```python
# Mock database containing employee pay rates using Employee ID as the primary key
database = {
    101: {"first_name": "Alice", "last_name": "Smith", "pay_rate": 30},
    102: {"first_name": "Bob", "last_name": "Johnson", "pay_rate": 25},
    103: {"first_name": "Charlie", "last_name": "Brown", "pay_rate": 28},
    104: {"first_name": "David", "last_name": "Wilson", "pay_rate": 22},
    105: {"first_name": "Eve", "last_name": "Miller", "pay_rate": 26},
    106: {"first_name": "Frank", "last_name": "Taylor", "pay_rate": 24},
    107: {"first_name": "Grace", "last_name": "Anderson", "pay_rate": 27},
    108: {"first_name": "Heidi", "last_name": "Thomas", "pay_rate": 29},
    109: {"first_name": "Ivan", "last_name": "Jackson", "pay_rate": 31},
    110: {"first_name": "Judy", "last_name": "White", "pay_rate": 23},
}

# Constants
STATE_TAX_RATE = 0.056  # 5.6% state tax
FEDERAL_TAX_RATE = 0.079  # 7.9% federal tax
DEPENDENT_DEDUCTION = 25  # $25 per dependent
OVERTIME_RATE = 1.5  # Overtime rate for hours over 40


# Function to calculate the gross pay, overtime, and tax amounts
def calculate_pay(employee_id, hours_worked, num_dependents):
    # Lookup employee info from the database using employee ID
    employee = database.get(employee_id)
    if not employee:
        print(f"Error: Employee with ID {employee_id} not found.")
        return

    first_name = employee["first_name"]
    last_name = employee["last_name"]
    pay_rate = employee["pay_rate"]

    # Calculate gross pay
    if hours_worked > 40:
        regular_pay = 40 * pay_rate
        overtime_pay = (hours_worked - 40) * pay_rate * OVERTIME_RATE
    else:
        regular_pay = hours_worked * pay_rate
        overtime_pay = 0

    gross_pay = regular_pay + overtime_pay

    # Calculate dependent deduction
    dependent_deduction = num_dependents * DEPENDENT_DEDUCTION

    # Calculate pre-tax amount
    pre_tax_amount = gross_pay - dependent_deduction

    # Calculate taxes
    state_tax = pre_tax_amount * STATE_TAX_RATE
    federal_tax = pre_tax_amount * FEDERAL_TAX_RATE

    # Calculate post-tax amount (net pay)
    post_tax_amount = pre_tax_amount - (state_tax + federal_tax)

    # Display payroll information
    print(f"Payroll details for {first_name} {last_name} (ID: {employee_id})")
    print(f"Hours Worked: {hours_worked}")
    print(f"Regular Pay: ${regular_pay:.2f}")
    print(f"Overtime Pay: ${overtime_pay:.2f}")
    print(f"Gross Pay: ${gross_pay:.2f}")
    print(f"Dependents: {num_dependents}")
    print(f"Dependent Deduction: ${dependent_deduction:.2f}")
    print(f"Pre-Tax Amount: ${pre_tax_amount:.2f}")
    print(f"State Tax: ${state_tax:.2f}")
    print(f"Federal Tax: ${federal_tax:.2f}")
    print(f"Post-Tax Amount (Net Pay): ${post_tax_amount:.2f}")
    print("=" * 50)


# Main program loop to input employee details and calculate payroll
while True:
    print("Processing Employee")

    # Input employee details
    employee_id = int(input("Enter employee ID: "))
    hours_worked = float(input("Enter hours worked: "))
    num_dependents = int(input("Enter number of dependents: "))

    # Calculate and display payroll
    calculate_pay(employee_id, hours_worked, num_dependents)

    # Ask if the user wants to look up another employee
    lookup_another = input("Do you want to look up another employee? (yes/no): 
").lower()

    # Exit the loop if the user does not want to look up another employee
    if lookup_another != 'yes':
        print("Payroll Processing Complete")
        break

print("Thank You for Using!")
```