PSEUDOCODE FOR THE WHOLE PROGRAM FLOW

```
Start

Initialize Database Manager
Initialize Account Manager
Initialize Login Window
Show Login Window
Start Application Event Loop

Function: login()
    Prompt for student name and number
    If student name is "Admin" and student number is "admin":
        Call open_admin_app()
    Else if student name and student number are empty:
        Display "No Input" error message
    Else if student name or student number is empty:
        Display "Missing Input" error message
    Else if credentials are valid:
        Call open_borrowing_app(student_name, student_number)
    Else:
        Display "Incorrect credentials" error message

Function: register()
    Prompt for student name and number
    If student name or student number is empty:
        Display "Name and student number cannot be blank" error message
    Else if student number is not numeric:
        Display "Student number must consist of integers only" error message
    Else if registration is successful:
        Display "Registration Successful" message
    Else:
        Display "Name or student number already exists" error message

Function: open_borrowing_app(student_name, student_number)
    Initialize BorrowingApp with student_name and student_number
    Show BorrowingApp

Function: open_admin_app()
    Initialize AdminApp
    Show AdminApp

Function: BorrowingApp.init_ui()
```

```
    Set window title and size
    Create layout
    Add title label to layout
    Add date_time_label to layout
    Add material_label to layout
    Create material_combo
    Populate material_combo with materials from DatabaseManager
    Add material_combo to layout
    Create available_label
    Add available_label to layout
    Connect material_combo.currentIndexChanged to update_available_quantity
    Create quantity_input
    Add quantity_input to layout
    Create add_button
    Connect add_button.clicked to add_material
    Add add_button to layout
    Create remove_button
    Connect remove_button.clicked to remove_material
    Add remove_button to layout
    Create finish_button
    Connect finish_button.clicked to finish_borrowing
    Add finish_button to layout
    Create materials_list
    Add materials_list to layout
    Create back_button
    Connect back_button.clicked to go_back
    Add back_button to layout

Function: BorrowingApp.update_available_quantity()
    Get current index from material_combo
    Get available quantity from material_combo data
    Update available_label text with available quantity

Function: BorrowingApp.add_material()
    If number of materials in list >= 12:
        Display "Limit Reached" error message
        Return
    Get material name from material_combo
    Get quantity from quantity_input
    If material name is valid and in DatabaseManager:
        Get available quantity from DatabaseManager
        If quantity > available quantity:
            Display "Quantity Error" message
            Return
```

```
        Add material to materials list
        Update materials_list display
        Decrease available quantity in DatabaseManager
        Refresh material_combo
        Update available quantity label
        Reset quantity_input to 1

Function: BorrowingApp.remove_material()
    Get selected item from materials_list
    If selected item:
        Get material name from selected item
        Remove material from materials list
        Remove selected item from materials_list display
    Else:
        Display "Selection Error" message

Function: BorrowingApp.finish_borrowing()
    If materials list is empty:
        Display "Input Error" message
        Return
    Get current time in Philippine Standard Time
    Prepare borrowing information string
    Display borrowing information in message box
    Call log_borrowing(current_time)
    Call update_database()
    Call clear_inputs()

Function: BorrowingApp.log_borrowing(current_time)
    Open 'log.csv' in append mode
    If log file is empty:
        Write header to log file
    Prepare materials_borrowed string
    Write student name, student number, current time, materials_borrowed to log file

Function: BorrowingApp.update_database()
    For each material in materials list:
        Decrease quantity in DatabaseManager
        If quantity < 0:
            Set quantity to 0
    Save updated materials to DatabaseManager

Function: BorrowingApp.clear_inputs()
    Clear materials list
    Clear materials_list display
```

```
    Clear material_combo
    Populate material_combo with materials from DatabaseManager
    Reset quantity_input to 1

Function: BorrowingApp.go_back()
    Close BorrowingApp
    Initialize LoginWindow
    Show LoginWindow

Function: AdminApp.init_ui()
    Set window title and size
    Create layout
    Add title label to layout
    Create material_name_input
    Add material_name_input to layout
    Create quantity_input
    Add quantity_input to layout
    Create add_button
    Connect add_button.clicked to add_update_material
    Add add_button to layout
    Create remove_button
    Connect remove_button.clicked to remove_material
    Add remove_button to layout
    Create materials_list
    Call update_materials_list()
    Add materials_list to layout
    Create back_button
    Connect back_button.clicked to go_back
    Add back_button to layout

Function: AdminApp.add_update_material()
    Get material name from material_name_input
    Get quantity from quantity_input
    If material name is valid:
        Add or update material in DatabaseManager
        Call update_materials_list()
        Clear material_name_input
        Reset quantity_input to 1

Function: AdminApp.remove_material()
    Get selected item from materials_list
    If selected item:
        Get material name from selected item
        Remove material from DatabaseManager
```

```
        Call update_materials_list()

Function: AdminApp.update_materials_list()
    Clear materials_list
    For each material in DatabaseManager:
        Add material to materials_list

Function: AdminApp.go_back()
    Close AdminApp
    Initialize LoginWindow
    Show LoginWindow

End
```

PSEUDOCODE FOR EACH CLASS (For Procedure Part as Guide for Coding)

- **Material Class**

```
Class Material
    Method __init__(name, quantity)
        Set self.name = name
        Set self.quantity = quantity
End Class
```

- **Account Manager Class**

```
Class AccountManager
    Method __init__(filename='accounts.txt')
        Set self.filename = filename
    End Method

    Method register(name, student_number)
        Open self.filename in append mode
        For each line in file
            Split line into existing_name, existing_number
            If existing_name == name OR existing_number == student_number
                Return False
        Write name, student_number to file
        Return True
    End Method

    Method login(name, student_number)
        Open self.filename in read mode
        For each line in file
            Split line into existing_name, existing_number
            If existing_name == name AND existing_number == student_number
                Return True
        Return False
    End Method
End Class
```

- **Database Manager Class**

```
Class DatabaseManager
    Method __init__(filename='database.txt')
        Set self.filename = filename
        Set self.materials = load_materials()
    End Method

    Method load_materials()
        Set materials = empty dictionary
        Try
            Open self.filename in read mode
```

```
            For each line in file
                Split line into name, quantity
                Add name, quantity to materials
        Except FileNotFoundError
            Pass
        Return materials
    End Method

    Method save_materials()
        Open self.filename in write mode
        For each name, quantity in self.materials
            Write name, quantity to file
    End Method

    Method add_material(name, quantity)
        Add name, quantity to self.materials
        Call save_materials()
    End Method

    Method remove_material(name)
        If name in self.materials
            Delete name from self.materials
            Call save_materials()
    End Method
End Class
```

- **Borrowing App / Window Class**

```
Class BorrowingApp Inherits QWidget
    Method __init__(student_name, student_number)
        Call super().__init__()
        Set self.student_name = student_name
        Set self.student_number = student_number
        Set self.materials = empty list
        Set self.db_manager = DatabaseManager()
        Call init_ui()
    End Method

    Method init_ui()
```

```
        Set window title and size
        Create layout
        Add title label to layout
        Add date_time_label to layout
        Add material_label to layout
        Create material_combo
        Populate material_combo with materials from DatabaseManager
        Add material_combo to layout
        Create available_label
        Add available_label to layout
        Connect material_combo.currentIndexChanged to update_available_quantity
        Create quantity_input
        Add quantity_input to layout
        Create add_button
        Connect add_button.clicked to add_material
        Add add_button to layout
        Create remove_button
        Connect remove_button.clicked to remove_material
        Add remove_button to layout
        Create finish_button
        Connect finish_button.clicked to finish_borrowing
        Add finish_button to layout
        Create materials_list
        Add materials_list to layout
        Create back_button
        Connect back_button.clicked to go_back
        Add back_button to layout
    End Method

    Method update_available_quantity()
        Get current index from material_combo
        Get available quantity from material_combo data
        Update available_label text with available quantity
    End Method

    Method add_material()
        If number of materials in list >= 12
            Display "Limit Reached" error message
            Return
        Get material name from material_combo
```

```
        Get quantity from quantity_input
        If material name is valid and in DatabaseManager
            Get available quantity from DatabaseManager
            If quantity > available quantity
                Display "Quantity Error" message
                Return
            Add material to materials list
            Update materials_list display
            Decrease available quantity in DatabaseManager
            Refresh material_combo
            Update available quantity label
            Reset quantity_input to 1
    End Method

    Method remove_material()
        Get selected item from materials_list
        If selected item
            Get material name from selected item
            Remove material from materials list
            Remove selected item from materials_list display
        Else
            Display "Selection Error" message
    End Method

    Method finish_borrowing()
        If materials list is empty
            Display "Input Error" message
            Return
        Get current time in Philippine Standard Time
        Prepare borrowing information string
        Display borrowing information in message box
        Call log_borrowing(current_time)
        Call update_database()
        Call clear_inputs()
    End Method

    Method log_borrowing(current_time)
        Open 'log.csv' in append mode
        If log file is empty
            Write header to log file
```

```
        Prepare materials_borrowed string
        Write student name, student number, current time, materials_borrowed to log file
    End Method

    Method update_database()
        For each material in materials list
            Decrease quantity in DatabaseManager
            If quantity < 0
                Set quantity to 0
        Save updated materials to DatabaseManager
    End Method

    Method clear_inputs()
        Clear materials list
        Clear materials_list display
        Clear material_combo
        Populate material_combo with materials from DatabaseManager
        Reset quantity_input to 1
    End Method

    Method go_back()
        Close BorrowingApp
        Initialize LoginWindow
        Show LoginWindow
    End Method
End Class
```

- **Admin App / Window Class**

```
Class AdminApp Inherits QWidget
    Method __init__()
        Call super().__init__()
        Set self.db_manager = DatabaseManager()
        Call init_ui()
    End Method

    Method init_ui()
        Set window title and size
        Create layout
```

```
    Add title label to layout
    Create material_name_input
    Add material_name_input to layout
    Create quantity_input
    Add quantity_input to layout
    Create add_button
    Connect add_button.clicked to add_update_material
    Add add_button to layout
    Create remove_button
    Connect remove_button.clicked to remove_material
    Add remove_button to layout
    Create materials_list
    Call update_materials_list()
    Add materials_list to layout
    Create back_button
    Connect back_button.clicked to go_back
    Add back_button to layout
End Method

Method add_update_material()
    Get material name from material_name_input
    Get quantity from quantity_input
    If material name is valid
        Add or update material in DatabaseManager
        Call update_materials_list()
        Clear material_name_input
        Reset quantity_input to 1
End Method

Method remove_material()
    Get selected item from materials_list
    If selected item
        Get material name from selected item
        Remove material from DatabaseManager
        Call update_materials_list()
End Method

Method update_materials_list()
    Clear materials_list
    For each material in DatabaseManager
```

```
            Add material to materials_list
    End Method

    Method go_back()
        Close AdminApp
        Initialize LoginWindow
        Show LoginWindow
    End Method
End Class
```

- **Login Window Class**

```
Class LoginWindow Inherits QWidget
    Method __init__()
        Call super().__init__()
        Set self.account_manager = AccountManager()
        Call init_ui()
    End Method

    Method init_ui()
        Set window title and size
        Create layout
        Add title_frame to layout
        Add title_label to title_frame
        Add title_frame to layout
        Create name_input
        Add name_input to layout
        Create number_input
        Add number_input to layout
        Create login_button
        Connect login_button.clicked to login
        Add login_button to layout
        Create register_button
        Connect register_button.clicked to register
        Add register_button to layout
    End Method

    Method login()
        Get student_name from name_input
```

```
        Get student_number from number_input
        If student_name == "Admin" AND student_number == "admin"
            Close self
            Call open_admin_app()
        Else if student_name == "" AND student_number == ""
            Display "No Input" error message
        Else if student_name == "" OR student_number == ""
            Display "Missing Input" error message
        Else if Call login(student_name, student_number) on self.account_manager
            Close self
            Call open_borrowing_app(student_name, student_number)
        Else
            Display "Incorrect credentials" error message
    End Method

    Method register()
     Get student_name from self.name_input and strip whitespace
    Get student_number from self.number_input and strip whitespace
    If student_name is empty OR student_number is empty
        Display warning message "Name and student number cannot be blank."
    Else if student_number is not a digit
        Display warning message "Student number must consist of integers only."
    Else if Call register(student_name, student_number) on self.account_manager
        Display information message "Registration Successful"
    Else if student_name is empty AND student_number is empty
        Display warning message "No Input. Please try again."
    Else if student_name is empty OR student_number is empty
        Display warning message "Missing Input. Please try again."
    Else
        Display warning message "Name or student number already exists."
End Function


    Method open_borrowing_app(student_name, student_number)
        Initialize BorrowingApp with student_name and student_number
        Show BorrowingApp
    End Method

    Method open_admin_app()
        Initialize AdminApp
```

```
        Show AdminApp
    End Method
End Class
```

- **Main Program Execution**

```
Main Program Execution
    If this script is the main module being run:
        Initialize QApplication with command line arguments
        Create an instance of LoginWindow
        Show the LoginWindow
        Start the application event loop
        Exit the program when the event loop ends
End Main Program Execution
```