# Security

* To log into the system, the users will use their employee IDs and 4-digit passwords
  + Only an authorized user can access the system through the login screen
  + When a new employee is added, the employee will be added as a user to the database with very limited permissions
    - The password a user enters will be hashed and compared to the one from the database
* Encryption/Decryption
  + Passwords will be hashed and stored as such with SHA256 (no salt)
  + Any sensitive employee information (SSN) will be encrypted/decrypted with AES
* Based on the permissions a user has for payroll, it will determine what buttons are clickable

# Hardware/Architecture

* Program will work on Windows 10/11
* AWS RDS for the database
* Internet required to use the program
* 2-tier client server
  + Diagram

    Description automatically generated

# User Interface

* To go to a previous screen, the user will click on the X on the top right of the program
  + If on the main payroll screen, the program will quit
  + When exiting a form, ensure it is closed completely and reshow the parent form
* In C#, we will use forms to traverse the application
  + When a new form is created, hide the current form, and show the new form
* Buttons will be used to navigate between the forms
* Each input box will have a descriptive name for it somewhere close to it
* When selecting multiple employees, the ones selected will be highlighted

# Coding Guidelines

* Use getters and setters for variables
* Any private variables will start with underscore (\_)
  + For example, \_private
* Variable and method names will be in camel case fashion
* Try-catch blocks and throwing exceptions will be used
* GitHub pushes will be used
  + Code will be reviewed each meeting
* Comments will be used to describe what a method \*should\* do and any other complex part of the code that needs it
* Unit testing should be done before each test for each method possible

# Classes

1. Properties
   1. Employee (Abstract)
      1. First Name
      2. Last Name
      3. Address
      4. City
      5. Zip Code [5]
      6. State [2]
      7. Bank Account Routing Number [9]
      8. Bank Account Number [17]
      9. Bank Name
      10. Employee ID [2 or 3?]
      11. Federal Tax Rate [XX.XX]
      12. Permissions (true or false)
      13. SSN [9]
      14. DOB [MMDDYYYY]
      15. Hire Date [MMDDYYYY]
      16. Phone Number [10]
      17. Department
      18. Deduction List
      19. Net Pay [99999.99]
   2. Hourly Employee
      1. Pay Per Hour [999.99]
      2. Hours Worked [99.99]
   3. Salary Employee
      1. Salary Per Pay Period [99999.99]
   4. Employee List
      1. Employee List
      2. Company
   5. Company
      1. Bank Account Routing Number [9]
      2. Bank Account Number [17]
      3. Bank Name
      4. Company Name
      5. Federal ID [9]
      6. Address
      7. City
      8. State [2]
      9. Zip Code [5]
      10. Phone Number [10]
   6. Deduction (Abstract)
      1. Name
   7. Flat Deduction
      1. Flat [999.99]
   8. Percentage Deduction
      1. Percentage [99.99%]
2. Methods
   1. Employee
      1. calculateGrossPay
      2. calculateFederalTax
      3. calculateStateTax
      4. calculateFICATax
      5. calculateMedTax
      6. calculateTotalNonTaxDeductionAmount
      7. calculateNetPay(int grossPay)
   2. Hourly Employee
      1. calculateGrossPay
      2. getPayrollHours
   3. Salary Employee
      1. calculateGrossPay
   4. Employee List
      1. generateEmployeeList(int compFedID)
      2. displaySelectableEmployeeList
      3. calculateGrossPayAllEmployeesInList
      4. getGrossPayCombined
      5. getSizeOfList
      6. getSumOfFirst8DigitsRoute
   5. Company
   6. Deduction
      1. calculateDeductionAmount(double grossPay)
   7. Flat Deduction
      1. calculateDeductionAmount()
   8. Percentage Deduction
      1. calculateDeductionAmount(double grossPay)