SRM Institute of Science and Technology
Delhi – Meerut Road, Sikri Kalan, Ghaziabad, Uttar Pradesh – 201204
Department of Computer Applications
Circular – 2023-24
BCA DS 5th Sem

No-Code Applications (UDS23G03J)

Lab Manual

Lab 1: Tour around the different No-Code Tool landscape

Title: Tour of No-Code Tools

Aim: To familiarize students with a variety of no-code platforms and understand their capabilities.

Procedure:

1. Explore the following no-code platforms: Webflow, Bubble, Glide, Thunkable, Google Teachable Machine, Lobe.ai, LandBot, AirTable, and Shopify.
2. For each platform, identify its primary purpose, target users, and key features.
3. Document your findings in a table or report, comparing the strengths and weaknesses of each tool.

Source Code: N/A (Exploratory Lab - No direct code involved)

Input: N/A

Expected Output: A comparative report or table summarizing the features and use cases of the explored no-code tools.

Lab 2: Building Workflow Automation using Low-Code

Title: Workflow Automation with Low-Code

Aim: To design and implement an automated workflow using a low-code platform.

Procedure:

1. Select a low-code platform (e.g., Zapier, IFTTT, or a similar tool).
2. Identify a real-world workflow that can be automated (e.g., sending email notifications for new form submissions, saving social media posts to a spreadsheet).
3. Design the workflow in the low-code platform, specifying the triggers, actions, and any necessary conditions.
4. Test the workflow thoroughly to ensure it functions correctly.

Source Code: (Provide the configuration or export of the workflow from the low-code platform, if available)

Input: (Describe the data or event that triggers the workflow. E.g., "A new submission to the contact form on website X.")

Expected Output: A functional automated workflow that performs the specified task, along with documentation of the workflow design and testing results.

Lab 3: Create a web scraping tool using No-Code

Title: No-Code Web Scraping Tool

Aim: To extract data from a website using a no-code web scraping tool.

Procedure:

1. Choose a no-code web scraping tool (e.g., Octoparse, ParseHub).
2. Identify a target website and the specific data to be extracted.
3. Configure the web scraping tool to navigate the website and extract the desired data (e.g., product names, prices, reviews).
4. Run the web scraping tool and verify that the data is extracted correctly.
5. Export the extracted data in a suitable format (e.g., CSV, JSON).

Source Code: (Provide the configuration or "recipe" of the web scraping tool)

Input: (Specify the URL of the target website)

Expected Output: A file (CSV, JSON, etc.) containing the extracted data from the website.

Lab 4: Working with the Designer interface of WebFlow

Title: WebFlow Designer Interface

Aim: To become proficient in using the WebFlow designer interface for website design.

Procedure:

1. Open the WebFlow designer and familiarize yourself with the different panels and tools (e.g., Navigator, Styles, Settings).
2. Create a new WebFlow project.
3. Experiment with adding and styling elements (e.g., divs, headings, paragraphs, images).
4. Learn how to use WebFlow's layout tools (e.g., Flexbox, Grid) to create responsive designs.

Source Code: (The Webflow project file or export, if available)

Input: N/A (Design exercise)

Expected Output: A basic website layout created in WebFlow, demonstrating understanding of the designer interface.

Lab 5: Create Responsive WebPage using WebFlow

Title: Responsive Webpage with WebFlow

Aim: To design and build a responsive webpage using WebFlow.

Procedure:

1. Use WebFlow to create a webpage with several sections (e.g., header, navigation, content area, footer).
2. Apply styles and layout to the webpage.
3. Utilize WebFlow's breakpoint feature to ensure the webpage adapts correctly to different screen sizes (desktop, tablet, mobile).
4. Test the responsiveness of the webpage using WebFlow's preview mode and by resizing the browser window.

Source Code: (The Webflow project file or export)

Input: N/A (Design exercise)

Expected Output: A fully responsive webpage that displays correctly on various devices.

Lab 6: Using Bubble build features like sign up forms, expense trackers, inboxes, shopping carts

Title: Building Features with Bubble

Aim: To implement common web application features using Bubble.

Procedure:

1. Create a new Bubble application.
2. For each of the following features, design and implement the necessary UI and logic:

   Sign-up form: Create input fields for user information and implement user authentication.

   Expense tracker: Design a data structure to store expenses and create UI for adding, viewing, and summarizing expenses.

   Inbox: Implement a system for sending and receiving messages, including UI for displaying message lists and individual messages.

   Shopping cart: Create a data structure for products and cart items, and implement UI for adding products to the cart, viewing the cart, and checking out.

Source Code: (The Bubble application file or export)

Input: (For each feature, describe the user input or actions required to test it. E.g., "For the sign-up form, enter a valid email and password.")

Expected Output: A Bubble application with functional sign-up forms, expense tracker, inbox, and shopping cart features.

Lab 7: Build a Mindfulness app using Glide

Title: Mindfulness App with Glide

Aim: To develop a mobile application focused on mindfulness using Glide.

Procedure:

1. Create a new Glide app.
2. Design the app's data structure (e.g., for guided meditations, inspirational quotes, progress tracking). This might involve using Google Sheets as the data source.
3. Create the app's user interface, including screens for accessing different mindfulness exercises, viewing quotes, and tracking progress.
4. Implement any necessary logic or functionality, such as displaying content dynamically based on user selections.

Source Code: (The Glide app configuration or export)

Input: (Describe the data used in the app, e.g., "Mindfulness exercises and quotes stored in a Google Sheet.")

Expected Output: A functional mobile app that provides users with mindfulness resources and tools.

Lab 8: Build a Task Tracker App Using Glide

Title: Task Tracker App with Glide

Aim: To build a mobile application for managing and tracking tasks using Glide.

Procedure:

1. Create a new Glide app.
2. Define the data structure for tasks (e.g., task name, description, due date, status). This will likely involve using a Google Sheet.
3. Design the app's UI, including screens for viewing task lists, adding new tasks, editing existing tasks, and marking tasks as complete.
4. Implement features such as sorting, filtering, and searching for tasks.

Source Code: (The Glide app configuration)

Input: (Describe the task data used for testing, e.g., "A sample list of tasks with varying due dates and statuses in a Google Sheet.")

Expected Output: A mobile app that allows users to effectively manage and track their tasks.

Lab 9: Build an app using Thunkable to sell products

Title: E-commerce App with Thunkable

Aim: To develop a mobile app for selling products using Thunkable.

Procedure: 1. Create a new project in Thunkable. 2. Design the app's user interface, including screens for product listings, product details, and a shopping cart. 3. Implement the app's functionality, such as browsing products, adding products to the cart, and processing orders. You might need to integrate with a backend service for product data and order management. 4. Test the app on a mobile device or emulator.

Source Code: (The Thunkable project file)

Input: (Describe the product data used in the app, e.g., "A list of products with names, descriptions, prices, and images.")

Expected Output: A mobile app that allows users to browse, select, and purchase products.

Lab 10: Detect and Classify Face Masks using Google Teachable Machine

Title: Face Mask Detection with Google Teachable Machine

Aim: To train a machine learning model to detect and classify face masks using Google Teachable Machine.

Procedure:

1. Collect a dataset of images containing faces with and without masks.
2. Open Google Teachable Machine and create a new image project.
3. Upload the collected images to Teachable Machine, labeling them appropriately (e.g., "Mask" and "No Mask").
4. Train the machine learning model.
5. Test the model using the built-in preview and export the model.
6. (Optional) Integrate the exported model into a simple application (e.g., a web app) to perform real-time face mask detection.

Source Code: (The exported Teachable Machine model)

Input: (Describe the image data used for training and testing.)

Expected Output: A trained machine learning model that can accurately detect and classify face masks.

Lab 11: Build a Image Classification Model Using Lobe.ai

Title: Image Classification with Lobe.ai

Aim: To train an image classification model using Lobe.ai.

Procedure:

1. Download and install Lobe.ai.
2. Gather a dataset of images for the classification task (e.g., classifying different types of objects).
3. Create a new project in Lobe.ai and import the image dataset.
4. Label the images within Lobe.ai.
5. Train the image classification model.
6. Test the model within Lobe.ai and export it for use in other applications.

Source Code: (The exported Lobe.ai model)

Input: (Describe the image data used for training and testing the model.)

Expected Output: A trained image classification model that can accurately classify images into the defined categories.

Lab 12: Build a Conversational Chatbot using LandBot

Title: Conversational Chatbot with LandBot

Aim: To design and build a conversational chatbot using LandBot.

Procedure:

1. Create a LandBot account and start a new project.
2. Design the chatbot's conversation flow using LandBot's visual editor. Include different types of interactions (e.g., text responses, buttons, questions).
3. Configure the chatbot's responses and logic.
4. Test the chatbot using LandBot's preview feature.
5. Embed the chatbot on a website or integrate it with a messaging platform.

Source Code: (The LandBot configuration or export)

Input: (Describe the sample user interactions used to test the chatbot.)

Expected Output: A functional chatbot that can engage in conversations with users and provide relevant information or assistance.

Lab 13: Create a workflow in AirTable

Title: Workflow in AirTable

Aim: To design and implement an automated workflow using AirTable.

Procedure:

1. Create a new base in AirTable.
2. Design the tables and fields needed for the workflow (e.g., for managing projects, tracking tasks, or managing customer information).
3. Use AirTable's automation features to create a workflow that automates a specific process (e.g., sending email notifications when a task is completed, updating records based on certain conditions).
4. Test the workflow to ensure it functions correctly.

Source Code: (The AirTable base configuration or export, if available)

Input: (Describe the data or events that trigger the workflow in AirTable.)

Expected Output: A functional workflow in AirTable that automates the specified process.

Lab 14: Build Online Store using Shopify

Title: Online Store with Shopify

Aim: To set up and configure an online store using Shopify.

Procedure:

1. Create a Shopify account.
2. Choose a theme for the online store.
3. Add products to the store, including descriptions, images, and pricing.
4. Configure payment gateways and shipping options.
5. Customize the store's design and layout.
6. Test the store's functionality, including browsing products, adding items to the cart, and completing a checkout.

Source Code: N/A (Shopify is a hosted platform, but you can provide details of theme customizations or any custom code snippets used)

Input: (Describe the product data used in the store.)

Expected Output: A functional online store on the Shopify platform.

Lab 15: Develop a website using a No-Code Stack of your choice

Title: Website Development with a No-Code Stack

Aim: To design and build a complete website using a combination of no-code tools.

Procedure:

1. Choose a no-code stack of tools (e.g., Webflow for design, AirTable for content management, Zapier for automation).
2. Plan the website's structure, content, and functionality.
3. Use the selected no-code tools to build the website, integrating them as needed.
4. Test the website thoroughly to ensure it is functional, responsive, and user-friendly.
5. Deploy the website.

Source Code: (Provide details of the tools used and any relevant configurations or exports. For example, the Webflow project file, AirTable base structure, and Zapier workflow.)

Input: (Describe the content and data used on the website.)

Expected Output: A complete and functional website built using a no-code stack.