Exp 2 : Drop down list

Ex.no 2: DESIGN A DROP-DOWN LIST OR A MENU IN A GUI KEEPING IN VIEW THE SERIAL POSITION EFFECT

Aim: To design a drop-down list or a menu in a GUI keeping in view the serial position effect.

PROCEDURE: To execute HTML code, you need a web browser.

Follow the steps below to run HTML code on your computer:

1.Open a text editor or an integrated development environment (IDE) such as Notepad, Sublime Text, Visual Studio Code, or any other editor of your choice.

2.Copy the HTML code into the editor and save the file with a .html extension. For example, you can save it as "index.html".

3.Open the saved HTML file with a web browser. You can do this by double-clicking the file, or you can right-click on the file, select "Open with," and choose a web browser from the list.

4.The web browser will render and display the HTML code, executing any scripts or displaying the content as intended.

PROCEDURE- TO DESIGN A DROP-DOWN LIST OR A MENU IN A GUI KEEPING IN VIEW THE SERIAL POSITION EFFECT

1.Define User Goals and Context:

• Understand the specific goals and tasks users aim to accomplish with the drop-down list or menu.

• Identify the context in which the GUI will be used, including the target audience, platform, and any relevant constraints or requirements.

2.Conduct User Research:

• Gather user insights through interviews, surveys, or observations to understand user preferences, needs, and expectations related to drop-down lists or menus.

• Focus on understanding how users perceive and interact with lists, their familiarity with GUIs, and any challenges they face.

3.Identify Design Principles:

• Review existing research and design principles related to drop-down lists, menus, and the serial position effect.

• Determine which design principles are relevant to your specific context and align with the user goals.
• Define the Content:

• Determine the specific options or items that will be included in the drop-down list or menu.

• Consider the information architecture and categorize options if necessary.

4.Design the Visual Representation:

• Decide on the visual representation of the drop-down list or menu. This may include a traditional drop-down, cascading menu, or any other suitable design based on the context and platform.

• Ensure that the visual representation is consistent with the overall GUI design and follows established UX/UI standards.

5.Order and Group Options:

• Apply the serial position effect by ordering the options in a logical manner, such as alphabetical or numerical order.

• If the list is extensive, consider grouping options into relevant categories to aid user comprehension and ease of navigation.

6.Prototype and Test:

• Create a low-fidelity or high-fidelity prototype of the GUI, including the drop-down list or menu.

• Conduct usability testing sessions with representative users, asking them to perform tasks that involve interacting with the drop-down list or menu.

• Observe and collect feedback on the ease of use, efficiency, and user satisfaction.

• Iterate and refine the design based on user feedback and observations.

7.Implement and Evaluate:

• Once the design has been refined and validated through testing, implement the drop-down list or menu in the final GUI.

• Continuously evaluate the performance and user experience of the GUI in real-world usage.

• Collect feedback from users and monitor metrics to identify areas for improvement.

```
<!DOCTYPE html>
<html>
<head>
  <title>Serial Position Effect - Drop-down List</title>
  <style>
    /* Basic styling for the drop-down list */
    select {
      padding: 5px;
      font-size: 16px;
    }
  </style>
</head>
<body>
  <h1>Serial Position Effect - Drop-down List Example</h1>

  <label for="options">Select a Course:</label>
  <select id="options">
    <option value="" selected disabled hidden>Please select</option>
    <option value="option1">CSE</option>
    <option value="option2">EEE</option>
    <option value="option3">IT</option>
    <option value="option4">MECH</option>
```

```
    <option value="option5">CIVIL</option>
  </select>

  <script>
    // Add event listener to capture user selection
    var dropDown = document.getElementById("options");
    dropDown.addEventListener("change", function() {
      var selectedOption = dropDown.value;
      console.log("Selected option: " + selectedOption);
      // You can perform any desired actions based on the selected option here
    });
  </script>
</body>
</html>
```

Exp 3 : mobile key pad

Aim: To design Mobile Keypad focusing on size, layout and devilling

Procedure: some guidelines to focus on when designing a mobile keypad:

1.Size:

- Ensure that the keys are large enough to be easily tapped by users, especially on smaller screens.
- Consider the average size of a user's fingertip and allow for enough spacing between keys to minimize accidental touches.

2.Layout:

- Organize the keys in a logical and intuitive manner, such as following the QWERTY layout commonly used in mobile devices.
- Group related keys together, such as numbers, letters, symbols, and special characters, to improve usability and efficiency.
- Prioritize frequently used keys by placing them in prominent or easily accessible positions.

3.Dividing the Keypad:

- Separate the keypad into functional sections, such as a number pad, letters, and symbols, using clear visual cues or dividers.
- Implement tabs or swipe gestures to switch between different sections of the keypad, if necessary, to reduce clutter and improve usability.

4.Visual Design:

- Use clear and legible fonts for the labels on the keys to enhance readability.
- Ensure sufficient contrast between the key labels and the background color to improve visibility.

• Consider providing visual feedback, such as highlighting the pressed key or displaying characters as they are entered.

5.Predictive Input and Auto-correction:

• Implement predictive input mechanisms, such as auto-suggestions or word completion, to speed up text input and reduce errors.

• Incorporate auto-correction algorithms to automatically correct common typing mistakes, improving the accuracy of text entry.

6.User Testing and Feedback:

• Conduct user testing to evaluate the usability of the keypad design. Gather feedback from users to identify areas of improvement and address any usability issues.

• Iteratively refine the design based on user insights and feedback to ensure an intuitive and userfriendly experience.

```
<!DOCTYPE html>
<html>
<head>
  <title>Mobile Keypad Design</title>
  <style>
    /* Basic styling for the mobile keypad */
    .keypad {
      display: grid;
      grid-template-columns: repeat(3, 1fr);
      gap: 5px;
    }

    .key {
      padding: 15px;
      font-size: 18px;
      text-align: center;
      background-color: #f2f2f2;
      border-radius: 5px;
    }
  </style>
</head>
<body>
```

```
  <h1>Mobile Keypad Design</h1>

  <div class="keypad">
    <div class="key">1</div>
    <div class="key">2</div>
    <div class="key">3</div>
    <div class="key">4</div>
    <div class="key">5</div>
    <div class="key">6</div>
    <div class="key">7</div>
    <div class="key">8</div>
    <div class="key">9</div>
    <div class="key">*</div>
    <div class="key">0</div>
    <div class="key">#</div>
  </div>
</body>
</html>
```

Exp 4: house hold items

```
<!DOCTYPE html>
<html>
<head>
  <title>Household Items Ordering Menu</title>
  <style>
    /* Basic styling for the menu */
    ul {
      list-style-type: none;
    }

    li {
      padding: 10px;
```

```
    }

    .submenu {
      display: none;
    }

    /* Show submenus on hover */
    li:hover .submenu {
      display: block;
    }
  </style>
</head>
<body>
  <h1>Household Items Ordering Menu</h1>

  <ul>
    <li>Electronics
      <ul class="submenu">
        <li>TV</li>
        <li>Refrigerator</li>
        <li>Washing Machine</li>
      </ul>
    </li>
    <li>Furniture
      <ul class="submenu">
        <li>Sofa</li>
        <li>Bed</li>
        <li>Dining Table</li>
      </ul>
    </li>
    <li>Kitchen Appliances
      <ul class="submenu">
        <li>Microwave Oven</li>
```

```
      <li>Coffee Maker</li>
      <li>Toaster</li>
     </ul>
   </li>
   <li>Home Decor
    <ul class="submenu">
      <li>Rugs</li>
      <li>Curtains</li>
      <li>Wall Art</li>
    </ul>
   </li>
  </ul>

</body>
</html>
```

Exp 5: atm interface

AIM: to design a UI ATM interface

procedure

1. Define the document type and HTML root element.
2. Set the page title and include CSS styles.
3. Create a container for centering the content.
4. Display the title "ATM Interface".
5. Create a form with input fields and a submit button.
6. Include an input field and label for the account number.
7. Include an input field and label for the PIN.
8. Add a submit button with the label "Login".
9. Close the form, container, body, and HTML tags.

```
<!DOCTYPE html>
<html>
<head>
  <title>ATM Interface</title>
  <style>
    body {font-family: Arial, sans-serif; text-align: center;}
```

```
   .container {margin: 50px auto; width: 300px;}
   .input-group {margin-bottom: 20px;}
   .input-group label {display: block; text-align: left; margin-bottom: 5px;}
   .input-group input {width: 100%; padding: 5px; font-size: 16px; border-radius: 5px; border: 1px solid #ccc;}
   .button {background-color: #4CAF50; color: white; padding: 10px 20px; font-size: 16px; border: none; border-radius: 5px; cursor: pointer;}
   .button:hover {background-color: #45a049;}
  </style>
</head>
<body>
  <div class="container">
   <h1>ATM Interface</h1>
   <form>
    <div class="input-group">
     <label for="account">Account Number</label>
     <input type="text" id="account" name="account" placeholder="Enter account number">
    </div>
    <div class="input-group">
     <label for="pin">PIN</label>
     <input type="password" id="pin" name="pin" placeholder="Enter PIN">
    </div>
    <div class="input-group">
     <input type="submit" value="Login" class="button">
    </div>
   </form>
  </div>
</body>
</html>
```

Exp 6: tv remote prototype

AIM: to design a protype for a tTV control panel
procedure:
1.The HTML document creates a mobile keypad interface with buttons enclosed in a container.

2.CSS styles are applied to define the layout and appearance of the keypad.

3.JavaScript code tracks the pressed keys using an array.

4.When a button is clicked, the handleKeyPress() function is invoked, which adds the clicked key to the pressedKeys array.

5.The feedback message is updated by joining the elements of the pressedKeys array

```
<!DOCTYPE html>
<html>
<head>
  <title>Mobile Keypad</title>
  <style>
    body {
      display: flex;
      justify-content: center;
      align-items: center;
      height: 100vh;
      margin: 0;
    }
    .keypad {
      display: grid;
      grid-template-columns: repeat(3, 1fr);
      gap: 10px;
      width: 200px
```

```
}
    .key  {
     display: flex;
     justify-content: center;
     align-items: center;
     height: 50px;
     background-color: #ccc;
     border-radius: 5px;
     cursor: pointer;
    }
    #feedback {
     margin-top: 10px;
     font-weight: bold;
     text-align: center;
    }
    button {
     margin-top: 10px;
    }
  </style>
 </head>
 <body>
  <div class="keypad">
   <div class="key" onclick="handleKeyPress('1')">1</div>
   <div class="key" onclick="handleKeyPress('2')">2</div>
   <div class="key" onclick="handleKeyPress('3')">3</div>
   <div class="key" onclick="handleKeyPress('4')">4</div>
   <div class="key" onclick="handleKeyPress('5')">5</div>
   <div class="key" onclick="handleKeyPress('6')">6</div>
   <div class="key" onclick="handleKeyPress('7')">7</div>
   <div class="key" onclick="handleKeyPress('8')">8</div>
   <div class="key" onclick="handleKeyPress('9')">9</div>
```

```
    <div class="key" onclick="handleKeyPress('*')">*</div>
    <div class="key" onclick="handleKeyPress('0')">0</div>
    <div class="key" onclick="handleKeyPress('#')">#</div>
  </div>

  <button onclick="deleteLastKeyPressed()">Delete</button>
  <p id="feedback"></p>

  <script>
    var pressedKeys = [];

    function
      handleKeyPress(key) {
      pressedKeys.push(key);
      updateFeedback();
    }

    function
      deleteLastKeyPressed() {
      pressedKeys.pop();
      updateFeedback();
    }

    function updateFeedback() {
      document.getElementById("feedback").innerText = "Pressed keys: " + pressedKeys.join("");
    }
  </script>
</body>
</html>
```

Exp 7: protype of an automatic vending machine

AIM: to design a protype of an automatic vending machine for drinks

procedure:

1. Define the document type and HTML root element.

2. Set the page title and include CSS styles.

3. Create a container for centering the content.

4. Display the title

5. Create a form with input fields and a submit button.

6. Close the form, container, body, and HTML tags.

```
<!DOCTYPE html>
<html>
<head>
  <title>Automatic Vending Machine</title>
  <style>
    body{font-family:Arial,sans-serif;background-color:#f2f2f2}.container{margin:50px auto;width:400px;text-align:center;padding:20px;background-color:#fff;border-radius:5px;box-shadow:0 0 10px rgba(0,0,0,.1)}.button{display:inline-block;padding:10px 20px;margin:10px;background-color:#4CAF50;color:#fff;text-decoration:none;border-radius:4px;transition:background-color .3s}.button:hover{background-color:#45a049}
  </style>
</head>
<body>
  <div class="container">
    <h1>Automatic Vending Machine</h1>
    <p>Select your drink:</p>

    <button class="button">Coca-Cola</button>
    <button class="button">Pepsi</button>
    <button class="button">Sprite</button>
    <button class="button">Fanta</button>

    <p>Payment Options:</p>
```

```
        <button class="button">Credit Card</button>
        <button class="button">Mobile Payment</button>

        <p>Thank you for using the vending machine!</p>
    </div>
</body>
</html>
```

EX. 8 CREATE AN INTERACTIVE PLAYER PROTOTYPE USING PROTOPIE

Aim: To create an interactive player prototype using protopie.

Procedure:

Step 1: Setting up the Project

1. Download and install ProtoPie (if you haven't already).

2. Launch ProtoPie and create a new project.

3. Set the canvas size to a mobile device resolution (e.g., 375x667 for iPhone 8).

Step 2: Designing the Player UI

1. Create a circle for the play/pause button.

2. Create another circle for the skip button.

3. Design a rectangle for the progress bar.

Step 3: Adding Interactions

1. Select the play/pause button and open the "Interaction" panel.

2. Click on "Add Interaction" and select "On Tap."

3. Choose "Toggle" as the interaction type.

4. Select the play/pause button again, and in the "Layers" panel, create two states: "Play" and "Pause." Design each state accordingly (e.g., play icon in the "Play" state and pause icon in the "Pause" state).

Step 4: Adding Progress Animation 1. Select the progress bar and open the "Interaction" panel. 2. Click on "Add Interaction" and select "On Tap." 3. Choose "Change Property" as the interaction type. 4. Select the progress bar, choose the "Transform" property, and set the "Scale" property to increase the width of the progress bar to simulate progress.

Step 5: Adding Skip Interaction 1. Select the skip button and open the "Interaction" panel. 2. Click on "Add Interaction" and select "On Tap." 3. Choose "Change Screen" as the interaction type. 4. Create a new screen that represents the next song or a skip action.

Step 6: Testing the Prototype 1. Connect a mobile device to the ProtoPie Player app. 2. Click the "Play" button to simulate the play/pause action. 3. Tap the progress bar to simulate song progress. 4. Tap the skip button to navigate to the next song screen.

Step 7: Refining the Prototype 1. Adjust animations, transitions, and timings to make the interactions feel natural. 2. Test the prototype with potential users and gather feedback for improvements.

Output: Result: Thus interactive player prototype using protopie has been created.

EX.9 DESIGN A MOBILE BANKING APPLICATION USING BALSAMIQ DESIGN SOFTWARE

Aim: To design a mobile banking application using balsamiq design software

Procedure:

Step 1: Gather Requirements before designing.

Step 2: Install Balsamiq If you haven't already, download and install Balsamiq on your computer. You can choose between the desktop version or the web-based version, depending on your preference. Step 3: Create a New Project Launch Balsamiq and create a new project for mobile banking app. Choose the appropriate screen size for mobile devices.

Step 4: Start Wireframing, begin creating wireframes for each screen or feature of your mobile banking app. Here's a list of screens you might want to design: 1.Login/Register: Create wireframes for the login and registration screens. Include fields for username/email and password, as well as registration fields if applicable. 2.Account Overview: Design a screen that shows the user's account balances, recent transactions, and other relevant account information. 3.Transactions: Create wireframes for the screen where users can view their transaction history and details. 4.Transfers: Design screens for transferring money between accounts, including selecting the source and destination accounts, entering the amount, and confirming the transfer. 5.Bill Payments: If your app supports bill payments, create wireframes for this feature. Include options to select the biller, enter the amount, and confirm the payment. 6.Settings: Design screens where users can manage their profile, security settings, and notification preferences. 7.Help/Support: Create wireframes for the help and support section, including FAQs, contact information, and chat support if applicable.

Step 5: Use Balsamiq Tools Balsamiq provides various tools to create wireframes quickly. Use basic shapes, text elements, buttons, and other UI components to create your screens. Keep the design simple and focused on functionality.

Step 6: Link Screens (Optional) If you want to create a clickable prototype, you can link the screens together in Balsamiq. This allows you to simulate the user flow by adding clickable areas to buttons and links, enabling you to demonstrate how users would navigate through the app.

Step 7: Review and Refine After creating the initial wireframes, review them to ensure they meet the requirements and provide a seamless user experience. Make any necessary adjustments to the layout, content, or interactions.

EX 10 DESIGN A WEB INTERFACE FOR ONLINE BANKING SYSTEM

Aim: To design a web interface for online banking systems using balsamiq design software
Procedure:

Step 1: Understand the Requirements Before you start designing, make sure you have a clear understanding of the features and functionalities required for the online banking system. This may include user authentication, account overview, fund transfers, transaction history, bill payments, and more.

Step 2: Start a New Project 1. Launch Balsamiq. 2. Create a new project with a suitable canvas size (e.g., desktop or mobile resolution).

Step 3: Design the Main Layout 1. Begin with the main layout, which typically includes a header, sidebar or navigation, and a content area. 2. Use simple shapes (rectangles, text boxes, buttons) to represent each of these elements. 3. Arrange them in a clean and intuitive manner.

Step 4: Design Key Pages Design the following key pages, keeping the overall user flow in mind: 1. Login Page: o Include fields for username and password. o Add a "Login" button. o Consider including a "Forgot Password" link. 2. Account Overview Page: o Display account balances. o Show recent transactions. o Include navigation to other sections. 3. Transfer Funds Page: o Provide fields for selecting source and destination accounts. o Include an amount field. o Add a "Transfer" button. 4. Transaction History Page: o Display a list of past transactions. o Include filters and sorting options. 5. Bill Payments Page: o Allow users to select the biller. o Include fields for entering payment details. o Add a "Pay" button.

Step 5: Use Balsamiq Components Take advantage of Balsamiq's pre-built components to create a more realistic and consistent design. Customize the components to match the branding and overall look of an online banking system.

Step 6: Maintain Clarity and Simplicity 1. Use clear and concise labels. 2. Maintain a clean and consistent visual style. 3. Prioritize user-friendly interactions. 4. Ensure the design is responsive if you're designing for multiple screen sizes.

Step 7: Review and Iterate 1. Review the wireframes/mockups for usability and completeness. 2. Gather feedback from stakeholders or potential users. 3. Iterate on the design based on the feedback received.

EX 11 DESIGN A PROTOTYPE FOR ONLINE MOBILE PAYMENT APPLICATION USING FIGMA

Aim:To design a prototype for online mobile payment application using figma.

Procedure:

Step 1: Understand the Requirements Before you start designing, make sure you have a clear understanding of the features and functionalities required for the mobile payment application. This may include user registration, linking bank accounts or cards, making payments, viewing transaction history, and more.

Step 2: Create a New Figma Project 1. Log in to Figma or create an account if you don't have one. 2. Start a new project and choose the appropriate device size (e.g., iPhone X).

Step 3: Design the Main Screens Design the following key screens that cover the primary user flow of the mobile payment application: 1. Welcome/Onboarding Screen: o Introduce the app and its

benefits. o Include a "Get Started" button. 2. Registration/Login Screen: o Provide fields for user registration (name, email, password). o Include social login options (e.g., Google, Facebook) if applicable. o Allow users to log in with existing credentials. 3. Home/Account Overview Screen: o Display user account balance. o Show quick links to popular actions (e.g., Send Money, Pay Bills). o Provide a transaction summary or recent activity. 4. Send Money Screen: o Allow users to select recipients (contacts or manually enter details). o Provide an amount field. o Include a "Send" button. 5. Transaction History Screen: o List past transactions with details (recipient, amount, date). o Include filters and sorting options.

Step 4: Design Interaction and Flows 1. Use Figma's prototyping tools to create interactions between screens. 2. Set up basic navigation, such as linking buttons to relevant screens. 3. Create transitions (slide, dissolve, etc.) to make the prototype more realistic.

Step 5: Design User Interface Elements 1. Use Figma's design tools to create buttons, input fields, icons, and other UI elements. 2. Maintain a consistent color scheme and typography. 3. Ensure the design is user-friendly and intuitive.

Step 6: Add Interactive Elements 1. Enhance the prototype with interactive elements like buttons, sliders, and toggles. 2. Show how user actions (e.g., tapping "Send") lead to changes on the screen.

Step 7: Review and Iterate 1. Review the prototype for usability and completeness. 2. Share the prototype with stakeholders or potential users to gather feedback. 3. Iterate on the design based on the feedback received.

EX 12 INTERACTIVE DESIGN ANALYSIS BASED ON PRINCIPLES OF UNIVERSAL DESIGN

Aim: To perform interactive design analysis based on principles of universal design.

Analyzing Google Play's design based on the principles of Universal Design, we can see how well it aligns with the goal of creating an accessible and inclusive user experience for a diverse range of users.

1. Equitable Use: o Positive: Google Play aims to provide the same means of accessing and discovering apps, games, movies, and books for all users, regardless of their abilities. o Consideration: The platform offers a wide variety of content, catering to different preferences and interests, which promotes equity in use.

2. Flexibility in Use: o Positive: Google Play offers multiple ways for users to find content. It provides search functionality, recommendations, and categorized sections, allowing users to choose how they explore the platform. o Consideration: It could further enhance flexibility by allowing users to customize the app's interface based on their needs or preferences.

3. Simple and Intuitive Use: o Positive: Google Play tends to follow Material Design principles, which generally prioritize simplicity and intuitive interactions. o Consideration: The platform's search and navigation are relatively straightforward, but there may be areas for improvement in making complex actions, such as account management, even more intuitive.

4. Perceptible Information: o Positive: Google Play typically provides clear visual and textual information about apps, games, movies, and books. This is essential for users with different sensory abilities. o Consideration: It's essential to ensure that the information is presented in a way that is accessible to users with visual impairments or those who rely on screen readers.

5. Tolerance for Error: o Positive: Google Play generally allows users to preview app details and read reviews before downloading, which helps reduce the risk of errors or unwanted installations. o Consideration: Enhancements can be made to error prevention, such as providing clearer cues for actions and reducing the likelihood of accidental purchases.

6. Low Physical Effort: o Positive: Google Play's design is mainly touch-based, which is user-friendly and accommodates users with varying physical abilities. o Consideration: The platform can continually optimize its touch interactions, taking into account users with motor impairments who may need more precise touch targets.

7. Size and Space for Approach and Use: o Positive: The overall layout of Google Play provides ample space for users to interact with content. o Consideration: The app could further ensure that buttons and interactive elements have sufficient spacing to accommodate users with limited dexterity