# **Tax Calculation Application Specification Document**

## **Product Overview**

The Tax Calculation Application is designed to calculate and manage tax details for properties and vehicles. The application provides a user-friendly interface for administrators to add, view, and compute tax-related data. This software ensures efficient handling of tax details, offering comprehensive summaries and organized data storage.

**Features and Capabilities**

1. **User Authentication**
   * Secure login mechanism with a predefined username and password.
   * Only authenticated users can access the application functionalities.
2. **Property Tax Management**
   * Add property details, including base value, built-up area, age, and location.
   * Calculate property tax based on the location and property attributes.
   * Display all property details, including computed tax values.
3. **Vehicle Tax Management**
   * Add vehicle details, such as brand, velocity, and seat capacity.
   * Calculate vehicle tax based on fuel type and purchase cost.
   * Display all vehicle details, including computed tax values.
4. **Total Tax**
   * Generate a summary report showing total tax collected from properties and vehicles.
   * Display quantities and tax breakdown for both categories.
5. **Error Handling**
   * Custom exception handling for invalid inputs and operations.
   * Input validation to ensure accurate data entry.

**Application Flow**

1. **User Authentication**
   * Launch the application.
   * Enter the username and password.
   * On successful login, proceed to the Main Menu.
2. **Main Menu**
   * Options available:
     1. Property Tax Menu
     2. Vehicle Tax Menu
     3. Total Tax
     4. Exit
3. **Property Tax Menu**
   * Add property details.
   * Calculate tax for a specific property by ID.
   * Display all property details.
   * Return to the Main Menu.
4. **Vehicle Tax Menu**
   * Add vehicle details.
   * Calculate tax for a specific vehicle by ID and fuel type.
   * Display all vehicle details.
   * Return to the Main Menu.
5. **Total Tax**
   * Display the total number of properties and vehicles.
   * Show total tax collected from each category.
   * Provide a cumulative tax summary.
6. **Exit**
   * Close the application.

**Sprint Planning**

**Sprint 1: Application Setup and Core Features**

**Duration**: 3 Days

**Goals:**

1. Implement the user authentication module.
2. Create the base Tax class and its subclasses (PropertyTax, VehicleTax).
3. Implement property and vehicle tax calculation logic.
4. Set up menus for property and vehicle management.

**Deliverables:**

* Functional Main Menu.
* Property and Vehicle tax computation working correctly.

**Sprint 2: Enhanced Features and Summary Reporting**

**Duration**: 5 Days

**Goals:**

1. Add display functionalities for properties and vehicles.
2. Develop the summary report generation module.
3. Implement robust error handling and validation.
4. Test and optimize the application for performance.

**Deliverables:**

* Fully operational application with error handling.
* Total Tax module completed.

**Sprint 3: Finalization and Documentation**

**Duration**: 4 Days

**Goals:**

1. Conduct comprehensive testing and debugging.
2. Finalize documentation for end-users and developers.
3. Package the application for deployment.

**Deliverables:**

* Completed and tested application.
* User and developer documentation.

**Java Concepts Used**

1. **OOP Principles**
   * Inheritance and Polymorphism: Base Tax class with specialized subclasses.
   * Encapsulation: Private attributes with getter methods.
2. **Exception Handling**
   * Custom exceptions to handle invalid inputs.
3. **Data Structures**
   * Lists for storing property and vehicle objects.
   * Streams for sorting and summarizing data.
4. **Java Utilities**
   * Use of Scanner for user input.
   * Collectors and Streams for efficient data handling.
5. **Control Structures**
   * Switch-case for menu navigation.
   * Loops for iterative tasks in menus.

# **Java Program Implementation**

**import** java.util.\*;

**import** java.util.stream.Collectors;

// Custom Exceptions

**class** InvalidInputException **extends** Exception {

**public** InvalidInputException(String message) {

**super**(message);

}

}

// Base Tax Class

**abstract** **class** Tax {

**protected** **static** **int** *idCounter* = 1;

**protected** **int** id;

**protected** **double** tax;

**public** Tax() {

**this**.id = *idCounter*++;

**this**.tax = 0.0; // Initially, the tax is not calculated

}

**public** **abstract** **void** calculateTax();

**public** **abstract** **void** displayDetails();

**public** **double** getTax() {

**return** tax;

}

}

// PropertyTax Class

**class** PropertyTax **extends** Tax {

**private** **int** baseValue;

**private** **int** builtUpArea;

**private** **int** age;

**private** **char** location;

**public** PropertyTax(**int** baseValue, **int** builtUpArea, **int** age, **char** location) {

**super**();

**this**.baseValue = baseValue;

**this**.builtUpArea = builtUpArea;

**this**.age = age;

**this**.location = location;

}

@Override

**public** **void** calculateTax() {

**if** (location == 'Y' || location == 'y') {

**this**.tax = (baseValue \* builtUpArea \* age) + (0.5 \* builtUpArea);

} **else** **if** (location == 'N' || location == 'n') {

**this**.tax = baseValue \* builtUpArea \* age;

}

}

@Override

**public** **void** displayDetails() {

System.***out***.printf("%5d %15d %10d %10c %10.2f\n", id, builtUpArea,baseValue, location, tax);

}

}

// VehicleTax Class

**class** VehicleTax **extends** Tax {

**private** **int** registrNo;

**private** String brand;

**private** **int** velocity;

**private** **int** seatCapacity;

**private** **int** type; // 1 for Petrol, 2 for Diesel, 3 for CNG

**private** **int** price; // Purchase cost

**public** VehicleTax(**int** registrNo, String brand, **int** velocity, **int** seatCapacity) {

**super**();

**this**.registrNo = registrNo;

**this**.brand = brand;

**this**.velocity = velocity;

**this**.seatCapacity = seatCapacity;

}

// Add a method to set type and price

**public** **void** setTypeAndPrice(**int** type, **int** price) {

**this**.type = type;

**this**.price = price;

}

**public** **void** calculateTax() {

**switch** (type) {

**case** 1 -> **this**.tax = velocity + seatCapacity + (0.01 \* price); // Petrol

**case** 2 -> **this**.tax = velocity + seatCapacity + (0.11 \* price); // Diesel

**case** 3 -> **this**.tax = velocity + seatCapacity + (0.12 \* price); // CNG

**default** -> **throw** **new** IllegalArgumentException("Invalid fuel type!");

}

}

**public** **int** getRegistrationNumber() {

**return** registrNo;

}

@Override

**public** **void** displayDetails() {

String fuelType = **switch** (type) {

**case** 1 -> "Petrol";

**case** 2 -> "Diesel";

**case** 3 -> "CNG";

**default** -> "Unknown";

};

System.***out***.printf(

"%5d %15s %10d %10d %10s %10d %10.2f\n",

registrNo, brand, velocity, seatCapacity, fuelType, price, tax

);

}

}

// Welcome Class

**class** Welcome {

**public** **static** **boolean** input() {

Scanner sc = **new** Scanner(System.***in***);

System.***out***.println("+-------------------------------------+");

System.***out***.println("| WELCOME TO TAX CALCULATION APP |");

System.***out***.println("+-------------------------------------+");

System.***out***.print("USERNAME: ");

String username = sc.nextLine();

System.***out***.print("PASSWORD: ");

String password = sc.nextLine();

**if** (username.equals("admin") && password.equals("admin123")) {

System.***out***.println("Authentication Successful\n");

**return** **true**;

} **else** {

System.***out***.println("Invalid Credentials. Exiting.");

**return** **false**;

}

}

}

// Main Application Class

**public** **class** AppTester {

**public** **static** **void** main(String[] args) {

Scanner sc = **new** Scanner(System.***in***);

List<PropertyTax> properties = **new** ArrayList<>();

List<VehicleTax> vehicles = **new** ArrayList<>();

**if** (!Welcome.*input*()) {

System.*exit*(0); // Exit if authentication fails

}

**while** (**true**) {

**try** {

// Main Menu

System.***out***.println("\n+-------------------------------------+");

System.***out***.println("| MAIN MENU |");

System.***out***.println("+-------------------------------------+");

System.***out***.println("1. Property Tax Menu");

System.***out***.println("2. Vehicle Tax Menu");

System.***out***.println("3. Total Tax");

System.***out***.println("4. Exit");

System.***out***.print("Select an option: ");

**int** mainChoice = sc.nextInt();

**switch** (mainChoice) {

**case** 1 -> *propertyTaxMenu*(sc, properties);

**case** 2 -> *vehicleTaxMenu*(sc, vehicles);

**case** 3 -> *total*(properties, vehicles);

**case** 4 -> {

System.***out***.println("Exiting the application. Thank you!");

System.*exit*(0);

}

**default** -> **throw** **new** InvalidInputException("Invalid choice! Please select a valid option.");

}

} **catch** (InputMismatchException e) {

System.***out***.println("Error: Invalid input type. Please enter a valid number.");

sc.nextLine(); // Clear invalid input

} **catch** (InvalidInputException e) {

System.***out***.println("Error: " + e.getMessage());

}

}

}

**private** **static** **void** propertyTaxMenu(Scanner sc, List<PropertyTax> properties) **throws** InvalidInputException {

**while** (**true**) {

System.***out***.println("\n+-------------------------------------+");

System.***out***.println("| PROPERTY TAX MENU |");

System.***out***.println("+-------------------------------------+");

System.***out***.println("1. Add Property Details");

System.***out***.println("2. Calculate Property Tax");

System.***out***.println("3. Display All Properties");

System.***out***.println("4. Back to Main Menu");

System.***out***.print("Select an option: ");

**int** choice = sc.nextInt();

**switch** (choice) {

**case** 1 -> {

System.***out***.print("Enter Base Value: ");

**int** baseValue = sc.nextInt();

System.***out***.print("Enter Built-Up Area: ");

**int** builtUpArea = sc.nextInt();

System.***out***.print("Enter Age: ");

**int** age = sc.nextInt();

System.***out***.print("Is Located in City? (Y/N): ");

**char** location = sc.next().charAt(0);

properties.add(**new** PropertyTax(baseValue, builtUpArea, age, location));

System.***out***.println("Property added successfully!");

}

**case** 2 -> {

System.***out***.print("Enter Property ID to calculate tax: ");

**int** id = sc.nextInt();

**if** (id > 0 && id <= properties.size()) {

properties.get(id - 1).calculateTax();

System.***out***.println("Property tax calculated successfully!");

} **else** {

**throw** **new** InvalidInputException("Invalid Property ID.");

}

}

**case** 3 -> {

System.***out***.println("\n+--------------------------------------------------+");

System.***out***.printf("%5s %15s %10s %10s %10s\n", "ID", "BuiltupArea", "Baseprice", "In City", "Property Tax");

System.***out***.println("+--------------------------------------------------+");

properties.forEach(PropertyTax::displayDetails);

}

**case** 4 -> {

**return**; // Back to main menu

}

**default** -> **throw** **new** InvalidInputException("Invalid choice! Please select a valid option.");

}

}

}

**private** **static** **void** vehicleTaxMenu(Scanner sc, List<VehicleTax> vehicles) **throws** InvalidInputException {

**while** (**true**) {

System.***out***.println("\n+-------------------------------------+");

System.***out***.println("| VEHICLE TAX MENU |");

System.***out***.println("+-------------------------------------+");

System.***out***.println("1. Add Vehicle Details");

System.***out***.println("2. Calculate Vehicle Tax");

System.***out***.println("3. Display All Vehicles");

System.***out***.println("4. Back to Main Menu");

System.***out***.print("Select an option: ");

**int** choice = sc.nextInt();

**switch** (choice) {

**case** 1 -> {

System.***out***.println("Enter Registration Number: ");

**int** registrNo = sc.nextInt();

sc.nextLine(); // Consume leftover newline

System.***out***.print("Enter Brand: ");

String brand = sc.nextLine();

System.***out***.print("Enter Velocity: ");

**int** velocity = sc.nextInt();

System.***out***.print("Enter Seat Capacity: ");

**int** seatCapacity = sc.nextInt();

vehicles.add(**new** VehicleTax(registrNo, brand, velocity, seatCapacity));

System.***out***.println("Vehicle added successfully!");

}

**case** 2 -> {

System.***out***.print("Enter Registration Number to calculate tax: ");

**int** registrNo1 = sc.nextInt();

VehicleTax vehicle = vehicles.stream()

.filter(v -> v.getRegistrationNumber() == registrNo1)

.findFirst()

.orElse(**null**);

**if** (vehicle != **null**) {

System.***out***.println("Choose Type: 1. Petrol 2. Diesel 3. CNG");

**int** type = sc.nextInt();

System.***out***.print("Enter Purchase Cost: ");

**int** cost = sc.nextInt();

vehicle.setTypeAndPrice(type, cost);

vehicle.calculateTax();

System.***out***.println("Vehicle tax calculated successfully!");

} **else** {

System.***out***.println("Invalid Vehicle ID.");

}

}

**case** 3 -> {

System.***out***.println("\n+--------------------------------------------------------------------------------------------+");

System.***out***.printf("%5s %15s %10s %10s %10s %10s %10s\n", "ID", "Brand", "Velocity", "Seats", "Type", "Price", "Tax");

System.***out***.println("+----------------------------------------------------------------------------------------------+");

vehicles.forEach(VehicleTax::displayDetails);

}

**case** 4 -> {

**return**; // Back to main menu

}

**default** -> **throw** **new** InvalidInputException("Invalid choice! Please select a valid option.");

}

sc.close();

}

}

**private** **static** **void** total(List<PropertyTax> properties, List<VehicleTax> vehicles) {

**double** totalPropertyTax = properties.stream().mapToDouble(PropertyTax::getTax).sum();

**double** totalVehicleTax = vehicles.stream().mapToDouble(VehicleTax::getTax).sum();

System.***out***.println("\n+--------------------------------------------------+");

System.***out***.printf("%5s %15s %10s %10s\n", "SR No", "Particular", "Quantity", "Tax");

System.***out***.println("+--------------------------------------------------+");

System.***out***.printf("%5d %15s %10d %10.2f\n", 1, "Properties", properties.size(), totalPropertyTax);

System.***out***.printf("%5d %15s %10d %10.2f\n", 2, "Vehicles", vehicles.size(), totalVehicleTax);

System.***out***.printf("%5s %15s %10d %10.2f\n", "", "Total", properties.size() + vehicles.size(), totalPropertyTax + totalVehicleTax);

}

}