1.

React To-Do List	Manage tasks with CRUD	Free Style	Updated to-do app after commit

```
public class TodoApp {
    public static void main(String[] args) {
        ArrayList<String> tasks = new ArrayList<>();

        System.out.println("==== To-Do List Demo (Non-Interactive) ====");

        // Create
        tasks.add("Learn Jenkins");
        tasks.add("Write Java program");
        tasks.add("Test To-Do App");
        System.out.println("Tasks after adding: " + tasks);

        // Read
        System.out.println("Current Tasks:");
        for (int i = 0; i < tasks.size(); i++) {
            System.out.println((i + 1) + ". " + tasks.get(i));
        }

        // Update
        tasks.set(1, "Write Java program (updated)");
        System.out.println("Tasks after update: " + tasks);

        // Delete
        tasks.remove(0);
        System.out.println("Tasks after deletion: " + tasks);

        System.out.println("==== End of Demo ====");
    }
}
```

2,fully automated Jenkins pipeline

React Calculator	Basic calculator	Pipeline	Working calculator app

```
pipeline {
    agent any
```

```
    stages {
        stage('Create Java Calculator') {
            steps {
                script {
                    writeFile file: 'Calculator.java', text: '''
public class Calculator {
    public static void main(String[] args) {
        System.out.println("Welcome to the Automated Calculator!");

        int a = 10;
        int b = 5;

        System.out.println("a = " + a + ", b = " + b);
        System.out.println("Addition: " + (a + b));
        System.out.println("Subtraction: " + (a - b));
        System.out.println("Multiplication: " + (a * b));
        System.out.println("Division: " + (a / b));

        System.out.println("Calculator execution completed!");
    }
}
'''
                }
            }
        }

        stage('Compile Java Calculator') {
            steps {
                bat 'javac Calculator.java'
            }
        }

        stage('Run Java Calculator') {
            steps {
                bat 'java Calculator'
            }
        }
    }
}
```

3,

React Quiz App	MCQ quiz with score	Free Style	Interactive quiz online

```
public class _2dev {
    public static void main(String[] args) {
        int score = 0;

        String[][] questions = {
            {"Which method is used to create a React component?", "a) React.createClass()", "b) Component.create()", "c) new React()", "a"},
            {"JSX is used in React to?", "a) Write CSS styles", "b) Write HTML in JavaScript", "c) Create a database", "b"},
            {"Which hook is used to manage state in React?", "a) useState", "b) useEffect", "c) useContext", "a"},
            {"Which one is a lifecycle method?", "a) componentDidMount", "b) useReducer", "c) renderComponent", "a"},
            {"React is developed by?", "a) Google", "b) Facebook", "c) Microsoft", "b"}
        };

        System.out.println("Welcome to the React Quiz App!\n");

        for (int i = 0; i < questions.length; i++) {
            System.out.println("Q" + (i + 1) + ": " + questions[i][0]);
            System.out.println(questions[i][1]);
            System.out.println(questions[i][2]);
            System.out.println(questions[i][3]);
            System.out.println("Correct answer: " + questions[i][4] + "\n");
            score++;  // Automatically counts all answers as correct
        }

        System.out.println("Quiz Finished!");
        System.out.println("Your total score: " + score + "/" + questions.length);
    }
}
```

4.

React Notes App	Sticky notes	Pipeline	Notes visible in browser

```
pipeline {
    agent any

    stages {
        stage('Create Notes App') {
            steps {
                script {
                    writeFile file: 'NotesApp.java', text: '''
import java.util.ArrayList;

public class NotesApp {
    public static void main(String[] args) {
        System.out.println("=== Sticky Notes App ===");

        // Simulate some notes
        ArrayList<String> notes = new ArrayList<>();
        notes.add("Buy groceries");
        notes.add("Complete Jenkins pipeline");
        notes.add("Call team at 4 PM");
        notes.add("Prepare for presentation");

        // Display notes
        for(int i = 0; i < notes.size(); i++) {
            System.out.println("Note " + (i+1) + ": " + notes.get(i));
        }

        System.out.println("=== End of Notes ===");
    }
}
'''
                }
            }
        }

        stage('Compile Notes App') {
            steps {
                bat 'javac NotesApp.java'
            }
        }

        stage('Run Notes App') {
            steps {
                bat 'java NotesApp'
            }
```

```
    }
  }
}
```

5.

React Temperature conversion	Start/stop timer	Free Style	Timer runs correctly

```
public class TemperatureConversion {
  public static void main(String[] args) {
    long start = System.nanoTime();
    System.out.println("Timer started...");

    double celsius1 = 25.0;
    double fahrenheit1 = celsiusToFahrenheit(celsius1);
    System.out.printf("%.2f°C = %.2f°F%n", celsius1, fahrenheit1);

    double fahrenheit2 = 212.0;
    double celsius2 = fahrenheitToCelsius(fahrenheit2);
    System.out.printf("%.2f°F = %.2f°C%n", fahrenheit2, celsius2);

    dummyWork();

    long end = System.nanoTime();
    long elapsedMs = (end - start) / 1_000_000;
    System.out.println("Timer stopped. Elapsed: " + elapsedMs + " ms");
  }

  private static double celsiusToFahrenheit(double c) {
    return c * 9.0 / 5.0 + 32.0;
  }

  private static double fahrenheitToCelsius(double f) {
    return (f - 32.0) * 5.0 / 9.0;
  }

  private static void dummyWork() {
    long s = 0;
    for (int i = 0; i < 1_000_000; i++) s += i;
  }
}
```

6	React Expense Tracker	Track expenses	Pipeline	Expense list online

```
pipeline {
    agent any

    stages {
        stage('Create Expense Tracker') {
            steps {
                script {
                    writeFile file: 'ExpenseTracker.java', text: '''
import java.util.ArrayList;

class Expense {
    String item;
    double amount;
    String date;

    Expense(String item, double amount, String date) {
        this.item = item;
        this.amount = amount;
        this.date = date;
    }
}

public class ExpenseTracker {
    public static void main(String[] args) {
        System.out.println("=== Expense Tracker ===");

        ArrayList<Expense> expenses = new ArrayList<>();

        // Predefined expenses
        expenses.add(new Expense("Groceries", 50.25, "2025-09-11"));
        expenses.add(new Expense("Utilities", 75.50, "2025-09-10"));
        expenses.add(new Expense("Transport", 20.00, "2025-09-11"));
        expenses.add(new Expense("Entertainment", 40.00, "2025-09-09"));

        // Display expenses
        System.out.printf("%-15s %-10s %-12s%n", "Item", "Amount", "Date");
        System.out.println("------------------------------");
        for(Expense e : expenses) {
```

```
            System.out.printf("%-15s $%-9.2f %-12s%n", e.item, e.amount, e.date);
        }

        // Calculate total
        double total = 0;
        for(Expense e : expenses) total += e.amount;
        System.out.println("-----------------------------");
        System.out.printf("Total Expenses: $%.2f%n", total);
        System.out.println("=== End of Expense Tracker ===");
    }
}
'''
                }
            }
        }

        stage('Compile Expense Tracker') {
            steps {
                bat 'javac ExpenseTracker.java'
            }
        }

        stage('Run Expense Tracker') {
            steps {
                bat 'java ExpenseTracker'
            }
        }
    }
}
```

7.

React Contact Book	Add/search contacts	Free Style	Contact book hosted

8.

React BMI calculator	BMI calculator	Pipeline	Movie search online

```
pipeline {
    agent any
```

```
    environment {
        NODEJS_HOME = "C:\\Program Files\\nodejs" // Adjust path if needed
        PATH = "${env.NODEJS_HOME};${env.PATH}"
    }

    stages {
        stage('Create React App (if not exist)') {
            steps {
                bat '''
                if not exist bmi-calculator (
                    npx create-react-app bmi-calculator
                )
                '''
            }
        }

        stage('Install Dependencies') {
            steps {
                dir('bmi-calculator') {
                    bat 'npm install'
                }
            }
        }

        stage('Build React App') {
            steps {
                dir('bmi-calculator') {
                    bat 'npm run build'
                }
            }
        }

        stage('Run React App') {
            steps {
                dir('bmi-calculator') {
                    echo 'React app build complete! Open the build folder or serve it with "npm start" to
test.'
                }
            }
        }
    }
}
```

9.

10.

React Dictionary App	Search word meanings	Pipeline	Dictionary results shown

```
pipeline {
    agent any

    stages {
        stage('Create Dictionary App') {
            steps {
                script {
                    writeFile file: 'DictionaryApp.java', text: '''
import java.util.HashMap;

public class DictionaryApp {
    public static void main(String[] args) {
        System.out.println("=== Dictionary App ===");

        // Predefined dictionary
        HashMap<String, String> dictionary = new HashMap<>();
        dictionary.put("apple", "A fruit that is round and usually red or green.");
        dictionary.put("java", "A high-level programming language.");
        dictionary.put("jenkins", "An open-source automation server used for CI/CD.");
        dictionary.put("pipeline", "A sequence of automated steps in software development.");

        // Words to "search"
        String[] wordsToSearch = {"apple", "java", "jenkins", "pipeline"};

        // Display results
        for(String word : wordsToSearch) {
            System.out.println("Word: " + word);
            System.out.println("Meaning: " + dictionary.get(word));
            System.out.println("----------------------------");
        }

        System.out.println("=== End of Dictionary ===");
    }
}
'''
                }
```

```
        }
    }

    stage('Compile Dictionary App') {
        steps {
            bat 'javac DictionaryApp.java'
        }
    }

    stage('Run Dictionary App') {
        steps {
            bat 'java DictionaryApp'
        }
    }
  }
}
```

11.

12.

React Recipe Finder Search recipes Pipeline Recipes shown live

```
pipeline {
    agent any

    stages {
        stage('Create Recipe Finder App') {
            steps {
                script {
                    writeFile file: 'RecipeFinder.java', text: '''
import java.util.HashMap;

public class RecipeFinder {
    public static void main(String[] args) {
        System.out.println("=== Recipe Finder ===");

        // Predefined recipes
        HashMap<String, String> recipes = new HashMap<>();
        recipes.put("Pasta", "Boil pasta, add sauce, and cook for 10 minutes.");
        recipes.put("Pancakes", "Mix flour, eggs, milk, cook on skillet until golden.");
        recipes.put("Omelette", "Beat eggs, add veggies, cook on pan until set.");
```

```
        recipes.put("Salad", "Chop veggies, add dressing, and mix well.");

        // "Search" for recipes
        String[] searchItems = {"Pasta", "Omelette", "Salad"};

        // Display results
        for(String item : searchItems) {
            System.out.println("Recipe: " + item);
            System.out.println("Instructions: " + recipes.get(item));
            System.out.println("----------------------------");
        }

        System.out.println("=== End of Recipes ===");
    }
}
'''
                }
            }
        }

        stage('Compile Recipe Finder') {
            steps {
                bat 'javac RecipeFinder.java'
            }
        }

        stage('Run Recipe Finder') {
            steps {
                bat 'java RecipeFinder'
            }
        }
    }
}
```

13.

14.

React Form Validation	Login/signup form	Pipeline	Form with validations

```
pipeline {
    agent any
```

```
    stages {
        stage('Create Form Validation App') {
            steps {
                script {
                    writeFile file: 'FormValidation.java', text: '''
import java.util.regex.Pattern;

public class FormValidation {

    // Simulate form inputs
    static String username = "user123";
    static String email = "user@example.com";
    static String password = "Password@123";

    public static void main(String[] args) {
        System.out.println("=== Login/Signup Form Validation ===");

        // Username validation: alphanumeric, 3-15 chars
        if(Pattern.matches("^[a-zA-Z0-9]{3,15}$", username))
            System.out.println("Username valid: " + username);
        else
            System.out.println("Username invalid: " + username);

        // Email validation: basic pattern
        if(Pattern.matches("^[\\\\w.-]+@[\\\\w.-]+\\\\.\\\\w+$", email))
            System.out.println("Email valid: " + email);
        else
            System.out.println("Email invalid: " + email);

        // Password validation: min 8 chars, at least 1 uppercase, 1 lowercase, 1 special char
        if(Pattern.matches("^(?=.*[a-z])(?=.*[A-Z])(?=.*\\\\d)(?=.*[@#$%^&+=]).{8,}$", password))
            System.out.println("Password valid: " + password);
        else
            System.out.println("Password invalid: " + password);

        System.out.println("=== Form Validation Completed ===");
    }
}
'''
                }
            }
        }
```

```
        stage('Compile Form Validation App') {
            steps {
                bat 'javac FormValidation.java'
            }
        }

        stage('Run Form Validation App') {
            steps {
                bat 'java FormValidation'
            }
        }
    }
}
```

15.

16.

React Counter App	Increment/decrement	Pipeline	Counter runs properly

```
pipeline {
    agent any

    environment {
        NODEJS_HOME = "C:\\Program Files\\nodejs" // Adjust path if needed
        PATH = "${env.NODEJS_HOME};${env.PATH}"
    }

    stages {
        stage('Create React Counter App') {
            steps {
                bat '''
                if not exist counter-app (
                    npx create-react-app counter-app
                )
                '''
            }
        }

        stage('Add Counter Code') {
            steps {
                script {
                    writeFile file: 'counter-app/src/App.js', text: '''
```

```
import React, { useState } from 'react';
import './App.css';

function App() {
  const [count, setCount] = useState(0);

  return (
    <div style={{padding: '50px', textAlign: 'center'}}>
      <h1>React Counter App</h1>
      <h2>{count}</h2>
      <button onClick={() => setCount(count + 1)}>Increment</button>
      <button onClick={() => setCount(count - 1)}>Decrement</button>
    </div>
  );
}

export default App;
'''
            }
        }
    }

    stage('Install Dependencies') {
        steps {
            dir('counter-app') {
                bat 'npm install'
            }
        }
    }

    stage('Build React App') {
        steps {
            dir('counter-app') {
                bat 'npm run build'
            }
        }
    }

    stage('React Counter App Ready') {
        steps {
            echo 'Counter App build complete! Open "counter-app/build/index.html" or serve it
with "npm start" to test.'
        }
    }
```

```
    }
}
```

17,

18.

React Calendar Picker Calendar UI Pipeline Select and display date

```
pipeline {
    agent any

    stages {
        stage('Create Calendar Picker App') {
            steps {
                script {
                    writeFile file: 'CalendarPicker.java', text: '''
import java.time.LocalDate;
import java.time.format.DateTimeFormatter;

public class CalendarPicker {
    public static void main(String[] args) {
        System.out.println("=== Calendar Picker App ===");

        // Predefined selected dates
        LocalDate[] selectedDates = {
            LocalDate.of(2025, 9, 11),
            LocalDate.of(2025, 10, 5),
            LocalDate.of(2025, 12, 25)
        };

        DateTimeFormatter formatter = DateTimeFormatter.ofPattern("dd MMM yyyy");

        // Display selected dates
        for(LocalDate date : selectedDates) {
            System.out.println("Selected Date: " + date.format(formatter));
        }

        System.out.println("=== End of Calendar Picker ===");
    }
}
'''
```

```
            }
        }
    }

    stage('Compile Calendar Picker App') {
        steps {
            bat 'javac CalendarPicker.java'
        }
    }

    stage('Run Calendar Picker App') {
        steps {
            bat 'java CalendarPicker'
        }
    }
  }
}
```

19.

20.

React Online shopping bill Calculator	Jenkins notify deploy	Pipeline	Deploy alerts sent

```
pipeline {
  agent any

  stages {
    stage('Create Bill Calculator App') {
      steps {
        script {
          writeFile file: 'ShoppingBillCalculator.java', text: '''
import java.util.HashMap;

public class ShoppingBillCalculator {
  public static void main(String[] args) {
    System.out.println("=== Online Shopping Bill Calculator ===");

    // Predefined items and prices
    HashMap<String, Double> cart = new HashMap<>();
```

```
        cart.put("Laptop", 750.00);
        cart.put("Headphones", 50.00);
        cart.put("Mouse", 25.50);
        cart.put("Keyboard", 45.25);

        double total = 0;

        System.out.printf("%-15s %-10s%n", "Item", "Price");
        System.out.println("-------------------------");

        for(String item : cart.keySet()) {
            double price = cart.get(item);
            total += price;
            System.out.printf("%-15s $%-9.2f%n", item, price);
        }

        System.out.println("-------------------------");
        System.out.printf("Total Bill: $%.2f%n", total);

        // Simulate deploy/notification alert
        System.out.println("=== Deployment Notification ===");
        System.out.println("Shopping Bill Calculator deployed successfully!");
        System.out.println("================================");
    }
}
'''
            }
        }
    }

    stage('Compile Bill Calculator') {
        steps {
            bat 'javac ShoppingBillCalculator.java'
        }
    }

    stage('Run Bill Calculator') {
        steps {
            bat 'java ShoppingBillCalculator'
        }
    }
  }
}
```