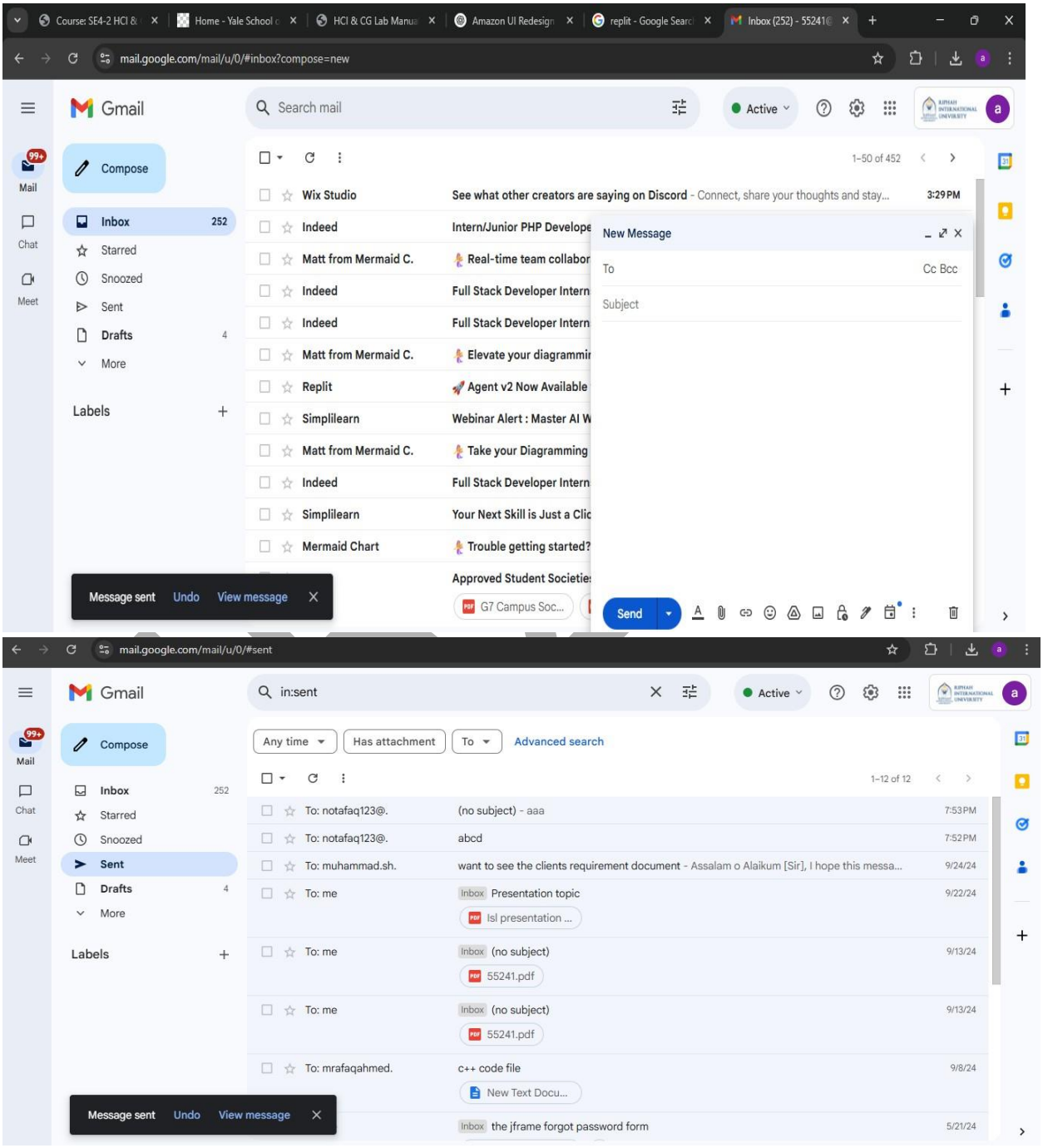
Example: Gmail's Email Sending Process Poor Implementation (Missing Visibility of System Status)

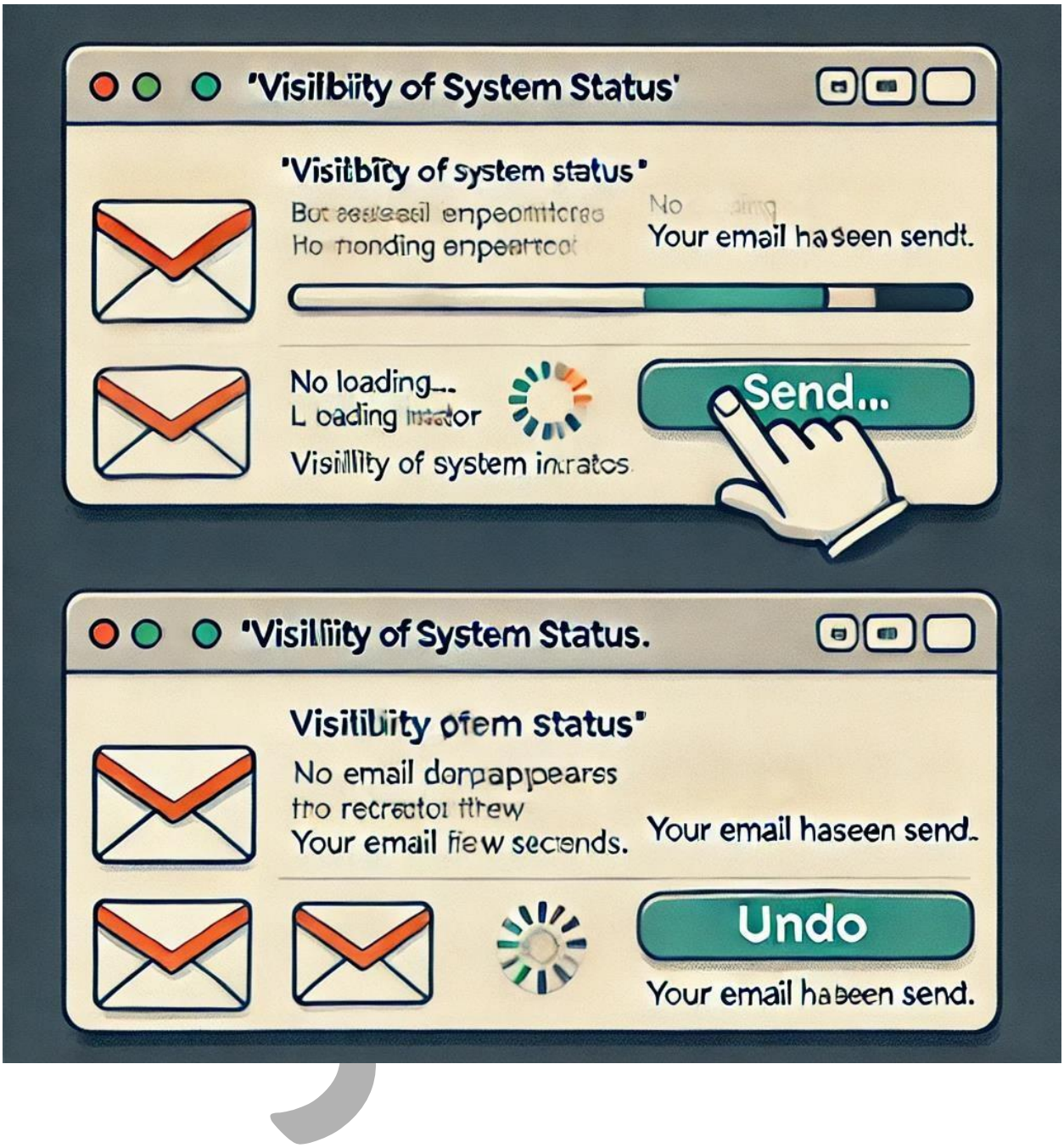
- User clicks "Send", but there is no feedback.
- No loading indicator, leaving the user unsure if the email is being sent.
- Email disappears with no confirmation message.



Good Implementation (Following Visibility of System Status)

- A "Sending..." message appears immediately.
- A green confirmation appears: "Your email has been sent."
- An Undo option is available to retract the email for a few seconds.




'Visilbiity of System Status'
Your email ha seen sendt.
No loading...
Send...
'Visilliity of System Status.
Your email haseen send.
Undo
Your email haseen send.

LAB 8:

Lab Activities

Exercises:

1. Heuristic Evaluation: Evaluate an existing website or application using Nielsen's 10 usability heuristics.
2. User Interview: Conduct an interview with a potential user to understand their needs and goals for a specific task or application.

3. Wireframing: Create a wireframe for a simple application or website.

4. Usability Testing Plan: Develop a plan for conducting usability testing for a specific application or website.

Example: Evaluating Instagram's mobile app.

Heuristic	**Observation**	**Severity**
1. Visibility of system status	App shows loading spinners and notifications clearly.	1 (Good)
2. Match between system and real world	Uses intuitive icons and familiar terms (like "Home," "Search").	1
3. User control and freedom	Back button exists, but accidental actions (like deleting) are hard to undo.	3
4. Consistency and standards	Follows standard UI patterns across iOS/Android.	1
5. Error prevention	Some error prevention exists (e.g., warning before leaving unsaved post).	2
6. Recognition rather than recall	Tabs and icons reduce memory load.	1
7. Flexibility and efficiency of use	Lacks shortcuts or gestures for power users.	3
8. Aesthetic and minimalist design	Clean UI, minimal distractions.	1
9. Help users recognize, diagnose, recover	Error messages are vague (e.g., "Something went wrong").	4 (High)
10. Help and documentation	No direct help section in app.	3

2. User Interview

Objective: Understand user needs and goals for a mobile banking app.

Sample Questions:

- Can you describe how you usually transfer money using a mobile app?
- What do you find difficult or annoying about your current app?
- What features do you wish your banking app had?
- How often do you use it, and for what specific tasks?
- Can you recall a recent frustrating experience with the app?

User Summary:

- Name: Ahmed, 25 years old, Student
- Needs: Quick bill payments, simple navigation
- Goals: Check balance quickly, save frequent transactions
- Pain Points: App is slow, too many login steps, no dark mode

3. Wireframing

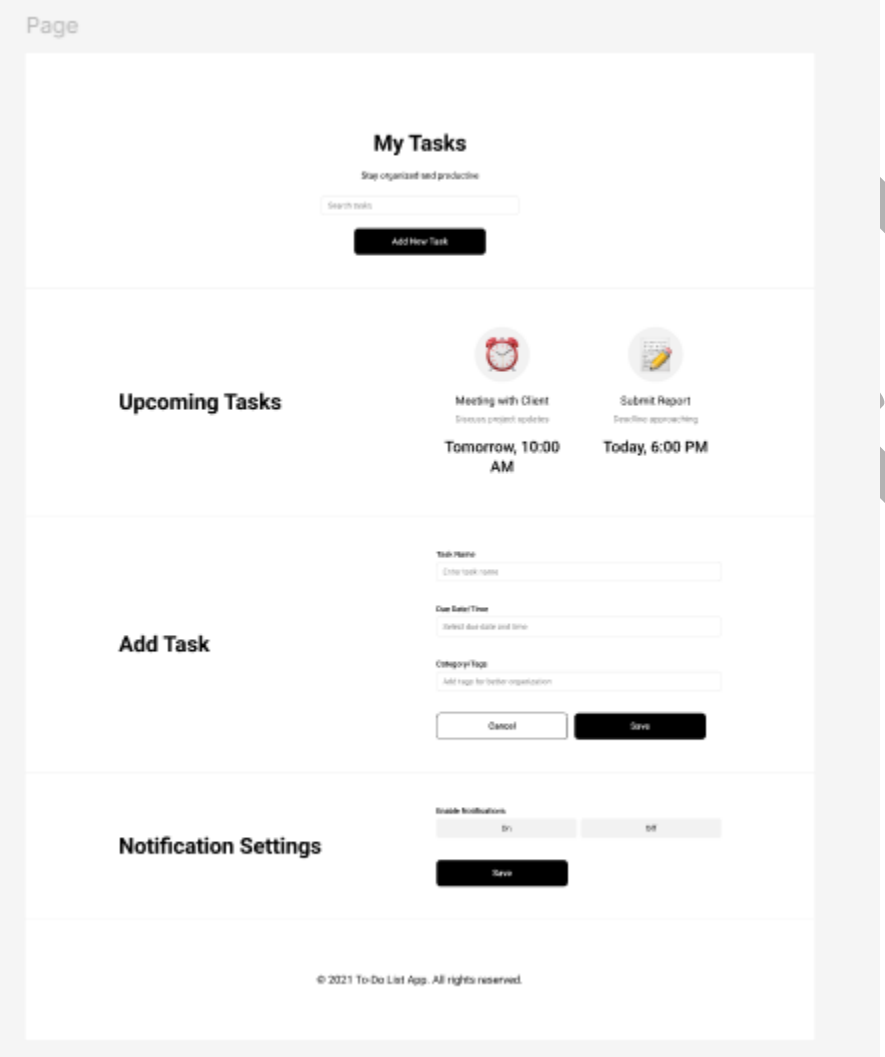
Task: Create a wireframe for a To-Do List App.

Suggested Screens:

1. Home/Dashboard
    - List of tasks
    - Add new task button
2. Add Task
    - Task name
    - Due date/time
    - Category/tags
    - Save/cancel
3. Settings
    - Dark mode toggle
    - Notification settings

LINK OF FIGMA:

https://www.figma.com/design/dLD4tvUNRsxqct9J0uw8Go/Untitled?node-id=0-1&t=LLeyWSHUImnimvfK-1



4. Usability Testing Plan

App: E-commerce website for clothing

Goals:

- Identify issues in the checkout process
- Test ease of navigation

Participants:

- 5 users aged 18–35
- Experience with online shopping

Tasks:

1. Find a specific item (e.g., black hoodie)
2. Add it to cart
3. Proceed to checkout and simulate purchase

Metrics:

- Time on task
- Task success rate
- User satisfaction (post-task survey)
- Errors or hesitations observed

Method:

- In-person or remote test
- Screen recording
- Pre/post-interview with users

LAB 9:

Lab Activities

Usability Evaluation of Facebook Using Nielsen's Heuristics

1. Nielsen's 10 Usability Heuristics & Relevance to Facebook

Nielsen's usability heuristics are widely used to evaluate interactive systems like Facebook. Below is a brief explanation of each heuristic and its relevance:

1. Visibility of System Status – Users should always know what's happening (e.g., loading indicators, confirmation messages).
2. Match Between System and Real World – The interface should use familiar language (e.g., "Add Friend" instead of technical terms).
3. User Control and Freedom – Users should easily undo actions (e.g., cancel a post, unfriend someone).
4. Consistency and Standards – Buttons and layouts should follow conventions (e.g., "Like" button placement).
5. Error Prevention – The system should prevent mistakes (e.g., confirming before deleting a post).
6. Recognition Rather Than Recall – Minimize memory load (e.g., showing recent searches).
7. Flexibility and Efficiency of Use – Shortcuts for power users (e.g., keyboard shortcuts, quick reactions).
8. Aesthetic and Minimalist Design – Avoid unnecessary clutter (e.g., streamlined news feed).
9. Help Users Recognize, Diagnose, and Recover from Errors – Clear error messages (e.g., "Password incorrect").
10. Help and Documentation – Provide guidance when needed (e.g., Facebook Help Center).

2. Heuristic Evaluation of Facebook Features

Selected Features & Violations

Feature	Heuristic Violated	Issue Description	Severity
Add Friend	#5 (Error Prevention)	No confirmation before sending a request; accidental clicks possible.	Major
Change Password	#9 (Error Recovery)	Weak password error messages are vague ("Choose a stronger password").	Minor
Upload Cover Photo	#1 (Visibility)	No progress bar during upload; users don't know if it's processing.	Major
Search	#6 (Recognition)	Recent searches disappear after logout; users must recall names.	Minor
Update Post	#3 (User Control)	No "Edit History" visible after editing; users can't track changes.	Critical

3. Structured Usability Questionnaire (Sample)

Based on a 5-point Likert scale (1=Strongly Disagree, 5=Strongly Agree):

1. "I can easily undo accidental actions (e.g., cancel a friend request)."
2. "Error messages help me fix issues quickly."
3. "The search feature shows my recent searches when I need them."
4. "Uploading media (photos/videos) has clear progress feedback."
5. "The interface follows familiar design conventions."

4. User Feedback Analysis

- Quantitative: 60% of users reported frustration with accidental friend requests (supports Heuristic #5 violation).
- Qualitative: Users mentioned, "I wish Facebook showed if my cover photo uploaded successfully." (Supports Heuristic #1 issue).

5. Proposed UI/UX Improvements

Feature	Current Issue	Suggested Improvement
Add Friend	No confirmation	Add a pop-up: "Send friend request to [Name]?"
Change Password	Vague error msg	Show password strength meter with tips.
Upload Cover Photo	No progress bar	Display upload % and success notification.
Search	Forgets history	Sync recent searches across devices.
Update Post	No edit history	Add "View Edit History" option.

6. Theory vs. Practice

- **Theory (Error Prevention):** Systems should confirm critical actions.
- **Practice:** Facebook allows accidental friend requests without confirmation.
- **Observation:** Users often send unintended requests, leading to awkward social situations.

7. Limitations of Heuristic Evaluation

- **Subjective:** Different evaluators may rate severity differently.
- **No Real User Data:** Heuristics don't replace actual user testing.
- **Static Analysis:** Doesn't account for dynamic user behavior.

Complementary Methods:

- **User Testing:** Observe real interactions.
- **A/B Testing:** Compare two UI versions.
- **Analytics:** Track click patterns and drop-offs.

Facebook Usability Evaluation Questionnaire

(Based on Nielsen's 10 Heuristics)

Instructions for Participants:

- Perform the following tasks on Facebook, then answer the questions.
- Use the provided scales (qualitative or Likert) to rate your experience.

Section 1: General Usability (Nielsen's Heuristics)

1. Visibility of System Status

- When performing actions (e.g., sending a friend request, uploading a photo), does Facebook provide clear feedback (e.g., loading indicators, success messages)?
    - ☐ Always

- ☐ Sometimes
- ☐ Never

2. Match Between System and Real World

- Are terms like "Timeline," "Cover Photo," and "Friend Request" easy to understand?
  - ☐ Very clear
  - ☐ Somewhat clear
  - ☐ Confusing

3. User Control and Freedom

- If you make a mistake (e.g., accidental friend request/post), is it easy to undo?
  - ☐ Very easy
  - ☐ Somewhat easy
  - ☐ Difficult

4. Consistency and Standards

- Do buttons (e.g., "Add Friend," "Update Post") behave the same way across Facebook?
  - ☐ Always
  - ☐ Sometimes
  - ☐ Never

5. Error Prevention

- Does Facebook warn you before irreversible actions (e.g., deleting a post)?
  - ☐ Always
  - ☐ Sometimes
  - ☐ Never

6. Recognition Rather Than Recall

- When searching, does Facebook show recent/searched profiles to help you?
  - ☐ Very helpful
  - ☐ Somewhat helpful

- ☐ Not helpful

7. Flexibility and Efficiency

- Are there shortcuts (e.g., keyboard shortcuts) to complete tasks faster?
  - ☐ Very efficient
  - ☐ Somewhat efficient
  - ☐ Not efficient

8. Aesthetic and Minimalist Design

- Is the interface cluttered or clean (e.g., when uploading a cover photo)?
  - ☐ Clean and minimal
  - ☐ Somewhat cluttered
  - ☐ Very cluttered

9. Error Recovery

- If an error occurs (e.g., failed upload), does Facebook explain how to fix it?
  - ☐ Very clear
  - ☐ Somewhat clear
  - ☐ Unclear

10. Help and Documentation

- Is it easy to find help (e.g., FAQs, Support Center)?
  - ☐ Very easy
  - ☐ Somewhat easy
  - ☐ Difficult

Section 2: Feature-Specific Usability

(Rate on a scale of 1–5, where 1 = Very Difficult/Poor, 5 = Very Easy/Excellent)

Feature	1	2	3	4	5
11. Adding a friend	☐	☐	☐	☐	☐
12. Changing password	☐	☐	☐	☐	☐
13. Uploading a cover photo	☐	☐	☐	☐	☐
14. Accuracy of search results	☐	☐	☐	☐	☐
15. Editing/deleting a post	☐	☐	☐	☐	☐

Section 3: Open-Ended Feedback

16. What is the most frustrating usability issue you've faced on Facebook?

- *[Text box]*

17. Suggest one improvement to make Facebook easier to use.

- *[Text box]*

LAB 10:

Lab Activities

TASK 1: PERSONA CREATION (INDIVIDUAL WORK)

Scenario: You are designing a food delivery app for a diverse user base. Create two distinct personas representing different user types. Steps:

1. Conduct Research (Hypothetical or Real Data)

- Identify target users (e.g., students, working professionals, elderly).
- Note down key behaviors, preferences, and pain points.

2. Define Persona Attributes

- Give each persona a name, age, job, and background.
- Include goals, frustrations, and tech usage

Code for restaurant app

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>FoodExpress - Food Delivery App</title>
  <script src="https://cdn.tailwindcss.com"></script>
  <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.4.0/css/all.min.css">
  <style>
    .active-tab {
      color: #FF6B6B;
      border-bottom: 2px solid #FF6B6B;
    }
    .food-card:hover {
      transform: translateY(-5px);
      box-shadow: 0 10px 20px rgba(0,0,0,0.1);
    }
```

```
        .cart-item:hover {
            background-color: #f8f9fa;
        }
        .slide-in {
            animation: slideIn 0.3s forwards;
        }
        @keyframes slideIn {
            from { transform: translateX(100%); }
            to { transform: translateX(0); }
        }
    </style>
</head>
<body class="bg-gray-50 font-sans">
    <!-- Main App Container -->
    <div class="max-w-md mx-auto bg-white min-h-screen relative overflow-hidden">
        <!-- Header -->
        <header class="bg-white shadow-sm p-4 sticky top-0 z-10">
            <div class="flex justify-between items-center">
                <div class="flex items-center space-x-2">
                    <i class="fas fa-map-marker-alt text-red-400"></i>
                    <div>
                        <p class="text-xs text-gray-500">Delivery to</p>
                        <p class="font-medium">123 Food Street</p>
                    </div>
                </div>
                <div class="relative">
                    <div class="w-10 h-10 rounded-full bg-gray-200 flex items-center justify-center">
                        <i class="fas fa-user text-gray-600"></i>
                    </div>
```

```
            <span class="absolute -top-1 -right-1 bg-red-500 text-white text-xs rounded-full w-5 h-5
flex items-center justify-center">3</span>
          </div>
        </div>
      </header>

      <!-- Main Content -->
      <main class="pb-20">
        <!-- Search Bar -->
        <div class="px-4 mt-2">
          <div class="relative">
            <input type="text" placeholder="Search for restaurants or food..."
                class="w-full p-3 pl-10 rounded-lg border border-gray-200 focus:outline-none
focus:ring-2 focus:ring-red-100">
            <i class="fas fa-search absolute left-3 top-3.5 text-gray-400"></i>
            <button class="absolute right-3 top-2.5">
              <i class="fas fa-sliders-h text-gray-400"></i>
            </button>
          </div>
        </div>

        <!-- Categories -->
        <div class="mt-6 px-4">
          <h2 class="text-lg font-semibold mb-3">Categories</h2>
          <div class="flex space-x-4 overflow-x-auto pb-2">
            <div class="flex-none text-center">
              <div class="w-16 h-16 rounded-full bg-red-50 flex items-center justify-center mx-auto">
                <i class="fas fa-hamburger text-red-400 text-xl"></i>
              </div>
              <p class="text-xs mt-1">Burgers</p>
```

```
            </div>
            <div class="flex-none text-center">
              <div class="w-16 h-16 rounded-full bg-yellow-50 flex items-center justify-center mx-
auto">
                <i class="fas fa-pizza-slice text-yellow-400 text-xl"></i>
              </div>
              <p class="text-xs mt-1">Pizza</p>
            </div>
            <div class="flex-none text-center">
              <div class="w-16 h-16 rounded-full bg-green-50 flex items-center justify-center mx-
auto">
                <i class="fas fa-drumstick-bite text-green-400 text-xl"></i>
              </div>
              <p class="text-xs mt-1">Chicken</p>
            </div>
            <div class="flex-none text-center">
              <div class="w-16 h-16 rounded-full bg-blue-50 flex items-center justify-center mx-
auto">
                <i class="fas fa-ice-cream text-blue-400 text-xl"></i>
              </div>
              <p class="text-xs mt-1">Desserts</p>
            </div>
            <div class="flex-none text-center">
              <div class="w-16 h-16 rounded-full bg-purple-50 flex items-center justify-center mx-
auto">
                <i class="fas fa-coffee text-purple-400 text-xl"></i>
              </div>
              <p class="text-xs mt-1">Drinks</p>
            </div>
          </div>
        </div>
```

```
<!-- Popular Restaurants -->
<div class="mt-6 px-4">
  <div class="flex justify-between items-center mb-3">
    <h2 class="text-lg font-semibold">Popular Restaurants</h2>
    <button class="text-red-400 text-sm">See all</button>
  </div>

  <!-- Restaurant Cards -->
  <div class="space-y-4">
    <div class="border border-gray-100 rounded-xl p-3 shadow-sm">
      <div class="flex">
        <img src="https://images.unsplash.com/photo-1555396273-367ea4eb4db5?ixlib=rb-1.2.1&auto=format&fit=crop&w=500&q=60"
          alt="Burger King" class="w-20 h-20 rounded-lg object-cover">
        <div class="ml-3 flex-1">
          <div class="flex justify-between">
            <h3 class="font-medium">Burger King</h3>
            <div class="flex items-center">
              <i class="fas fa-star text-yellow-400 text-xs"></i>
              <span class="text-xs ml-1">4.5</span>
            </div>
          </div>
          <p class="text-gray-500 text-xs mt-1">American • Burgers • Fast Food</p>
          <div class="flex items-center mt-2">
            <i class="fas fa-clock text-gray-400 text-xs"></i>
            <span class="text-xs text-gray-500 ml-1">15-25 min</span>
            <span class="text-xs text-gray-500 mx-2">•</span>
            <i class="fas fa-biking text-gray-400 text-xs"></i>
            <span class="text-xs text-gray-500 ml-1">$1.99</span>
```

```
                    </div>
                  </div>
                </div>
                <button class="mt-3 w-full py-2 bg-red-50 text-red-400 rounded-lg text-sm font-
medium">
                  View Menu
                </button>
              </div>

              <div class="border border-gray-100 rounded-xl p-3 shadow-sm">
                <div class="flex">
                  <img src="https://images.unsplash.com/photo-1513104890138-
7c749659a591?ixlib=rb-1.2.1&auto=format&fit=crop&w=500&q=60"
                    alt="Pizza Hut" class="w-20 h-20 rounded-lg object-cover">
                  <div class="ml-3 flex-1">
                    <div class="flex justify-between">
                      <h3 class="font-medium">Pizza Hut</h3>
                      <div class="flex items-center">
                        <i class="fas fa-star text-yellow-400 text-xs"></i>
                        <span class="text-xs ml-1">4.3</span>
                      </div>
                    </div>
                    <p class="text-gray-500 text-xs mt-1">Italian • Pizza • Pasta</p>
                    <div class="flex items-center mt-2">
                      <i class="fas fa-clock text-gray-400 text-xs"></i>
                      <span class="text-xs text-gray-500 ml-1">20-30 min</span>
                      <span class="text-xs text-gray-500 mx-2">•</span>
                      <i class="fas fa-biking text-gray-400 text-xs"></i>
                      <span class="text-xs text-gray-500 ml-1">$2.99</span>
                    </div>
```

```
                </div>
              </div>
              <button class="mt-3 w-full py-2 bg-red-50 text-red-400 rounded-lg text-sm font-medium">
                View Menu
              </button>
            </div>
          </div>
        </div>

        <!-- Menu Items (shown when viewing a restaurant) -->
        <div id="menuSection" class="hidden slide-in absolute top-0 left-0 w-full bg-white min-h-screen z-20">
          <div class="p-4">
            <div class="flex items-center mb-4">
              <button id="backButton" class="mr-3">
                <i class="fas fa-arrow-left text-gray-600"></i>
              </button>
              <h2 class="text-xl font-semibold">Burger King Menu</h2>
            </div>

            <div class="flex border-b mb-4">
              <button class="tab-btn px-4 py-2 font-medium active-tab">Burgers</button>
              <button class="tab-btn px-4 py-2 font-medium text-gray-500">Sides</button>
              <button class="tab-btn px-4 py-2 font-medium text-gray-500">Drinks</button>
            </div>

            <div class="space-y-4">
              <div class="food-card border border-gray-100 rounded-xl p-3 shadow-sm transition duration-300">
                <div class="flex">
```

```
                    <img src="https://images.unsplash.com/photo-1568901346375-
23c9450c58cd?ixlib=rb-1.2.1&auto=format&fit=crop&w=500&q=60"
                      alt="Whopper" class="w-20 h-20 rounded-lg object-cover">
                    <div class="ml-3 flex-1">
                      <h3 class="font-medium">Whopper</h3>
                      <p class="text-gray-500 text-xs mt-1">Flame-grilled beef patty with juicy
tomatoes, fresh lettuce, creamy mayonnaise</p>
                      <div class="flex justify-between items-center mt-2">
                        <span class="font-medium text-red-500">$5.99</span>
                        <button class="add-to-cart bg-red-400 text-white rounded-full w-8 h-8 flex
items-center justify-center">
                          <i class="fas fa-plus"></i>
                        </button>
                      </div>
                    </div>
                  </div>
                </div>

                <div class="food-card border border-gray-100 rounded-xl p-3 shadow-sm transition
duration-300">
                  <div class="flex">
                    <img src="https://images.unsplash.com/photo-1571091718767-
18b5b1457add?ixlib=rb-1.2.1&auto=format&fit=crop&w=500&q=60"
                      alt="Cheeseburger" class="w-20 h-20 rounded-lg object-cover">
                    <div class="ml-3 flex-1">
                      <h3 class="font-medium">Cheeseburger</h3>
                      <p class="text-gray-500 text-xs mt-1">100% beef patty with melted cheese,
pickles, mustard and ketchup</p>
                      <div class="flex justify-between items-center mt-2">
                        <span class="font-medium text-red-500">$3.99</span>
                        <button class="add-to-cart bg-red-400 text-white rounded-full w-8 h-8 flex
items-center justify-center">
```

```
                        <i class="fas fa-plus"></i>
                      </button>
                    </div>
                  </div>
                </div>
              </div>
            </div>

        <!-- Cart Section -->
        <div id="cartSection" class="hidden slide-in absolute top-0 left-0 w-full bg-white min-h-screen
z-20">
          <div class="p-4">
            <div class="flex items-center mb-4">
              <button id="backFromCartButton" class="mr-3">
                <i class="fas fa-arrow-left text-gray-600"></i>
              </button>
              <h2 class="text-xl font-semibold">Your Cart</h2>
            </div>

            <div class="space-y-3 mb-4">
              <div class="cart-item border-b pb-3 transition duration-200">
                <div class="flex justify-between">
                  <div>
                    <h3 class="font-medium">Whopper</h3>
                    <p class="text-gray-500 text-sm">Burger King</p>
                  </div>
                  <span class="font-medium">$5.99</span>
                </div>
```

```
    <div class="flex items-center justify-between mt-2">
      <div class="flex items-center border rounded-lg">
        <button class="px-3 py-1 text-gray-500">-</button>
        <span class="px-2">1</span>
        <button class="px-3 py-1 text-gray-500">+</button>
      </div>
      <button class="text-red-400">
        <i class="fas fa-trash"></i>
      </button>
    </div>
  </div>

  <div class="cart-item border-b pb-3 transition duration-200">
    <div class="flex justify-between">
      <div>
        <h3 class="font-medium">Cheeseburger</h3>
        <p class="text-gray-500 text-sm">Burger King</p>
      </div>
      <span class="font-medium">$3.99</span>
    </div>
    <div class="flex items-center justify-between mt-2">
      <div class="flex items-center border rounded-lg">
        <button class="px-3 py-1 text-gray-500">-</button>
        <span class="px-2">2</span>
        <button class="px-3 py-1 text-gray-500">+</button>
      </div>
      <button class="text-red-400">
        <i class="fas fa-trash"></i>
      </button>
    </div>
```

```
            </div>
          </div>

          <div class="bg-gray-50 p-3 rounded-lg mb-4">
            <div class="flex justify-between mb-2">
              <span class="text-gray-500">Subtotal</span>
              <span>$13.97</span>
            </div>
            <div class="flex justify-between mb-2">
              <span class="text-gray-500">Delivery Fee</span>
              <span>$1.99</span>
            </div>
            <div class="flex justify-between font-medium">
              <span>Total</span>
              <span>$15.96</span>
            </div>
          </div>

          <button id="checkoutButton" class="w-full py-3 bg-red-400 text-white rounded-lg font-
medium">
            Proceed to Checkout
          </button>
        </div>
      </div>

      <!-- Checkout Section -->
      <div id="checkoutSection" class="hidden slide-in absolute top-0 left-0 w-full bg-white min-h-
screen z-20">
        <div class="p-4">
          <div class="flex items-center mb-4">
```

```
            <button id="backFromCheckoutButton" class="mr-3">
              <i class="fas fa-arrow-left text-gray-600"></i>
            </button>
            <h2 class="text-xl font-semibold">Checkout</h2>
          </div>

          <div class="mb-6">
            <h3 class="font-medium mb-2">Delivery Address</h3>
            <div class="border border-gray-200 rounded-lg p-3">
              <div class="flex items-start">
                <i class="fas fa-map-marker-alt text-red-400 mt-1 mr-2"></i>
                <div>
                  <p class="font-medium">Home</p>
                  <p class="text-gray-500 text-sm">123 Food Street, Apt 4B, New York, NY
10001</p>
                </div>
              </div>
            </div>
          </div>

          <div class="mb-6">
            <h3 class="font-medium mb-2">Payment Method</h3>
            <div class="space-y-2">
              <div class="border border-gray-200 rounded-lg p-3 flex items-center">
                <i class="far fa-credit-card text-gray-400 mr-3"></i>
                <span>Credit Card</span>
              </div>
              <div class="border border-gray-200 rounded-lg p-3 flex items-center">
                <i class="fab fa-paypal text-blue-400 mr-3"></i>
                <span>PayPal</span>
```

```
                </div>
                <div class="border border-red-200 bg-red-50 rounded-lg p-3 flex items-center">
                  <i class="fas fa-money-bill-wave text-red-400 mr-3"></i>
                  <span>Cash on Delivery</span>
                </div>
              </div>
            </div>

            <div class="bg-gray-50 p-3 rounded-lg mb-4">
              <div class="flex justify-between mb-2">
                <span class="text-gray-500">Subtotal</span>
                <span>$13.97</span>
              </div>
              <div class="flex justify-between mb-2">
                <span class="text-gray-500">Delivery Fee</span>
                <span>$1.99</span>
              </div>
              <div class="flex justify-between font-medium">
                <span>Total</span>
                <span>$15.96</span>
              </div>
            </div>

            <button id="placeOrderButton" class="w-full py-3 bg-red-400 text-white rounded-lg font-
medium">
              Place Order
            </button>
          </div>
        </div>
```

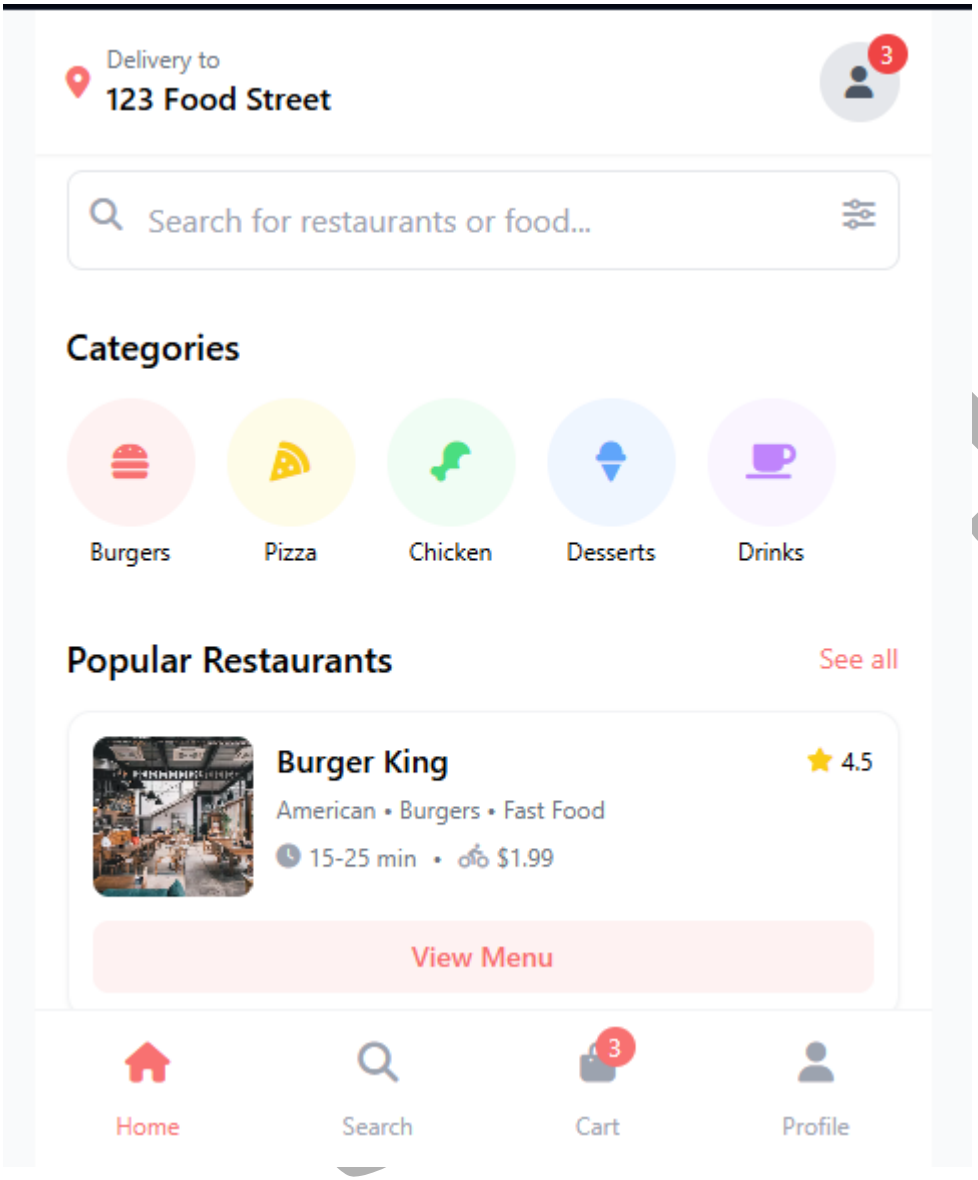
```
        <!-- Order Confirmation -->
        <div id="confirmationSection" class="hidden slide-in absolute top-0 left-0 w-full bg-white min-h-screen z-20">
          <div class="p-4 flex flex-col items-center justify-center h-full">
            <div class="w-24 h-24 bg-green-100 rounded-full flex items-center justify-center mb-4">
              <i class="fas fa-check text-green-500 text-4xl"></i>
            </div>
            <h2 class="text-2xl font-semibold mb-2">Order Confirmed!</h2>
            <p class="text-gray-500 text-center mb-6">Your order has been placed successfully. Estimated delivery time is 25 minutes.</p>
            <div class="w-full border border-gray-200 rounded-lg p-4 mb-6">
              <div class="flex justify-between mb-3">
                <span class="text-gray-500">Order ID</span>
                <span class="font-medium">#123456</span>
              </div>
              <div class="flex justify-between mb-3">
                <span class="text-gray-500">Total</span>
                <span class="font-medium">$15.96</span>
              </div>
              <div class="flex justify-between">
                <span class="text-gray-500">Payment</span>
                <span class="font-medium">Cash on Delivery</span>
              </div>
            </div>
            <button id="backToHomeButton" class="w-full py-3 bg-red-400 text-white rounded-lg font-medium">
              Back to Home
            </button>
          </div>
        </div>
      </main>
```

```
    <!-- Bottom Navigation -->
    <nav class="fixed bottom-0 w-full max-w-md bg-white border-t border-gray-100 flex justify-around py-3 z-10">
      <button class="text-center text-red-400">
        <i class="fas fa-home block text-xl mb-1"></i>
        <span class="text-xs">Home</span>
      </button>
      <button class="text-center text-gray-400">
        <i class="fas fa-search block text-xl mb-1"></i>
        <span class="text-xs">Search</span>
      </button>
      <button id="cartButton" class="relative text-center text-gray-400">
        <i class="fas fa-shopping-bag block text-xl mb-1"></i>
        <span class="text-xs">Cart</span>
        <span class="absolute -top-1 -right-2 bg-red-400 text-white text-xs rounded-full w-5 h-5 flex items-center justify-center">3</span>
      </button>
      <button class="text-center text-gray-400">
        <i class="fas fa-user block text-xl mb-1"></i>
        <span class="text-xs">Profile</span>
      </button>
    </nav>
  </div>

  <script>
    // View Menu Button
    document.querySelectorAll('button:contains("View Menu")').forEach(button => {
      button.addEventListener('click', () => {
        document.getElementById('menuSection').classList.remove('hidden');
```

```
    });
});

// Back Button from Menu
document.getElementById('backButton').addEventListener('click', () => {
    document.getElementById('menuSection').classList.add('hidden');
});

// Tab Switching
document.querySelectorAll('.tab-btn').forEach(button => {
    button.addEventListener('click', () => {
        document.querySelectorAll('.tab-btn').forEach(btn => {
            btn.classList.remove('active-tab');
            btn.classList.add('text-gray-500');
        });
        button.classList.add('active-tab');
        button.classList.remove('text-gray-500');
    });
});

// Add to Cart
document.querySelectorAll('.add-to-cart').forEach(button => {
    button.addEventListener('click', () => {
        // In a real app, this would add the item to the cart
        alert('Item added to cart!');
    });
});

// Cart Button
document.getElementById('cartButton').addEventListener('click', () => {
```

```
    document.getElementById('cartSection').classList.remove('hidden');
  });

  // Back Button from Cart
  document.getElementById('backFromCartButton').addEventListener('click', () => {
    document.getElementById('cartSection').classList.add('hidden');
  });

  // Checkout Button
  document.getElementById('checkoutButton').addEventListener('click', () => {
    document.getElementById('checkoutSection').classList.remove('hidden');
  });

  // Back Button from Checkout
  document.getElementById('backFromCheckoutButton').addEventListener('click', () => {
    document.getElementById('checkoutSection').classList.add('hidden');
  });

  // Place Order Button
  document.getElementById('placeOrderButton').addEventListener('click', () => {
    document.getElementById('confirmationSection').classList.remove('hidden');
  });

  // Back to Home Button
  document.getElementById('backToHomeButton').addEventListener('click', () => {
    document.getElementById('confirmationSection').classList.add('hidden');
    document.getElementById('checkoutSection').classList.add('hidden');
    document.getElementById('cartSection').classList.add('hidden');
  });
</script>
```

```
</body>
</html>
```


Delivery to
123 Food Street
3
Search for restaurants or food...
Categories
Burgers
Pizza
Chicken
Desserts
Drinks
Popular Restaurants
See all
Burger King
4.5
American • Burgers • Fast Food
15-25 min • $1.99
View Menu
Home
Search
3
Cart
Profile

Lab Activity 2:

Task Brief: Design a simple student attendance mobile app.

It should allow a teacher to:

- Log in
- View class list
- Mark present/absent
- View attendance summary

Code for task

```
<!DOCTYPE html>

<html lang="en">

<head>

  <meta charset="UTF-8">

  <meta name="viewport" content="width=device-width, initial-scale=1.0">

  <title>EduTrack - Attendance App</title>

  <script src="https://cdn.tailwindcss.com"></script>

  <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-
awesome/6.4.0/css/all.min.css">

  <style>

    .active-tab {

      color: #3B82F6;

      border-bottom: 2px solid #3B82F6;

    }

    .slide-in {

      animation: slideIn 0.3s forwards;

    }

    @keyframes slideIn {

      from { transform: translateX(100%); }

      to { transform: translateX(0); }

    }

    .status-present {

      background-color: #D1FAE5;
```

```
            color: #065F46;
        }
        .status-absent {
            background-color: #FEE2E2;
            color: #B91C1C;
        }
    </style>
</head>
<body class="bg-gray-50 font-sans">
    <!-- Main App Container -->
    <div class="max-w-md mx-auto bg-white min-h-screen relative overflow-hidden">
        <!-- Login Screen -->
        <div id="loginScreen" class="min-h-screen flex flex-col justify-center px-6">
            <div class="text-center mb-8">
                <div class="w-20 h-20 bg-blue-100 rounded-2xl flex items-center justify-center mx-auto mb-4">
                    <i class="fas fa-chalkboard-teacher text-blue-500 text-3xl"></i>
                </div>
                <h1 class="text-2xl font-bold text-gray-800">EduTrack</h1>
                <p class="text-gray-500 mt-1">Teacher Attendance Portal</p>
            </div>

            <form id="loginForm" class="space-y-4">
                <div>
                    <label for="email" class="block text-sm font-medium text-gray-700 mb-1">Email</label>
                    <input type="email" id="email" required
                        class="w-full px-4 py-2 border border-gray-300 rounded-lg focus:ring-2 focus:ring-blue-500 focus:border-blue-500">
                </div>

                <div>
```

```
            <label for="password" class="block text-sm font-medium text-gray-700 mb-
1">Password</label>
            <input type="password" id="password" required
                class="w-full px-4 py-2 border border-gray-300 rounded-lg focus:ring-2 focus:ring-blue-
500 focus:border-blue-500">
        </div>

        <button type="submit"
            class="w-full py-3 bg-blue-500 text-white font-medium rounded-lg hover:bg-blue-600
transition duration-200">
            Sign In
        </button>
    </form>

    <div class="mt-6 text-center">
        <a href="#" class="text-sm text-blue-500 hover:underline">Forgot password?</a>
    </div>
</div>

<!-- Main App (hidden initially) -->
<div id="appScreen" class="hidden min-h-screen">
    <!-- Header -->
    <header class="bg-white shadow-sm p-4 sticky top-0 z-10">
        <div class="flex justify-between items-center">
            <div>
                <h1 class="text-xl font-bold text-gray-800">My Classes</h1>
                <p class="text-xs text-gray-500">Today, <span id="currentDate">Nov 15, 2023</span></p>
            </div>
            <div class="flex items-center space-x-3">
                <button class="w-10 h-10 rounded-full bg-gray-100 flex items-center justify-center">
                    <i class="fas fa-bell text-gray-600"></i>
```

```
            </button>
            <div class="w-10 h-10 rounded-full bg-blue-100 flex items-center justify-center">
              <i class="fas fa-user text-blue-500"></i>
            </div>
          </div>
        </div>
      </header>

      <!-- Main Content -->
      <main class="pb-20">
        <!-- Class Selection -->
        <div class="px-4 mt-4">
          <div class="relative">
            <select id="classSelect" class="w-full px-4 py-3 border border-gray-300 rounded-lg
appearance-none bg-white">
              <option>Grade 10 - Mathematics</option>
              <option>Grade 11 - Physics</option>
              <option>Grade 9 - Science</option>
            </select>
            <i class="fas fa-chevron-down absolute right-4 top-4 text-gray-400 pointer-events-
none"></i>
          </div>
        </div>

        <!-- Attendance Tabs -->
        <div class="mt-6 px-4">
          <div class="flex border-b">
            <button id="attendanceTab" class="tab-btn px-4 py-2 font-medium active-
tab">Attendance</button>
            <button id="summaryTab" class="tab-btn px-4 py-2 font-medium text-gray-
500">Summary</button>
```

```
    </div>
  </div>

  <!-- Attendance List -->
  <div id="attendanceList" class="mt-4 px-4 space-y-3">
    <div class="student-card bg-white border border-gray-200 rounded-lg p-3 shadow-sm">
      <div class="flex items-center justify-between">
        <div class="flex items-center">
          <div class="w-10 h-10 rounded-full bg-blue-100 flex items-center justify-center mr-3">
            <span class="font-medium text-blue-500">JD</span>
          </div>
          <div>
            <h3 class="font-medium">John Doe</h3>
            <p class="text-xs text-gray-500">ID: S1001</p>
          </div>
        </div>
        <div class="flex space-x-2">
          <button class="status-btn present-btn px-3 py-1 rounded-full status-present">
            <i class="fas fa-check mr-1"></i> Present
          </button>
          <button class="status-btn absent-btn px-3 py-1 rounded-full">
            <i class="fas fa-times mr-1"></i> Absent
          </button>
        </div>
      </div>
    </div>

    <div class="student-card bg-white border border-gray-200 rounded-lg p-3 shadow-sm">
      <div class="flex items-center justify-between">
        <div class="flex items-center">
```

```
                <div class="w-10 h-10 rounded-full bg-purple-100 flex items-center justify-center mr-
3">
                  <span class="font-medium text-purple-500">SJ</span>
                </div>
                <div>
                  <h3 class="font-medium">Sarah Johnson</h3>
                  <p class="text-xs text-gray-500">ID: S1002</p>
                </div>
              </div>
              <div class="flex space-x-2">
                <button class="status-btn present-btn px-3 py-1 rounded-full">
                  <i class="fas fa-check mr-1"></i> Present
                </button>
                <button class="status-btn absent-btn px-3 py-1 rounded-full status-absent">
                  <i class="fas fa-times mr-1"></i> Absent
                </button>
              </div>
            </div>
          </div>

          <div class="student-card bg-white border border-gray-200 rounded-lg p-3 shadow-sm">
            <div class="flex items-center justify-between">
              <div class="flex items-center">
                <div class="w-10 h-10 rounded-full bg-green-100 flex items-center justify-center mr-
3">
                  <span class="font-medium text-green-500">MB</span>
                </div>
                <div>
                  <h3 class="font-medium">Michael Brown</h3>
                  <p class="text-xs text-gray-500">ID: S1003</p>
```

```
            </div>
          </div>
          <div class="flex space-x-2">
            <button class="status-btn present-btn px-3 py-1 rounded-full status-present">
              <i class="fas fa-check mr-1"></i> Present
            </button>
            <button class="status-btn absent-btn px-3 py-1 rounded-full">
              <i class="fas fa-times mr-1"></i> Absent
            </button>
          </div>
        </div>
      </div>

      <div class="student-card bg-white border border-gray-200 rounded-lg p-3 shadow-sm">
        <div class="flex items-center justify-between">
          <div class="flex items-center">
            <div class="w-10 h-10 rounded-full bg-yellow-100 flex items-center justify-center mr-
3">
              <span class="font-medium text-yellow-500">EW</span>
            </div>
            <div>
              <h3 class="font-medium">Emma Wilson</h3>
              <p class="text-xs text-gray-500">ID: S1004</p>
            </div>
          </div>
          <div class="flex space-x-2">
            <button class="status-btn present-btn px-3 py-1 rounded-full">
              <i class="fas fa-check mr-1"></i> Present
            </button>
            <button class="status-btn absent-btn px-3 py-1 rounded-full">
```

```
                <i class="fas fa-times mr-1"></i> Absent
              </button>
            </div>
          </div>
        </div>
      </div>

      <!-- Attendance Summary -->
      <div id="attendanceSummary" class="hidden px-4 mt-4">
        <div class="bg-white rounded-lg shadow-sm p-4">
          <h3 class="font-medium mb-3">Today's Attendance Summary</h3>

          <div class="flex justify-between items-center mb-4">
            <div class="text-center">
              <div class="text-3xl font-bold text-green-600">75%</div>
              <div class="text-xs text-gray-500">Attendance Rate</div>
            </div>
            <div class="h-16 w-px bg-gray-200"></div>
            <div class="text-center">
              <div class="text-3xl font-bold text-green-600">15</div>
              <div class="text-xs text-gray-500">Present</div>
            </div>
            <div class="h-16 w-px bg-gray-200"></div>
            <div class="text-center">
              <div class="text-3xl font-bold text-red-600">5</div>
              <div class="text-xs text-gray-500">Absent</div>
            </div>
          </div>

          <div class="mb-4">
```

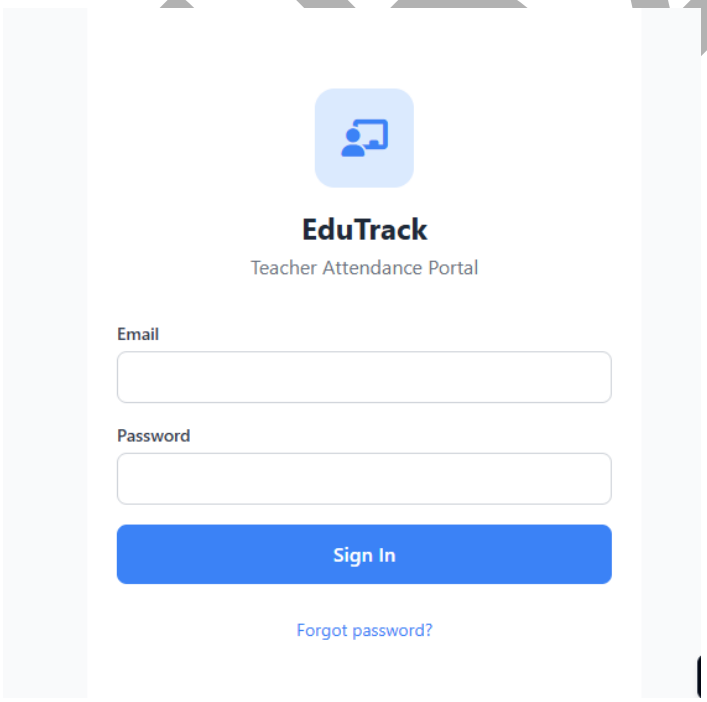
```
    <div class="flex justify-between text-sm mb-1">
      <span>Attendance Trend</span>
      <span>Last 7 days</span>
    </div>
    <div class="h-2 bg-gray-200 rounded-full overflow-hidden">
      <div class="h-full bg-blue-500 rounded-full" style="width: 80%"></div>
    </div>
  </div>

  <div>
    <h4 class="font-medium mb-2">Frequent Absentees</h4>
    <div class="space-y-2">
      <div class="flex items-center justify-between">
        <div class="flex items-center">
          <div class="w-8 h-8 rounded-full bg-red-100 flex items-center justify-center mr-2">
            <span class="text-xs font-medium text-red-500">SJ</span>
          </div>
          <span class="text-sm">Sarah Johnson</span>
        </div>
        <span class="text-sm text-red-500">3 absences</span>
      </div>
      <div class="flex items-center justify-between">
        <div class="flex items-center">
          <div class="w-8 h-8 rounded-full bg-red-100 flex items-center justify-center mr-2">
            <span class="text-xs font-medium text-red-500">DL</span>
          </div>
          <span class="text-sm">David Lee</span>
        </div>
        <span class="text-sm text-red-500">2 absences</span>
      </div>
```

```
            </div>
          </div>
        </div>

        <div class="bg-white rounded-lg shadow-sm p-4 mt-4">
          <h3 class="font-medium mb-3">Export Attendance</h3>
          <div class="flex space-x-3">
            <button class="flex-1 py-2 border border-blue-500 text-blue-500 rounded-lg flex items-center justify-center">
              <i class="fas fa-file-pdf mr-2"></i> PDF
            </button>
            <button class="flex-1 py-2 border border-green-500 text-green-500 rounded-lg flex items-center justify-center">
              <i class="fas fa-file-excel mr-2"></i> Excel
            </button>
            <button class="flex-1 py-2 border border-gray-500 text-gray-500 rounded-lg flex items-center justify-center">
              <i class="fas fa-print mr-2"></i> Print
            </button>
          </div>
        </div>
      </div>
    </main>

    <!-- Bottom Navigation -->
    <nav class="fixed bottom-0 w-full max-w-md bg-white border-t border-gray-100 flex justify-around py-3 z-10">
      <button class="text-center text-blue-500">
        <i class="fas fa-home block text-xl mb-1"></i>
        <span class="text-xs">Classes</span>
      </button>
```

```
            <button class="text-center text-gray-400">
                <i class="fas fa-calendar-alt block text-xl mb-1"></i>
                <span class="text-xs">Calendar</span>
            </button>
            <button class="text-center text-gray-400">
                <i class="fas fa-chart-bar block text-xl mb-1"></i>
                <span class="text-xs">Reports</span>
            </button>
            <button class="text-center text-gray-400">
                <i class="fas fa-cog block text-xl mb-1"></i>
                <span class="text-xs">Settings</span>
            </button>
        </nav>
    </div>
</div>

<script>
    // Set current date
    const today = new Date();
    const options = { month: 'short', day: 'numeric', year: 'numeric' };
    document.getElementById('currentDate').textContent = today.toLocaleDateString('en-US', options);

    // Login Form Submission
    document.getElementById('loginForm').addEventListener('submit', function(e) {
        e.preventDefault();
        // Simulate login
        document.getElementById('loginScreen').classList.add('hidden');
        document.getElementById('appScreen').classList.remove('hidden');
    });
```

```
// Tab Switching
document.getElementById('attendanceTab').addEventListener('click', function() {
    document.getElementById('attendanceList').classList.remove('hidden');
    document.getElementById('attendanceSummary').classList.add('hidden');
    this.classList.add('active-tab');
    this.classList.remove('text-gray-500');
    document.getElementById('summaryTab').classList.remove('active-tab');
    document.getElementById('summaryTab').classList.add('text-gray-500');
});

document.getElementById('summaryTab').addEventListener('click', function() {
    document.getElementById('attendanceList').classList.add('hidden');
    document.getElementById('attendanceSummary').classList.remove('hidden');
    this.classList.add('active-tab');
    this.classList.remove('text-gray-500');
    document.getElementById('attendanceTab').classList.remove('active-tab');
    document.getElementById('attendanceTab').classList.add('text-gray-500');
});

// Attendance Status Buttons
document.querySelectorAll('.status-btn').forEach(button => {
    button.addEventListener('click', function(e) {
        e.preventDefault();
        const parentCard = this.closest('.student-card');

        // Remove all status classes from both buttons in this card
        parentCard.querySelectorAll('.status-btn').forEach(btn => {
            btn.classList.remove('status-present', 'status-absent');
        });
```

```
            // Add appropriate class to clicked button
            if (this.classList.contains('present-btn')) {
                this.classList.add('status-present');
            } else if (this.classList.contains('absent-btn')) {
                this.classList.add('status-absent');
            }
        });
    });

    // Class Select Change
    document.getElementById('classSelect').addEventListener('change', function() {
        // In a real app, this would load the appropriate class data
        console.log('Selected class:', this.value);
    });
  </script>
</body>
</html>
```



Deliverables:

1. Document 2 Design Decisions (with justification) o E.g., Use of tabs vs dropdowns for class sections. 2. Paper Prototype (at least 3 screens) o Sketch screens and UI flow. o Include navigation and key actions. 3. Usability Testing o Ask 1–2 classmates to test your paper prototype. o Record 2 usability findings (what was confusing or smooth?) 4. Short Reflection o How did feedback affect your design? o Would you redesign anything?

1. Document 2 Design Decisions (with justification)

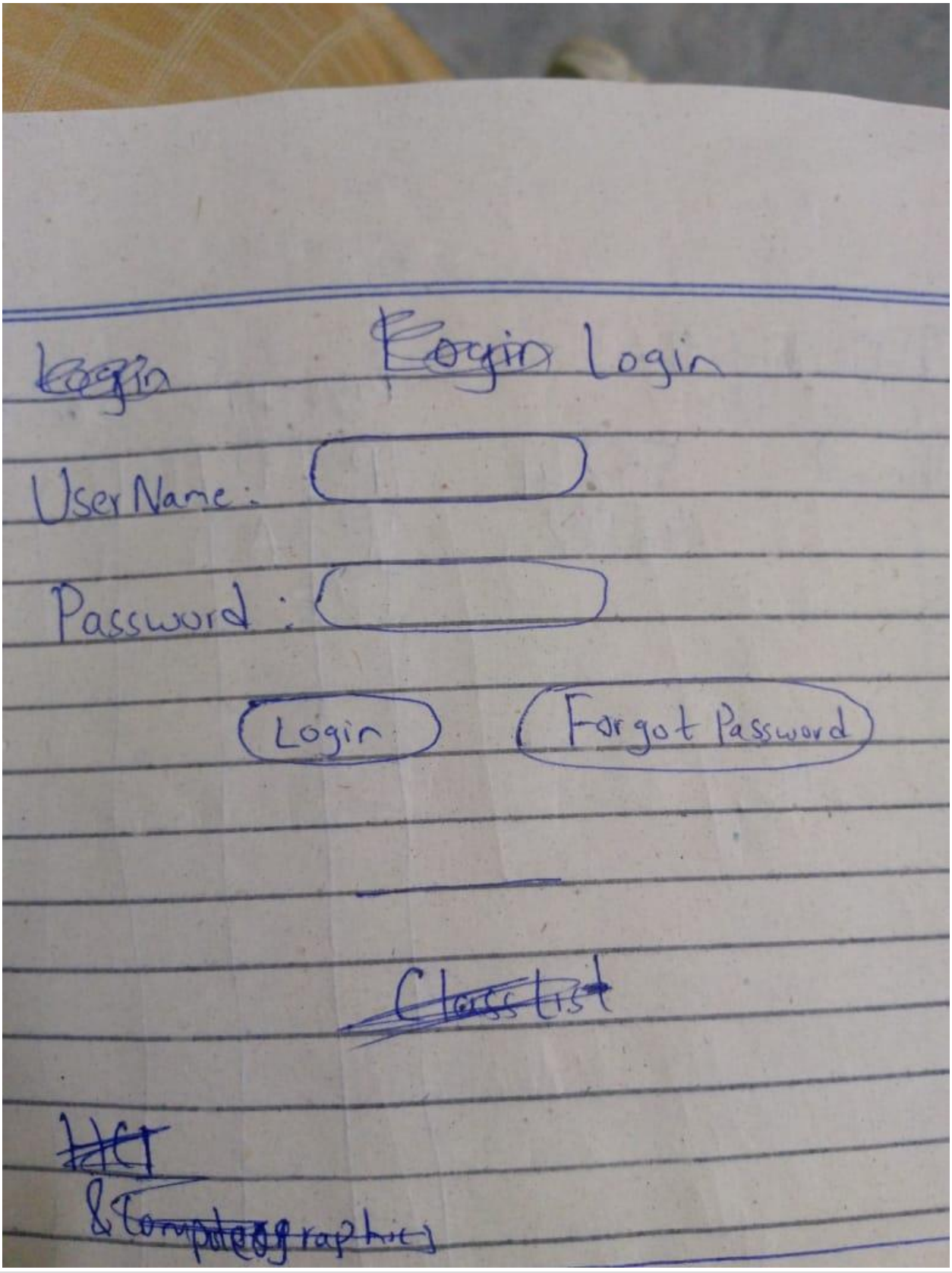
Design Decision 1: Use of Checkboxes for Attendance

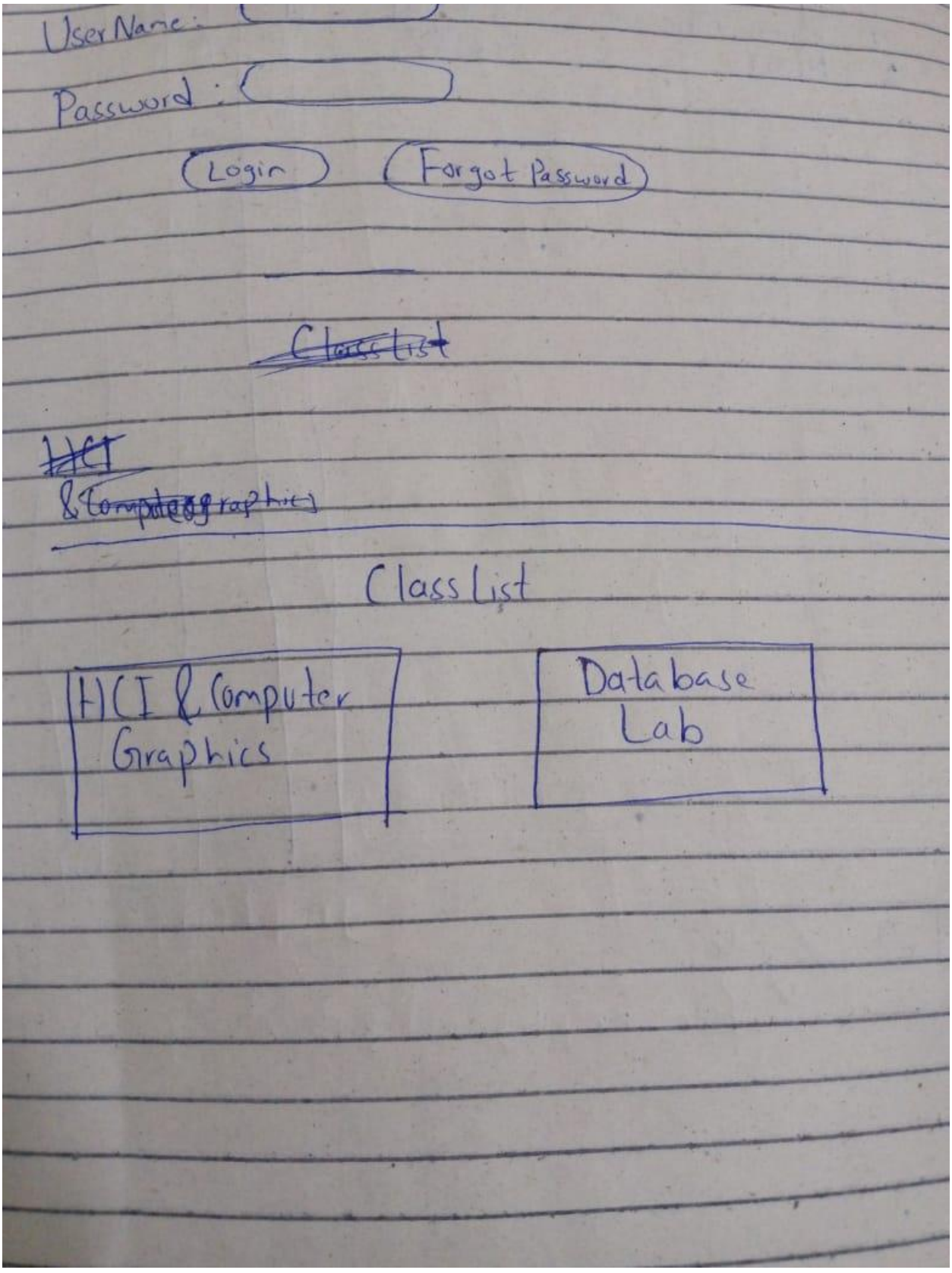
- **Justification:** Checkboxes offer a fast and familiar way to mark multiple students present/absent. They reduce the number of clicks and work well for binary input (present or not).

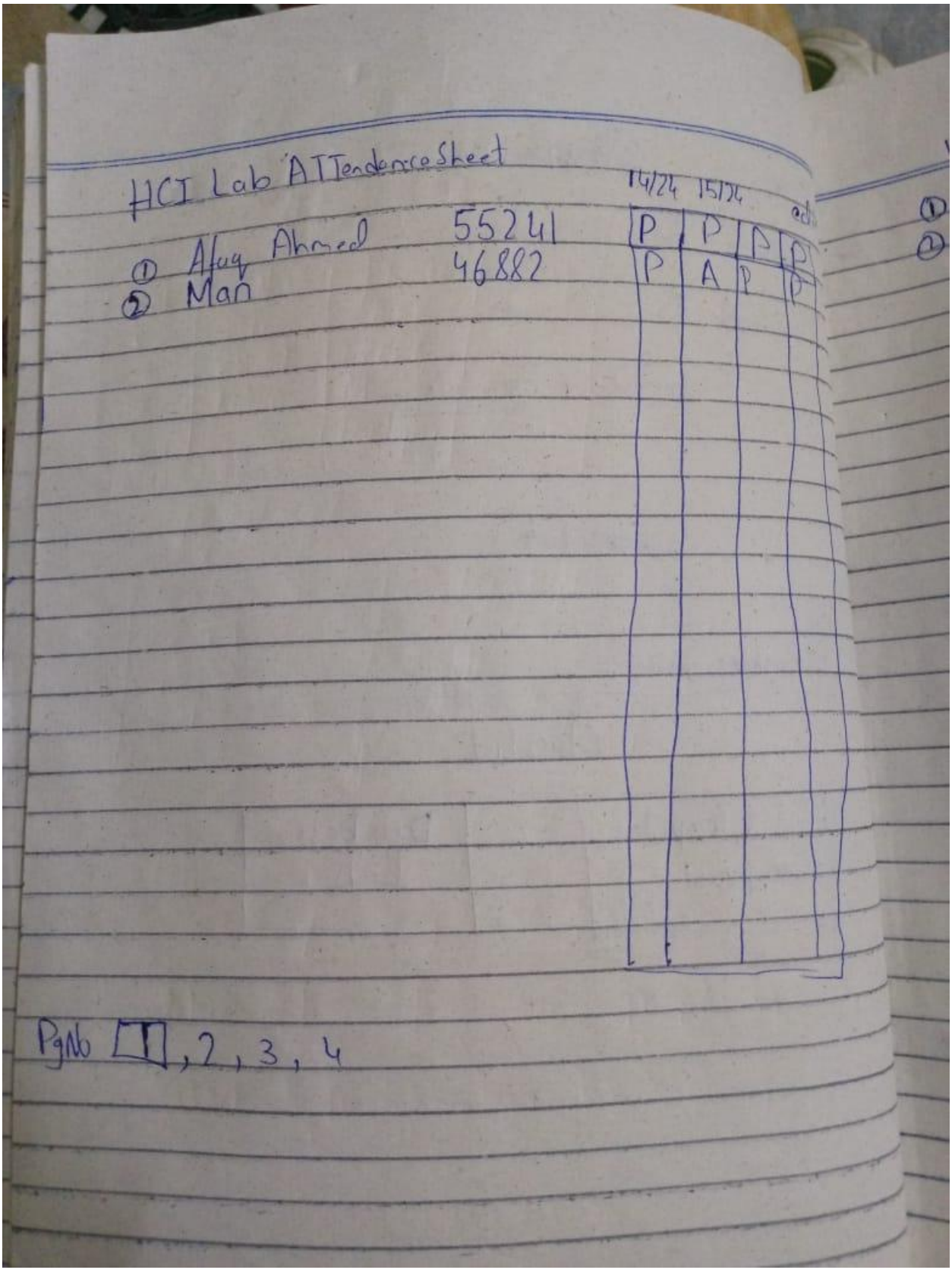
Design Decision 2: Use of a Single Page with Conditional Display

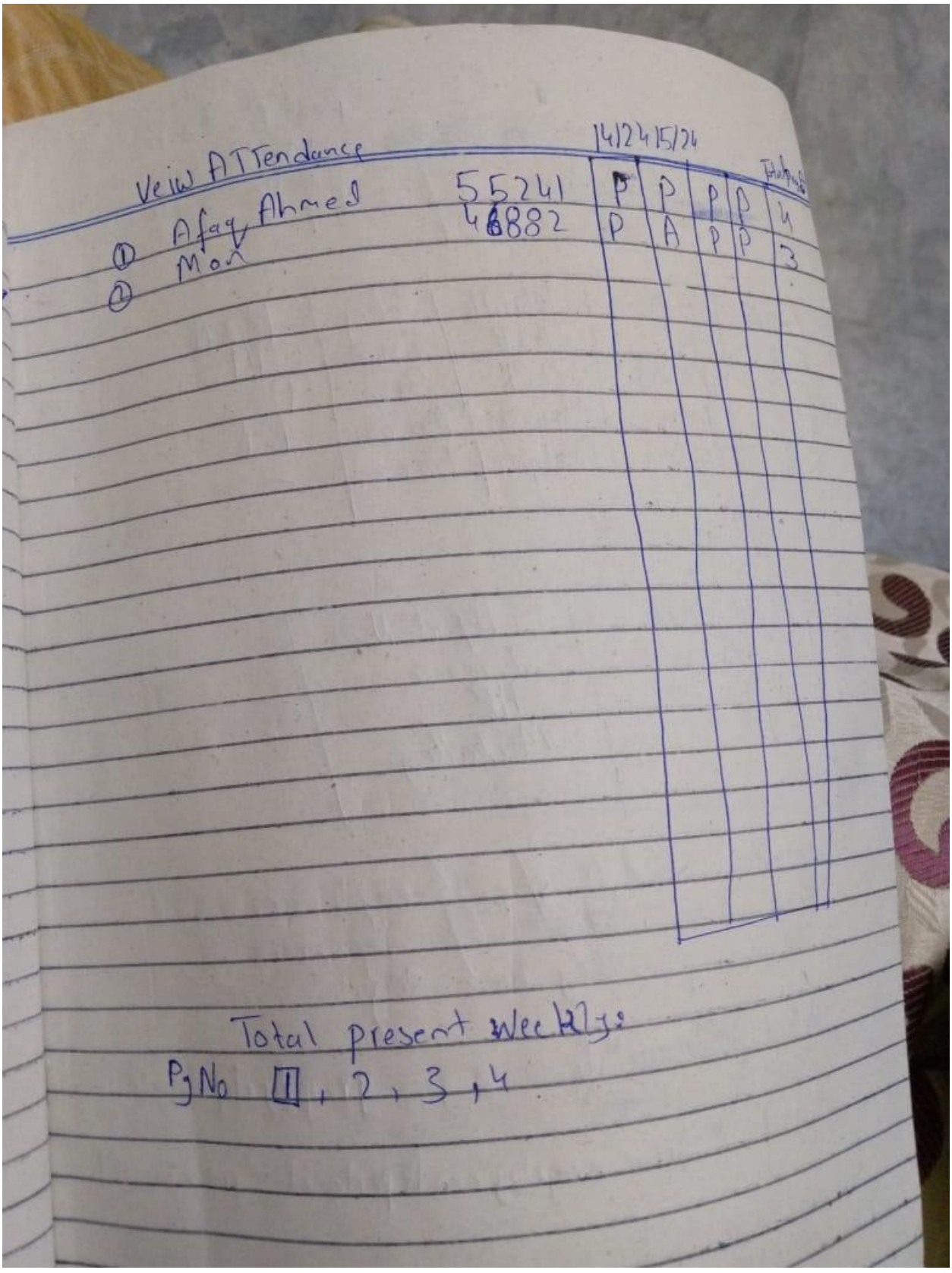
- **Justification:** Instead of navigating across multiple pages (login → attendance), the design uses display: none / classList.remove('hidden'). This improves speed, keeps things simple for users, and avoids unnecessary page loads.

2. Paper Prototype (at least 3 screens)


Login
Login
User Name:
Password :
Login
Forgot Password


User Name :
Password :
Login
Forgot Password
Class list
HCI
& Computer graphics
Class List
HCI & Computer Graphics
Database Lab


HCI Lab ATTendance Sheet
14/24 15/24
① Afaq Ahmed 55241 P P P P
② Man 46882 P A P P
Pg No 1, 2, 3, 4


View ATTendance
14/24/15/24
1. Afaq Ahmed 55241 P P P P 4
2. Mon 48882 P A P P 3
Total present weekly:
Pg No 1, 2, 3, 4

3. Usability Testing (2 Findings)

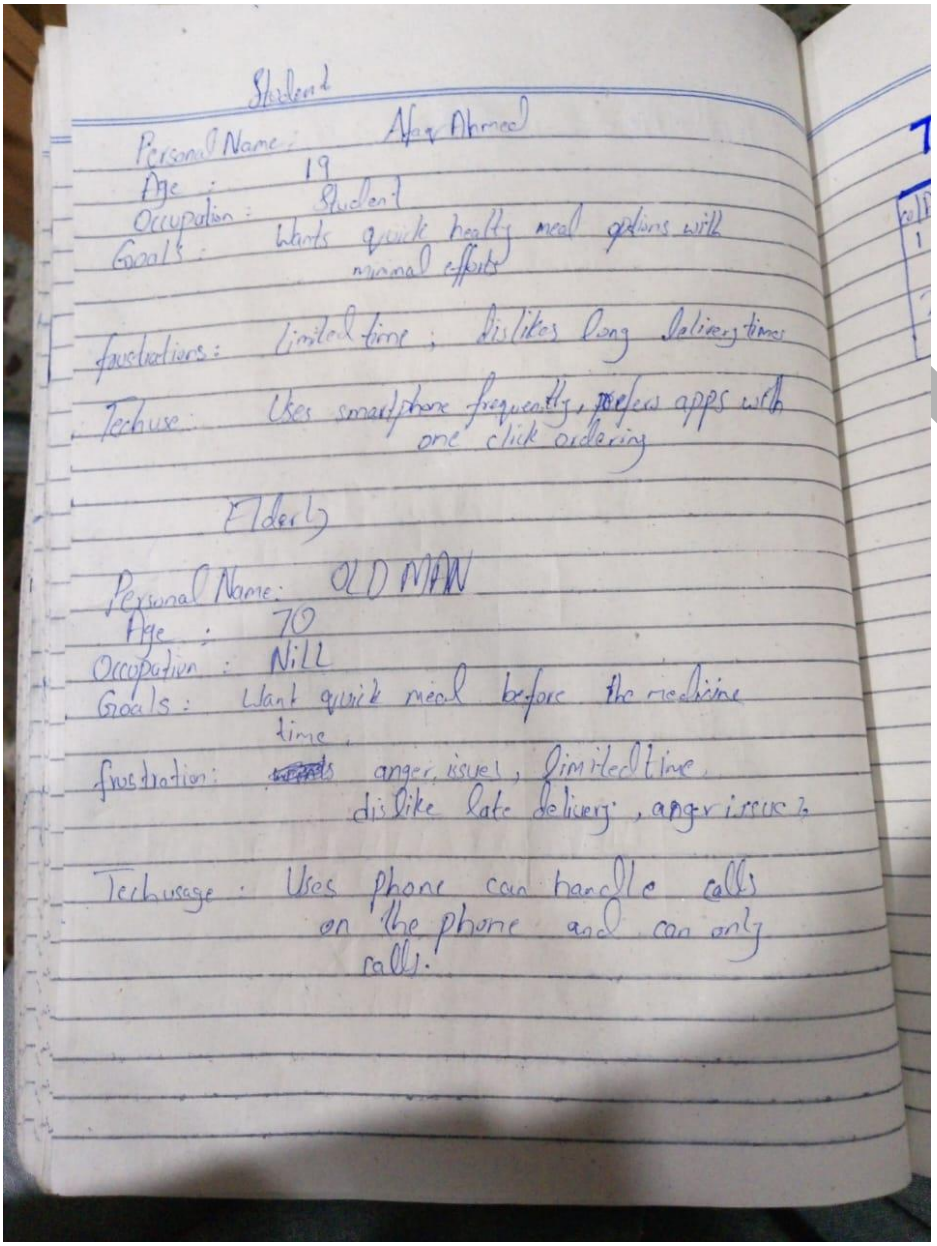
- **Finding 1:** *"It was confusing that the attendance list only appeared after logging in—maybe add a success message or transition animation."*
- **Finding 2:** *"The checkboxes were easy to understand. I liked how fast it was to mark multiple students."*

4. Short Reflection

The usability testing showed me that even small interactions (like showing or hiding sections) need clear visual feedback. One classmate expected a loading animation after clicking 'Login', so I would consider adding a transition. The checkbox system worked smoothly, so I plan to keep it. If I redesign, I might add a back button or clearer navigation.

LAB 11:

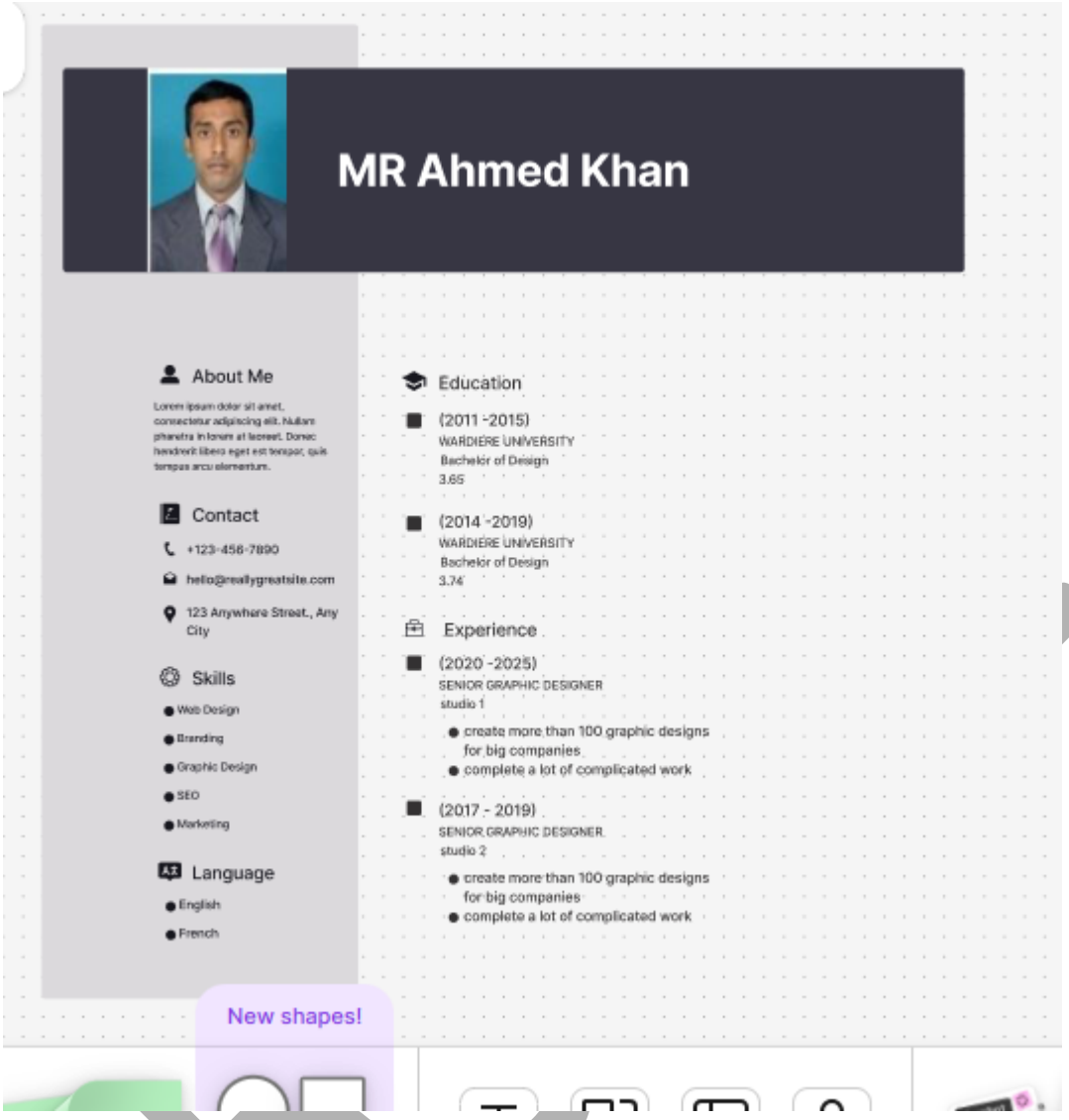
Persona:



Lab Activities

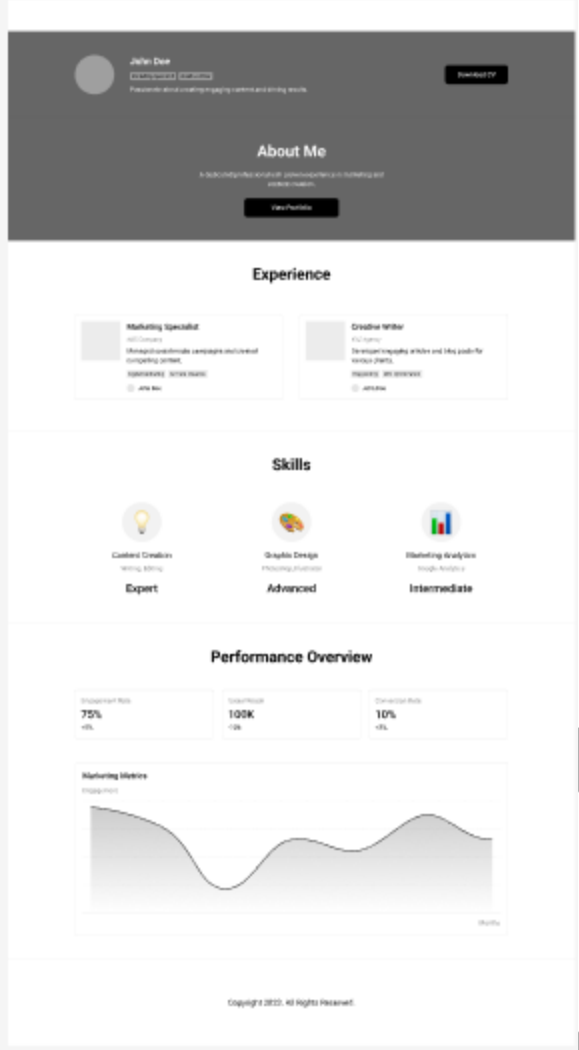
NON Technical user CV:

Link of Figma: https://www.figma.com/board/ficVUgjTYXlbsoxwiYaHzl/Black-and-Grey-Simple-Infographic-Resume?t=ccMUvJ8ykXpNPSLO-1



Technical user CV:

Link of Figma:
https://www.figma.com/design/yINfKxYaRiBrHM7br0QczC/Untitled?t=ccMUvJ8ykXpNPSLO-1


John Doe
About Me
Experience
Skills
Expert
Advanced
Intermediate
Performance Overview
75%
100K
10%