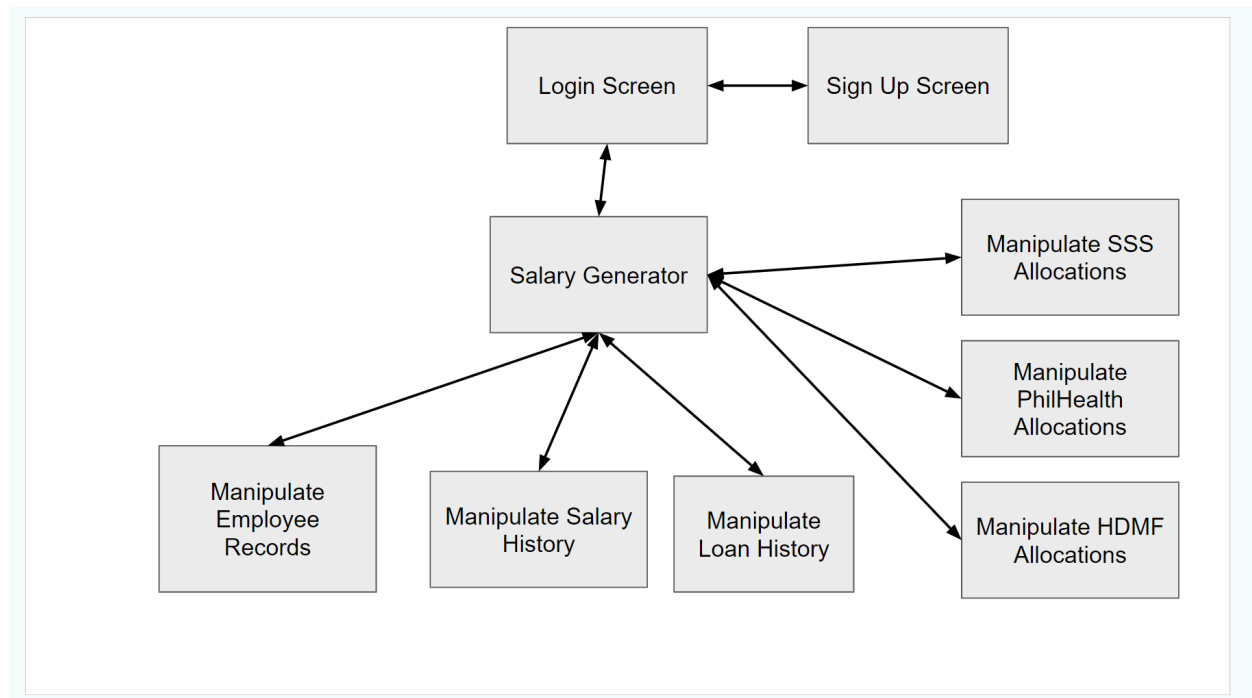# Program Structure

Description of Forms

Form:	Description:
Login	This is the starting page of the application that the user will always interact with first. Only one user can be created for the application.
Sign Up	This page takes user input to add the username and password of the single password that can use the application. It can be accessed via the login page.
Manipulate Employee Salary Data / Salary Generator	This page is the "landing page" of the application. It allows the user to automatically generate the salaries of all the employees within the system based on set parameters. This includes all benefit payments and extra fees allocated from a set starting amount. Both the parameters and individual amounts can be overwritten via user input. Once a salary is generated, it is saved to transaction history.
Manipulate Employee Records	This page allows for the user to add, edit, delete, sort, and search through employee personal records and to manipulate them with ease. It is accessed via the salary generator page.

Manipulate Salary History	The salary history contains every salary paid out to all employees using the system. The data values can be manipulated.
Manipulate Loan History	The loan history contains every payment or withdrawal made to an employee's loan using the system. The user can add, edit, delete, search, and sort through data values.
Manipulate SSS Allocations	It allows the user to only add, edit and delete individual values for the SSS allocations based on the Employer (ER) and Employee (EE) (Appendix B), which automatically update the SSS allocations of individual employees.
Manipulate PhilHealth Allocations	It allows the user to only add, edit and delete individual values for the PhilHealth allocations. It is usually 2.5% employee share, and 2.5% employer share for all wage levels. However, it can change. It is based on PhilHealth Contribution Premium (Appendix B)
Manipulate HDMF Allocations	It allows the user to only add, edit and delete individual values for the HDMF allocations based on the Circular No.460 (Appendix B)

Entity-relationship Diagram
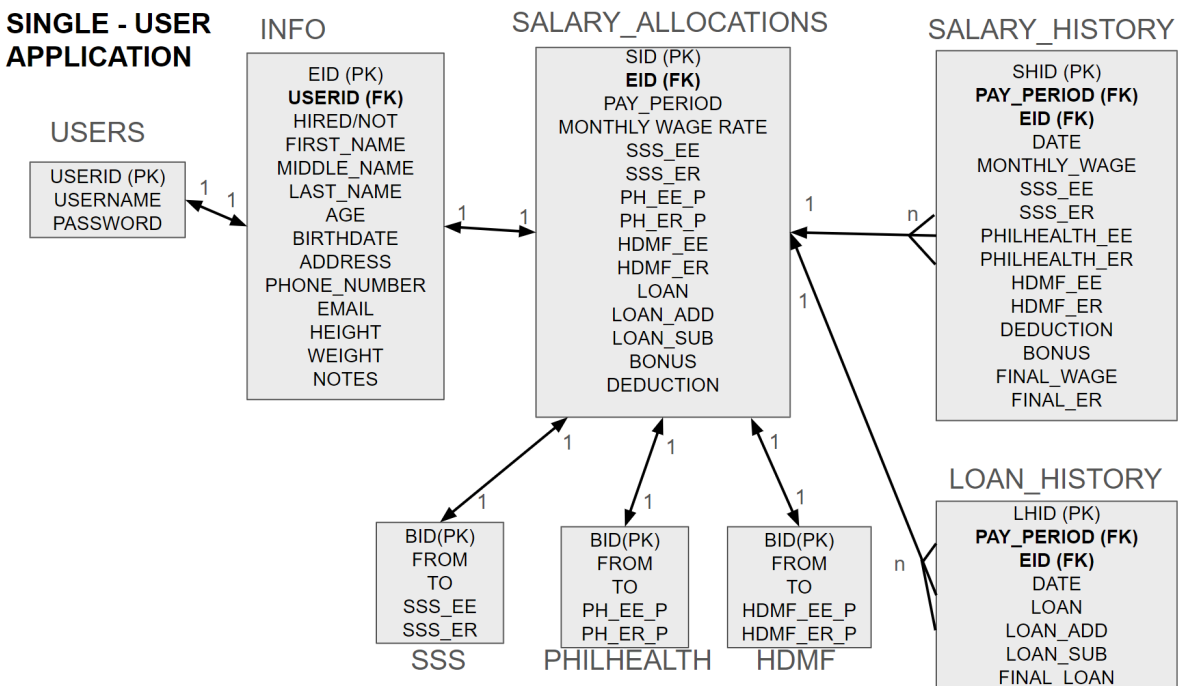
Legend:
PK: Primary Key
FK: Foreign Key
SSS: Social Security System
HDMF: Home Development Mutual Fund

Database Structure

USERS

Field Name	Data Type	Description
USERID	integer	Primary Key
USERNAME	string, varchar(100)	Username of the single-user
PASSWORD	string, varchar(100)	Password of the single-user

SSS

Field Name	Data Type	Description
BID	integer	Bracket ID, Primary Key
FROM	decimal(8,2)	Lower bound (inclusive) of bracket.
TO	decimal(8,2)	Upper bound (inclusive) of bracket.
SSS_EE	decimal(8,2)	Amount taken from employee's salary paid to the employee's SSS fund.
SSS_ER	decimal(8,2)	Amount paid by employer to employee's SSS fund.

PHILHEALTH

Field Name	Data Type	Description
BID	integer	Bracket ID, Primary Key
FROM	decimal(8,2)	Lower bound (inclusive) of bracket.
TO	decimal(8,2)	Upper bound (inclusive) of bracket.
PH_EE_P	decimal(8,2)	Percentage of employee's

		salary paid to the employee's PhilHealth fund.
PH_ER_P	decimal(8,2)	Percentage of employee's salary paid by employer to employee's PhilHealth fund.

HDMF

Field Name	Data Type	Description
BID	integer	Bracket ID, Primary Key
FROM	decimal(8,2)	Lower bound (inclusive) of bracket
TO	decimal(8,2)	Upper bound (inclusive) of bracket
PH_EE_P	decimal(8,2)	Percentage of employee's salary paid to the employee's HDMF fund.
PH_ER_P	decimal(8,2)	Percentage of employee's salary paid by employer to employee's HDMF fund.

INFO

Field Name	Data Type	Description
EID	integer	Employee ID, Primary Key
USERID	integer	Foreign Key
HIRED/NOT	boolean	True if hired, false if not hired.
FIRST_NAME	string, varchar(50)	First name of the employee
MIDDLE_NAME	string, varchar(50)	Middle name of the employee
LAST_NAME	string, varchar(50)	Last name of the employee
AGE	integer	Age of the employee
BIRTHDATE	datetime	Birthdate of the employee

ADDRESS	string, varchar(100)	Address of the employee
PHONE_NUMBER	string, varchar(50)	Phone number of the employee
EMAIL	string, varchar(50)	Email of the employee
HEIGHT	integer	Height of the employee
WEIGHT	integer	Weight of the employee
NOTES	string, varchar(200)	Notes regarding the employee

SALARY_ALLOCATIONS

Field Name	Data Type	Description
SID	integer	Salary ID, Primary Key
EID	integer	Employee ID, Foreign Key
PAY_PERIOD	integer	Increments up by 1 for every succeeding pay period.
MONTHLY_WAGE_RATE	decimal(8,2)	The monthly wage rate of the employee. (counted as 30 days of work)
SSS_EE	decimal(8,2)	Amount taken from employee's salary paid to the employee's SSS fund.
SSS_ER	decimal(8,2)	Amount paid by employer to employee's SSS fund.
PH_EE_P	decimal(8,2)	Percentage of employee's salary paid to the employee's PhilHealth fund.
PH_ER_P	decimal(8,2)	Percentage of employee's salary paid by employer to employee's PhilHealth fund.
HDMF_EE_P	decimal(8,2)	Percentage of employee's salary paid to the employee's HDMF fund.
HDMF_ER_P	decimal(8,2)	Percentage of employee's

		salary paid by employer to employee's HDMF fund.
LOAN	decimal(8,2)	The employee's current loan from the employer.
LOAN_ADD	decimal(8,2)	Loan increase of the employee for that pay period
LOAN_SUB	decimal(8,2)	Loan payment of the employee for that pay period.
DEDUCTION	decimal(8,2)	A catch-all integer that decreases salary for any reason.
BONUS	decimal(8,2)	A catch-all integer that increases salary for any reason.

SALARY_HISTORY

Field Name	Data Type	Description
SHID	integer	Salary History ID, Primary Key
PAY_PERIOD	integer	Pay Period, Foreign Key
EID	integer	Employee ID, Foreign Key
DATESAVED	date	Date saved to the table
MONTHLY_WAGE_RATE	decimal(8,2)	The monthly wage rate of the employee. (counted as 30 days of work)
SSS_EE	decimal(8,2)	Amount taken from employee's salary paid to the employee's SSS fund.
SSS_ER	decimal(8,2)	Amount paid by employer to employee's SSS fund.
PH_EE_P	decimal(8,2)	Percentage of employee's salary paid to the employee's PhilHealth fund.

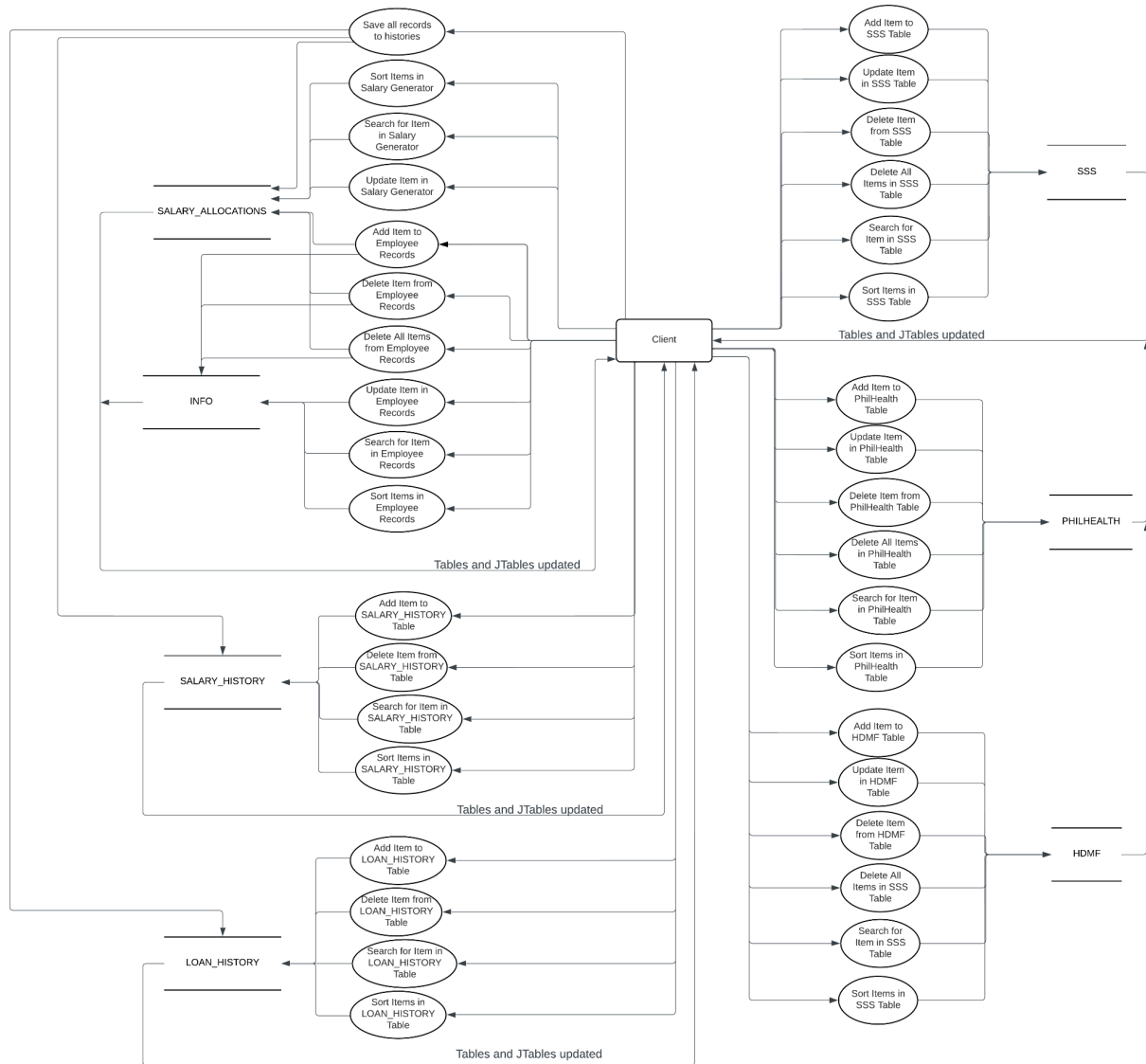| PH_ER_P	decimal(8,2)	Percentage of employee's salary paid by employer to employee's PhilHealth fund.
HDMF_EE_P	decimal(8,2)	Percentage of employee's salary paid to the employee's HDMF fund.
HDMF_ER_P	decimal(8,2)	Percentage of employee's salary paid by employer to employee's HDMF fund.
DEDUCTION	decimal(8,2)	A catch-all integer that decreases salary for any reason.
BONUS	decimal(8,2)	A catch-all integer that increases salary for any reason.
FINAL_WAGE	decimal(8,2)	The final wage of the employee when all the social security contributions have been made for that pay period.
FINAL_ER	decimal(8,2)	The total cost of social security contributions made by the employer for that employee for that pay period.

LOAN_HISTORY

Field Name	Data Type	Description
LHID	integer	Loan History ID, Primary Key
PAY_PERIOD	integer	Pay Period, Foreign Key
EID	integer	Employee ID, Foreign Key
DATESAVED	date	Date saved to the table
LOAN	decimal(8,2)	The employee's current loan from the employer.
LOAN_ADD	decimal(8,2)	Loan increase of the employee for that pay period

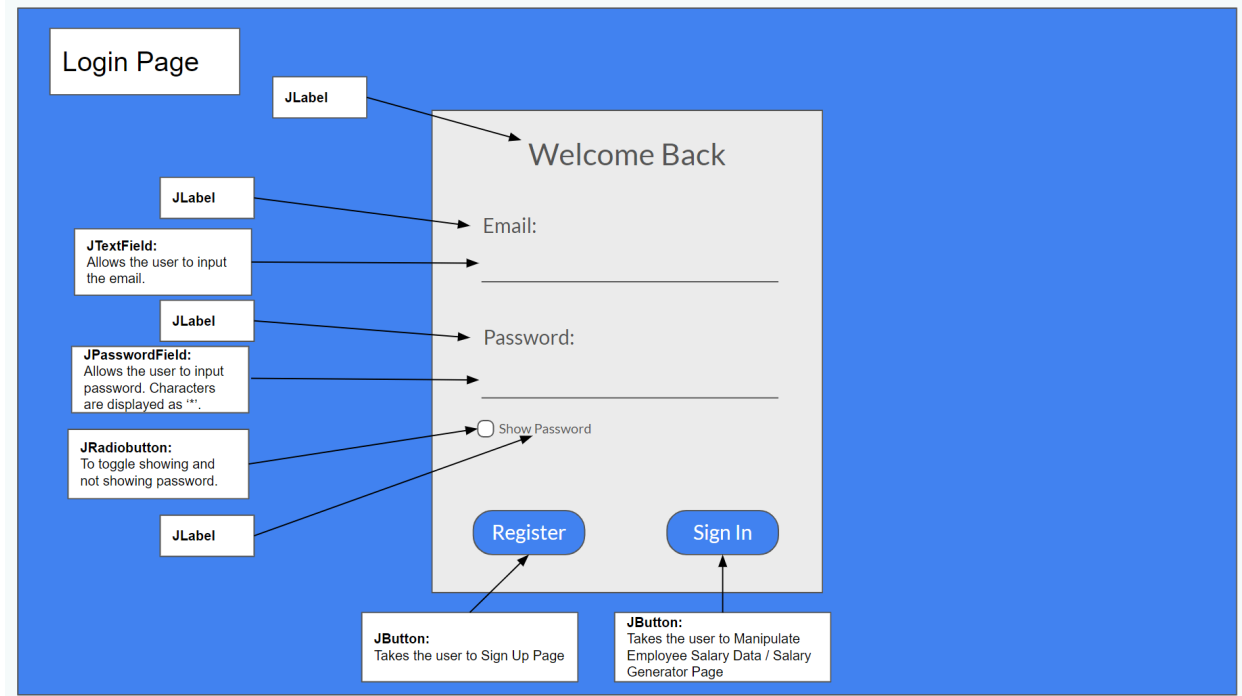|		
LOAN_SUB	decimal(8,2)	Loan payment of the employee for that pay period.
FINAL_LOAN	decimal(8,2)	The final loan amount left after the subtractions and additions during the pay period.

Data Flow Diagram (DFD)

SALARY_ALLOCATIONS

INFO

SALARY_HISTORY

LOAN_HISTORY

Save all records to histories

Sort Items in Salary Generator

Search for Item in Salary Generator

Update Item in Salary Generator

Add Item to Employee Records

Delete Item from Employee Records

Delete All Items from Employee Records

Update Item in Employee Records

Search for Item in Employee Records

Sort Items in Employee Records

Tables and JTables updated

Add Item to SALARY_HISTORY Table

Delete Item from SALARY_HISTORY Table

Search for Item in SALARY_HISTORY Table

Sort Items in SALARY_HISTORY Table

Tables and JTables updated

Add Item to LOAN_HISTORY Table

Delete Item from LOAN_HISTORY Table

Search for Item in LOAN_HISTORY Table

Sort Items in LOAN_HISTORY Table

Tables and JTables updated

Client

Add Item to SSS Table

Update Item in SSS Table

Delete Item from SSS Table

Delete All Items in SSS Table

Search for Item in SSS Table

Sort Items in SSS Table

SSS

Tables and JTables updated

Add Item to PhilHealth Table

Update Item in PhilHealth Table

Delete Item from PhilHealth Table

Delete All Items in PhilHealth Table

Search for Item in PhilHealth Table

Sort Items in PhilHealth Table

PHILHEALTH

Add Item to HDMF Table

Update Item in HDMF Table

Delete Item from HDMF Table

Delete All Items in SSS Table
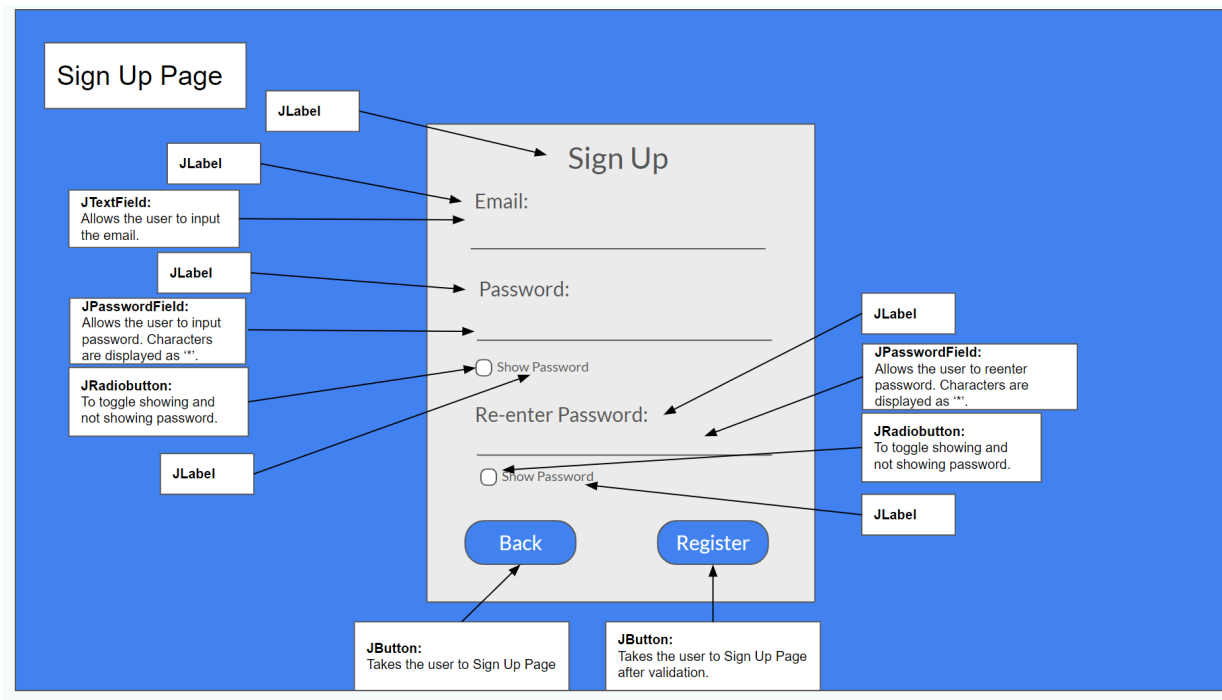
Search for Item in SSS Table
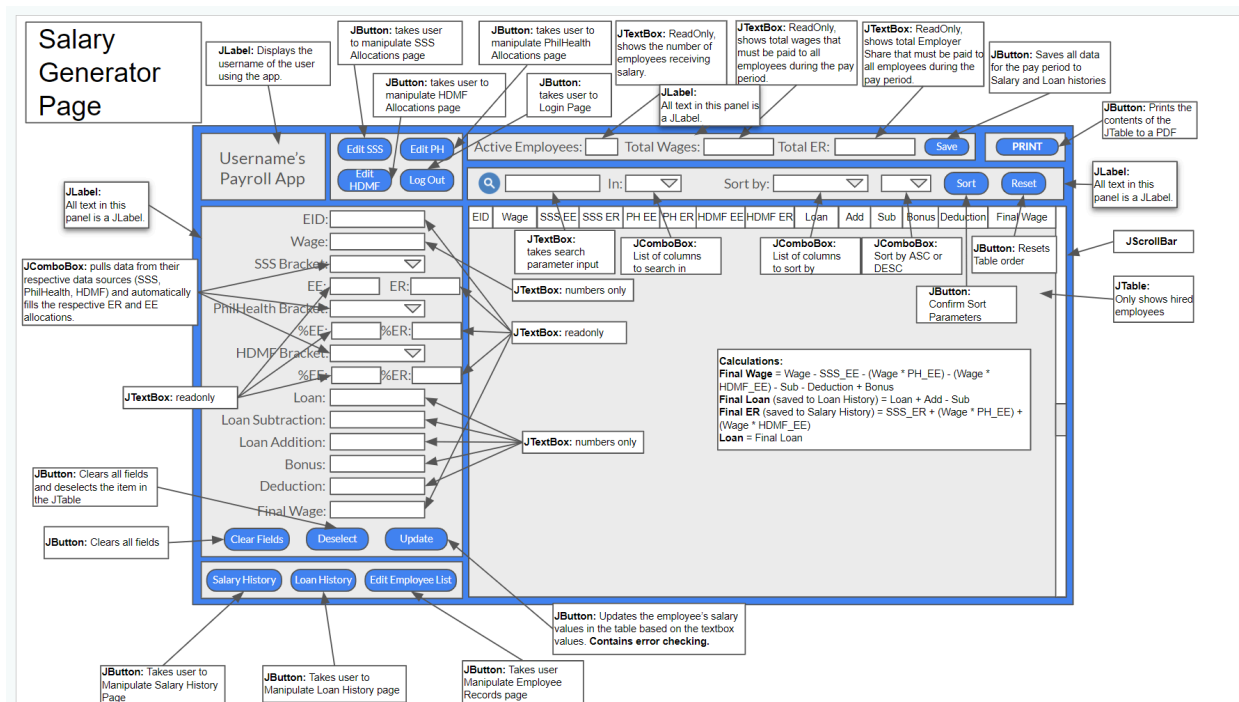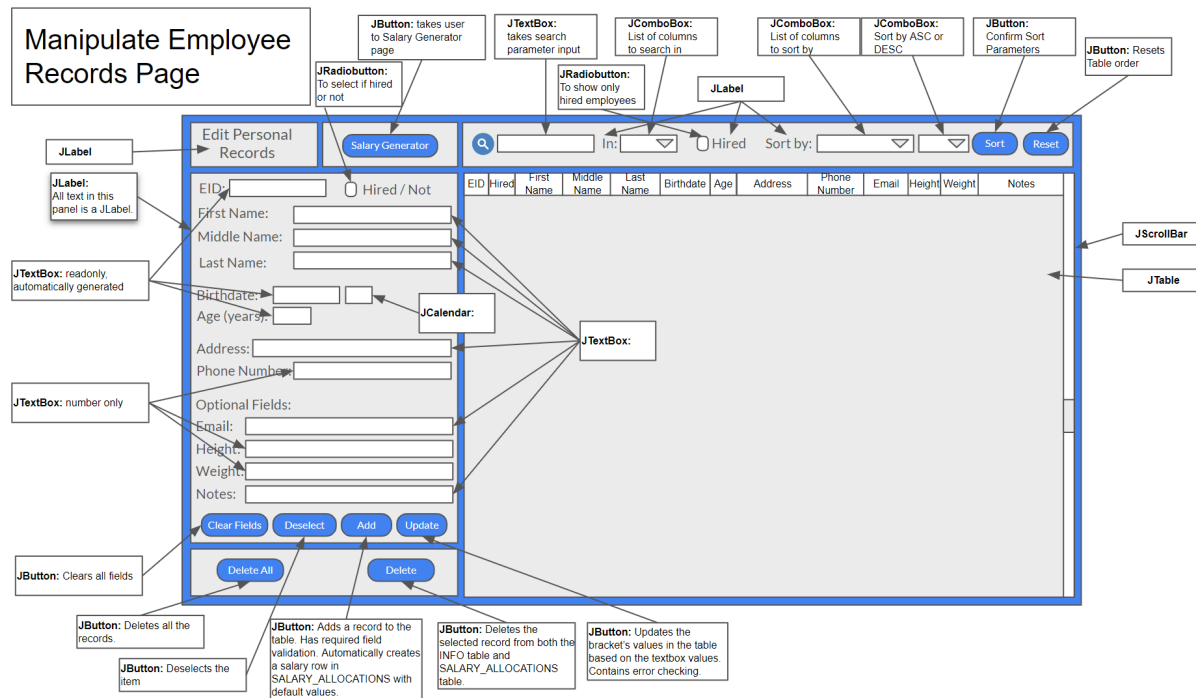
Sort Items in SSS Table

HDMF

Form Design:

Login Page:

Sign Up Page:

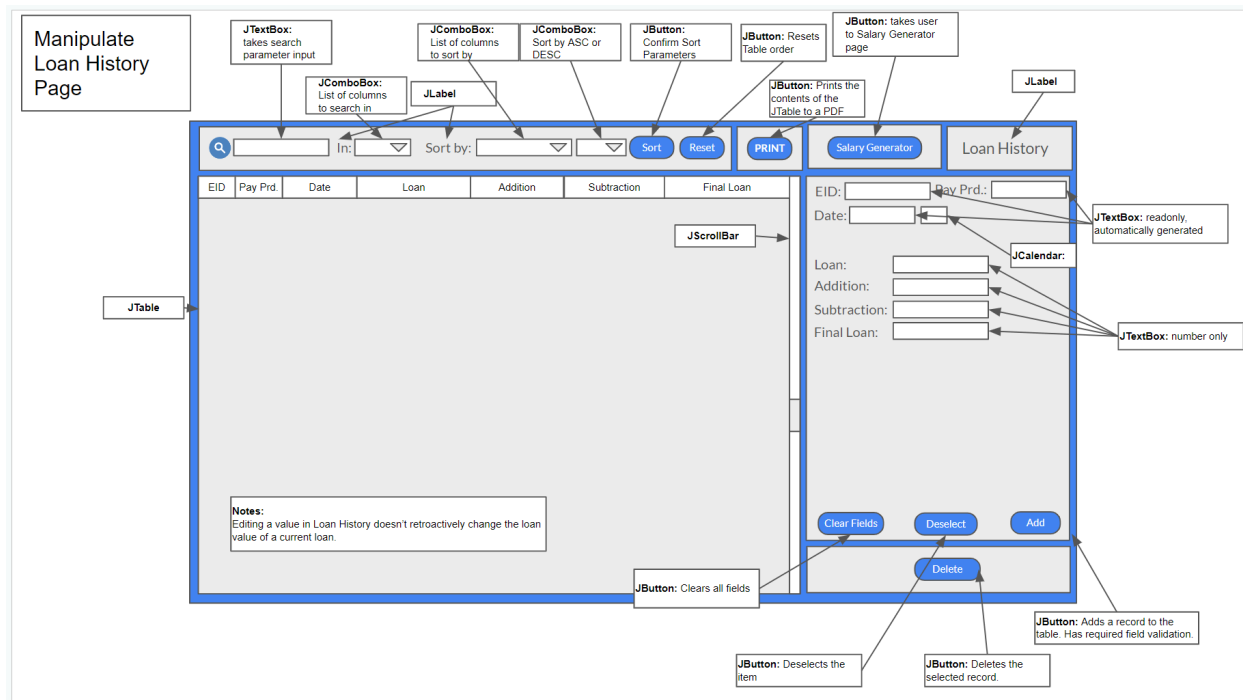# Salary Generator:

Manipulate Employee Records:

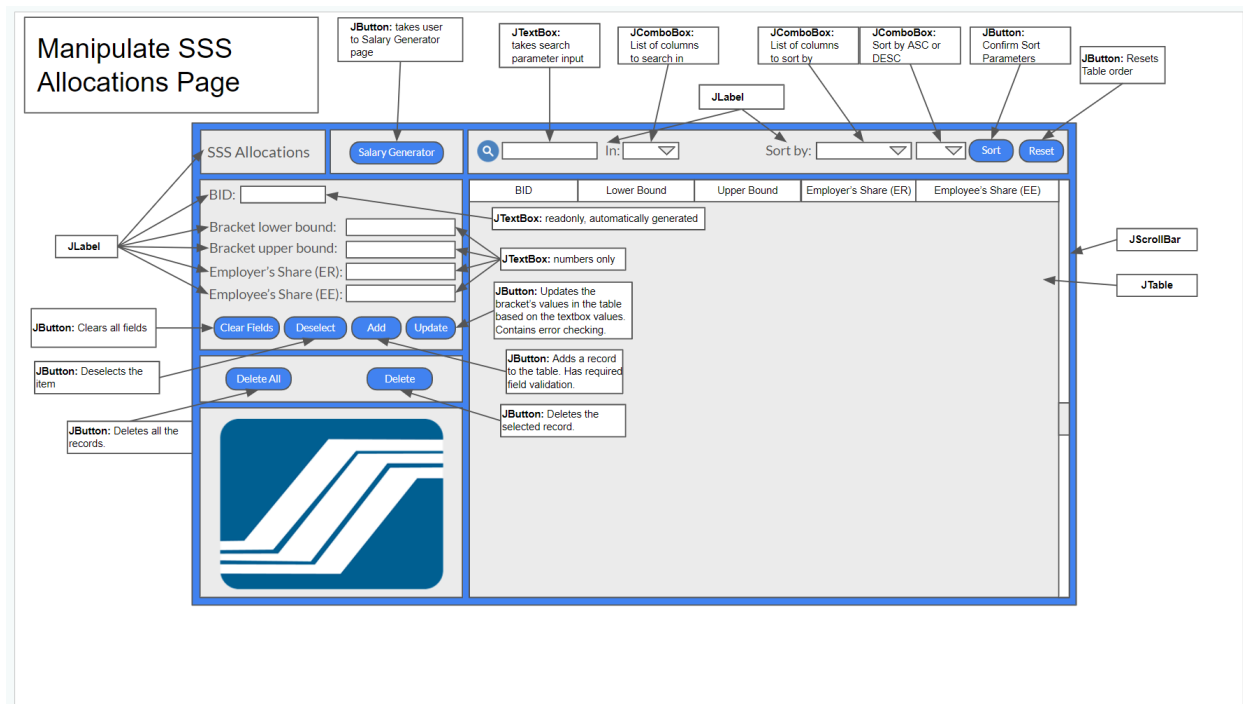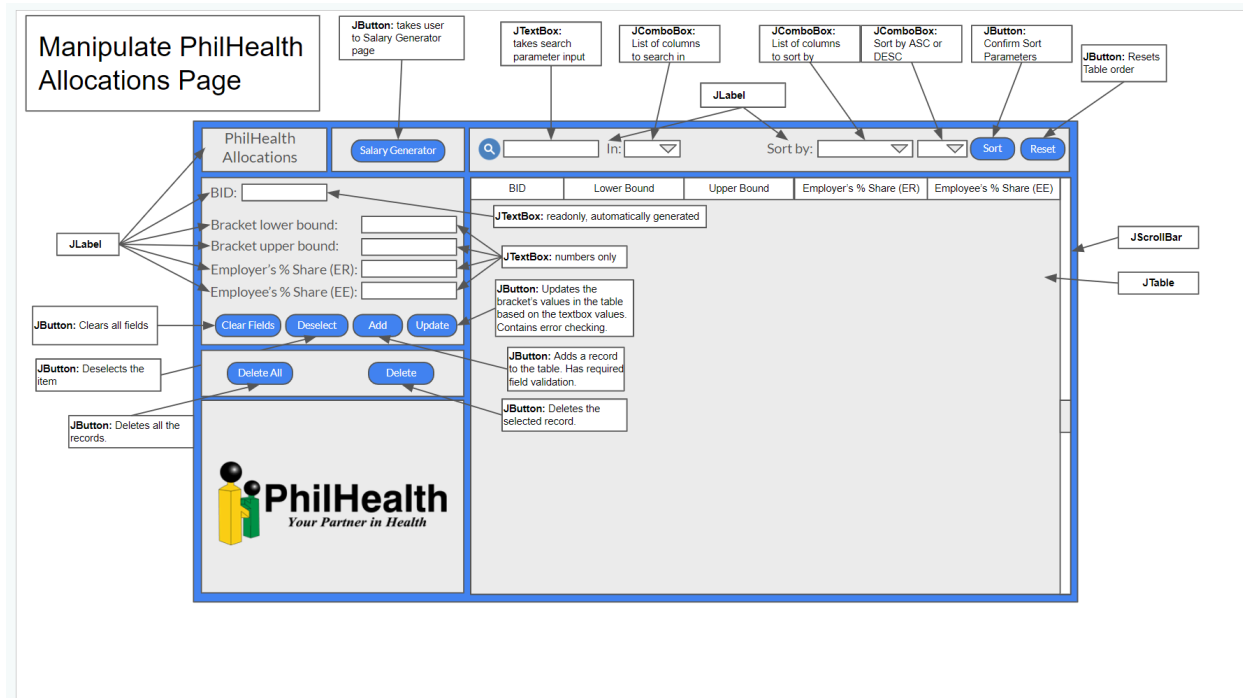**Manipulate Employee Records Page**

- **JButton:** takes user to Salary Generator page
- **JRadiobutton:** To select if hired or not
- **JTextBox:** takes search parameter input
- **JRadiobutton:** To show only hired employees
- **JComboBox:** List of columns to search in
- **JLabel**
- **JComboBox:** List of columns to sort by
- **JComboBox:** Sort by ASC or DESC
- **JButton:** Confirm Sort Parameters
- **JButton:** Resets Table order
- **JLabel**
- **JLabel:** All text in this panel is a JLabel.
- **JTextBox:** readonly, automatically generated
- **JTextBox:** number only
- **JCalendar:**
- **JTextBox:**
- **JScrollBar**
- **JTable**
- **JButton:** Clears all fields
- **JButton:** Deletes all the records.
- **JButton:** Deselects the item
- **JButton:** Adds a record to the table. Has required field validation. Automatically creates a salary row in SALARY_ALLOCATIONS with default values.
- **JButton:** Deletes the selected record from both the INFO table and SALARY_ALLOCATIONS table.
- **JButton:** Updates the bracket's values in the table based on the textbox values. Contains error checking.

Manipulate Salary History:

Manipulate Salary History Page

- **JTextBox:** takes search parameter input
- **JComboBox:** List of columns to search in
- **JComboBox:** List of columns to sort by
- **JLabel**
- **JComboBox:** Sort by ASC or DESC
- **JButton:** Confirm Sort Parameters
- **JButton:** takes user to Salary Generator page
- **JButton:** Resets Table order
- **JButton:** Prints the contents of the JTable to a PDF
- **JLabel**
- **JTextBox:** readonly, automatically generated
- **JCalendar:**
- **JTextBox:** number only
- **JTextBox:** number only
- **JLabel:** All text in this panel is a JLabel.
- **JTextBox:** readonly, automatically generated
- **JScrollBar**
- **JTable**
- **JButton:** Deselects the item
- **JButton:** Clears all fields
- **JButton:** Adds a record to the table. Has required field validation.
- **JButton:** Deletes the selected record.
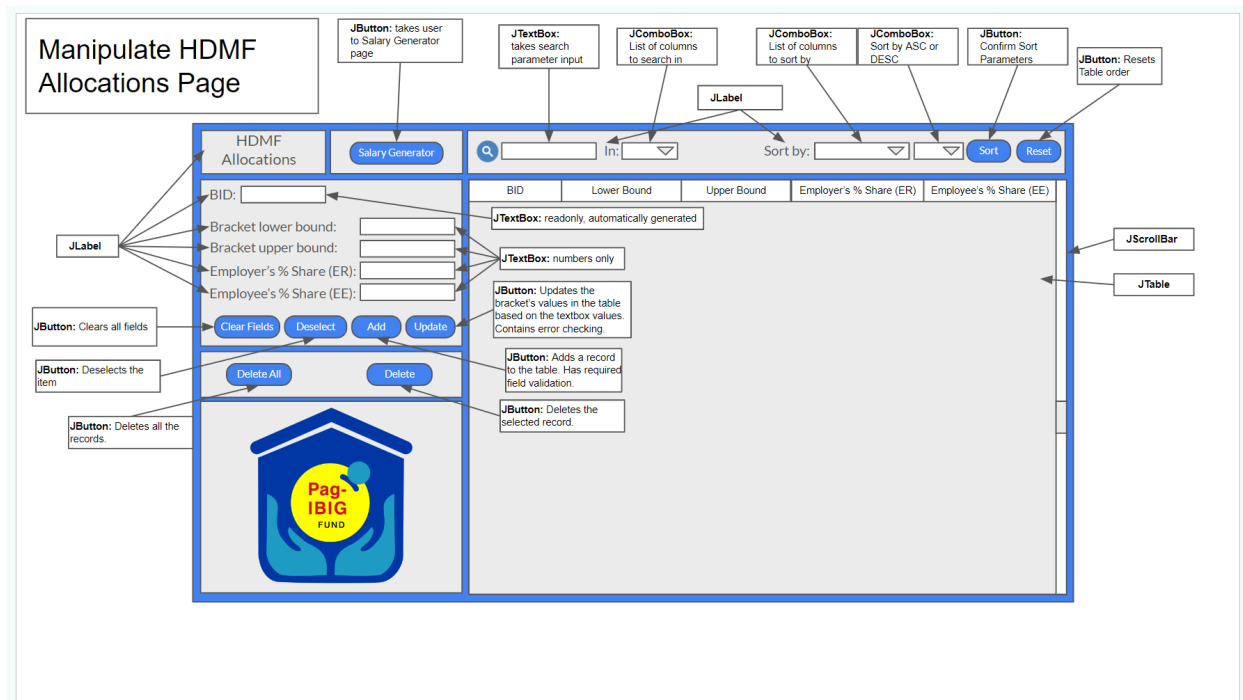
12

Manipulate Loan History:

Manipulate SSS Allocations:
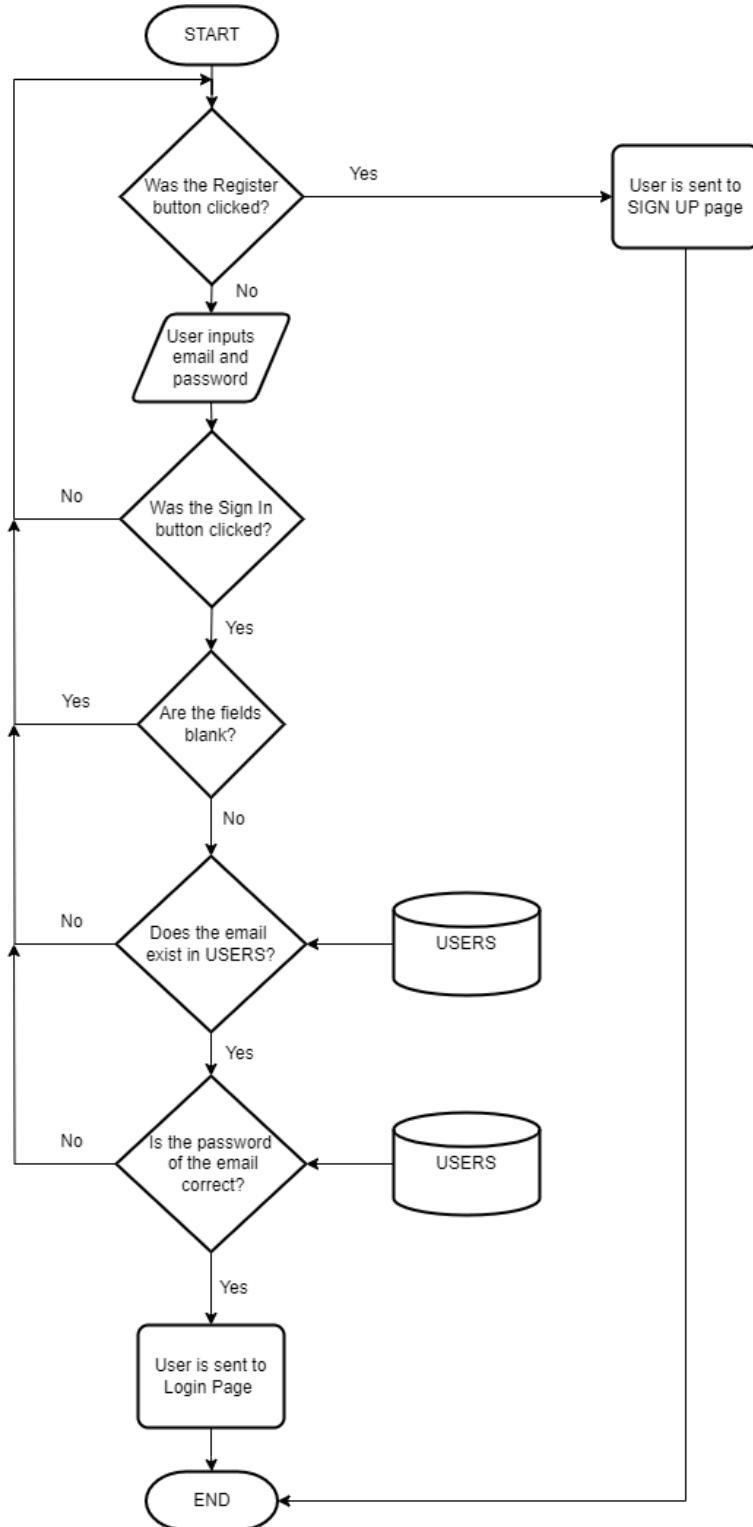
Manipulate PhilHealth Allocations:
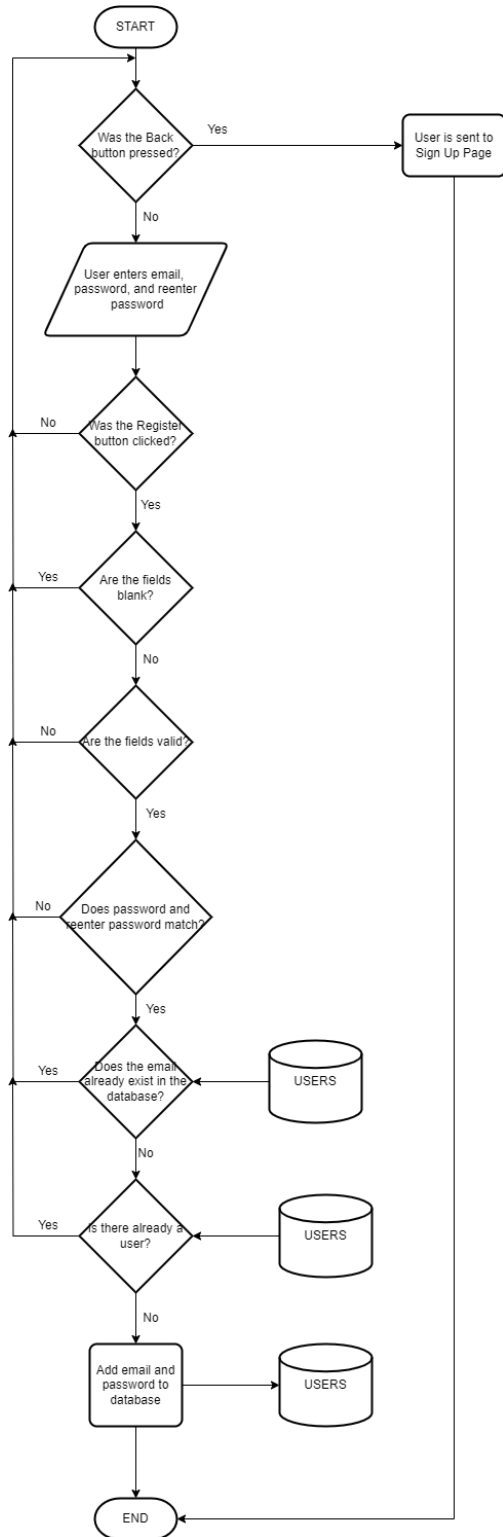

Manipulate HDMF Allocations:

(all icons are cited in Appendix B)
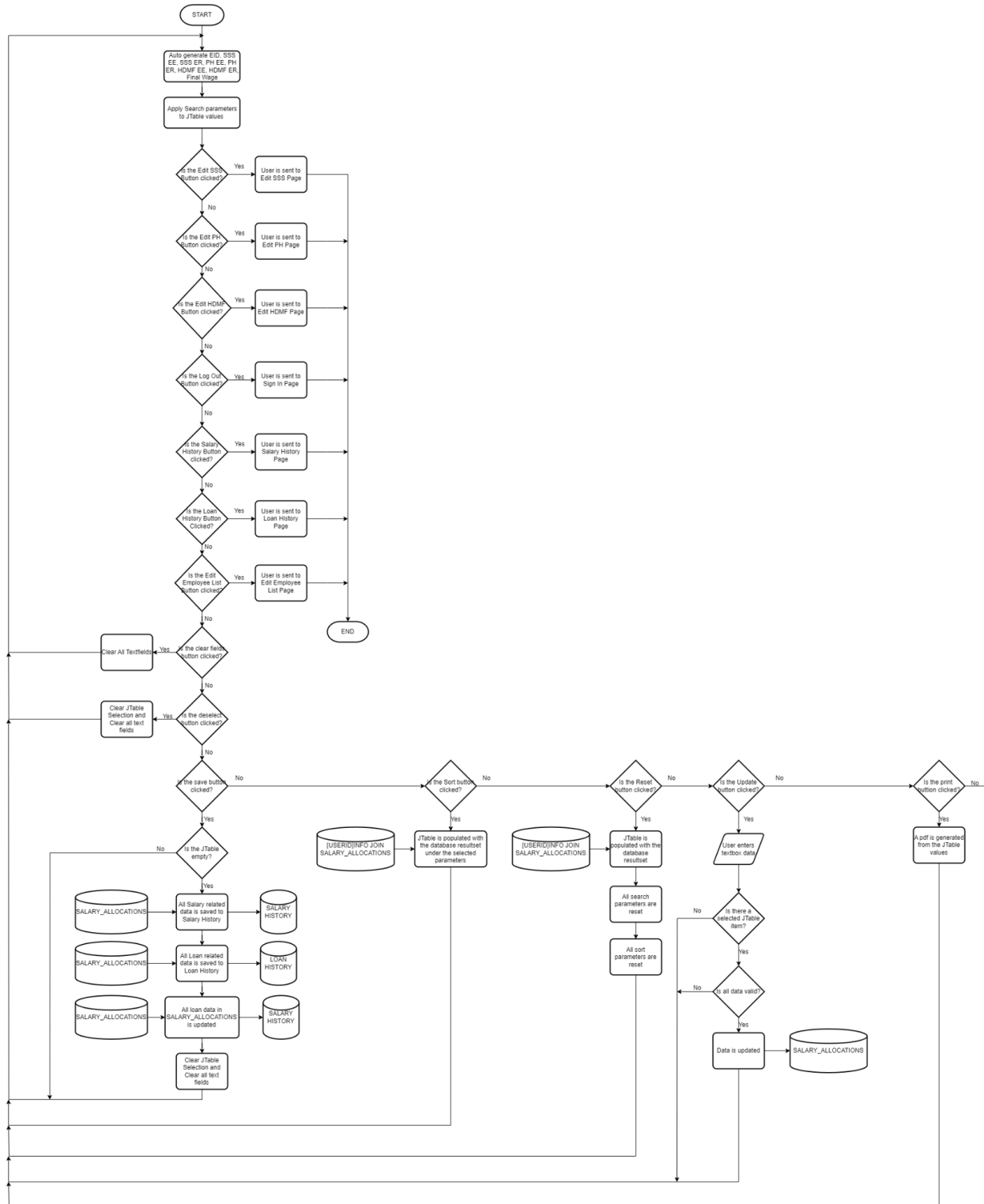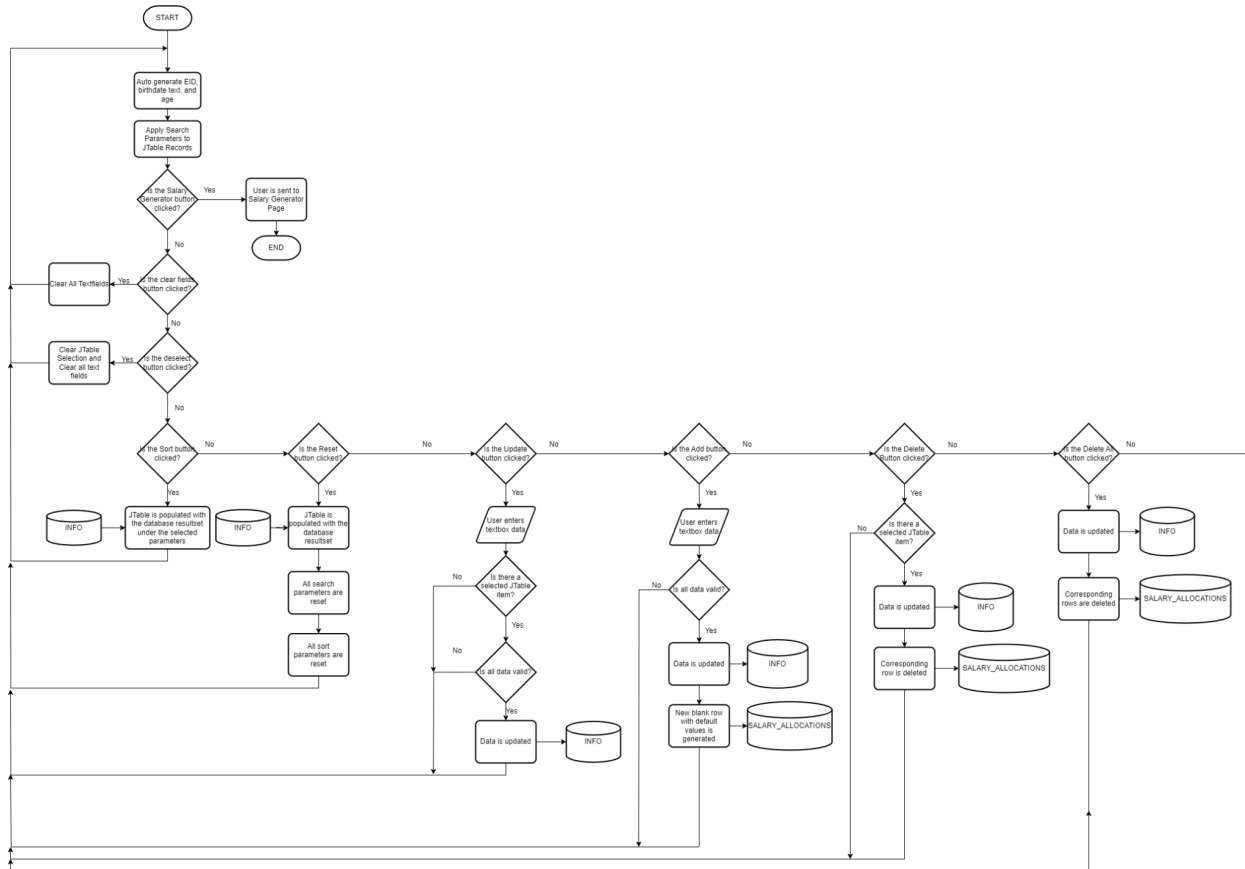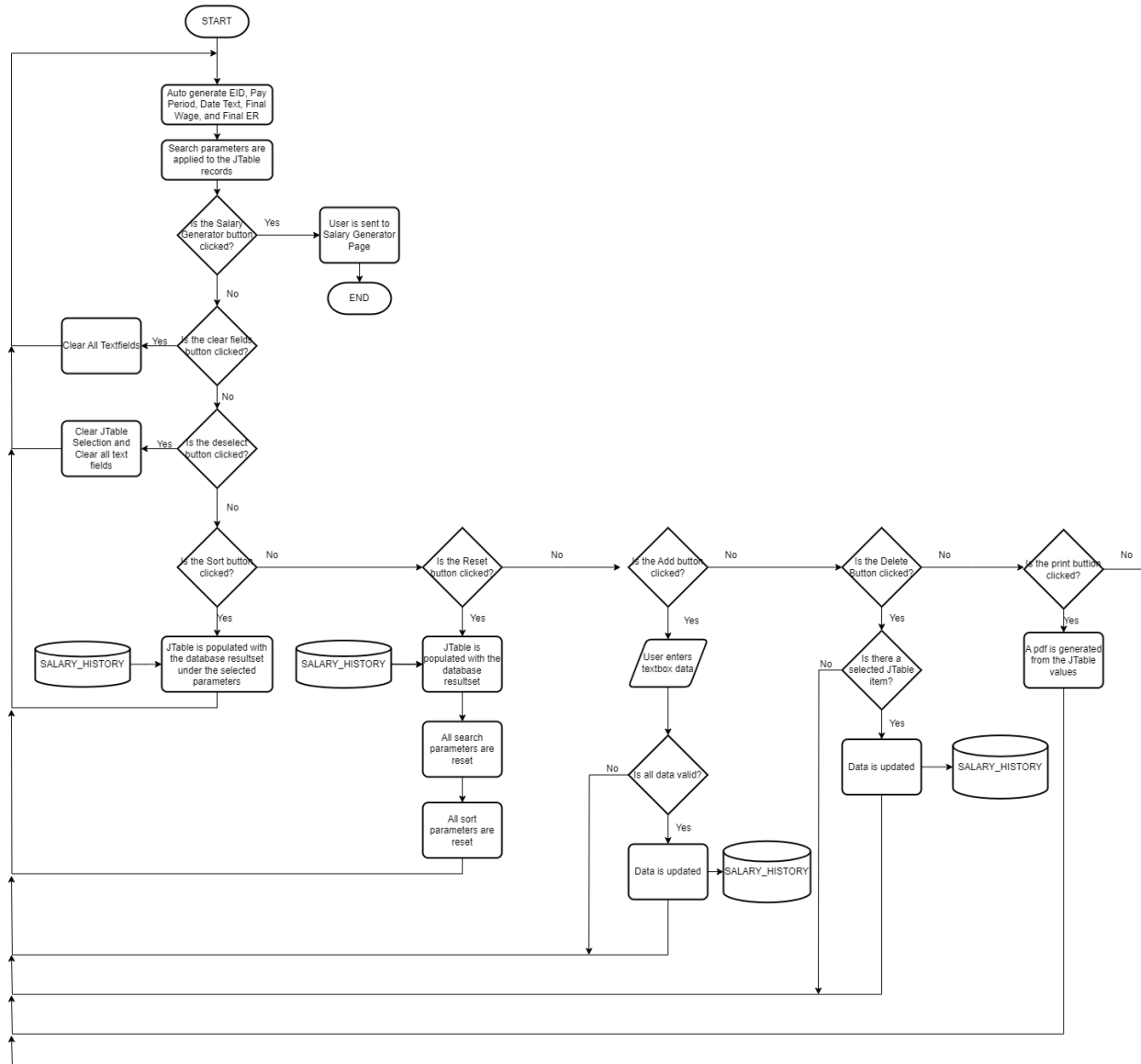
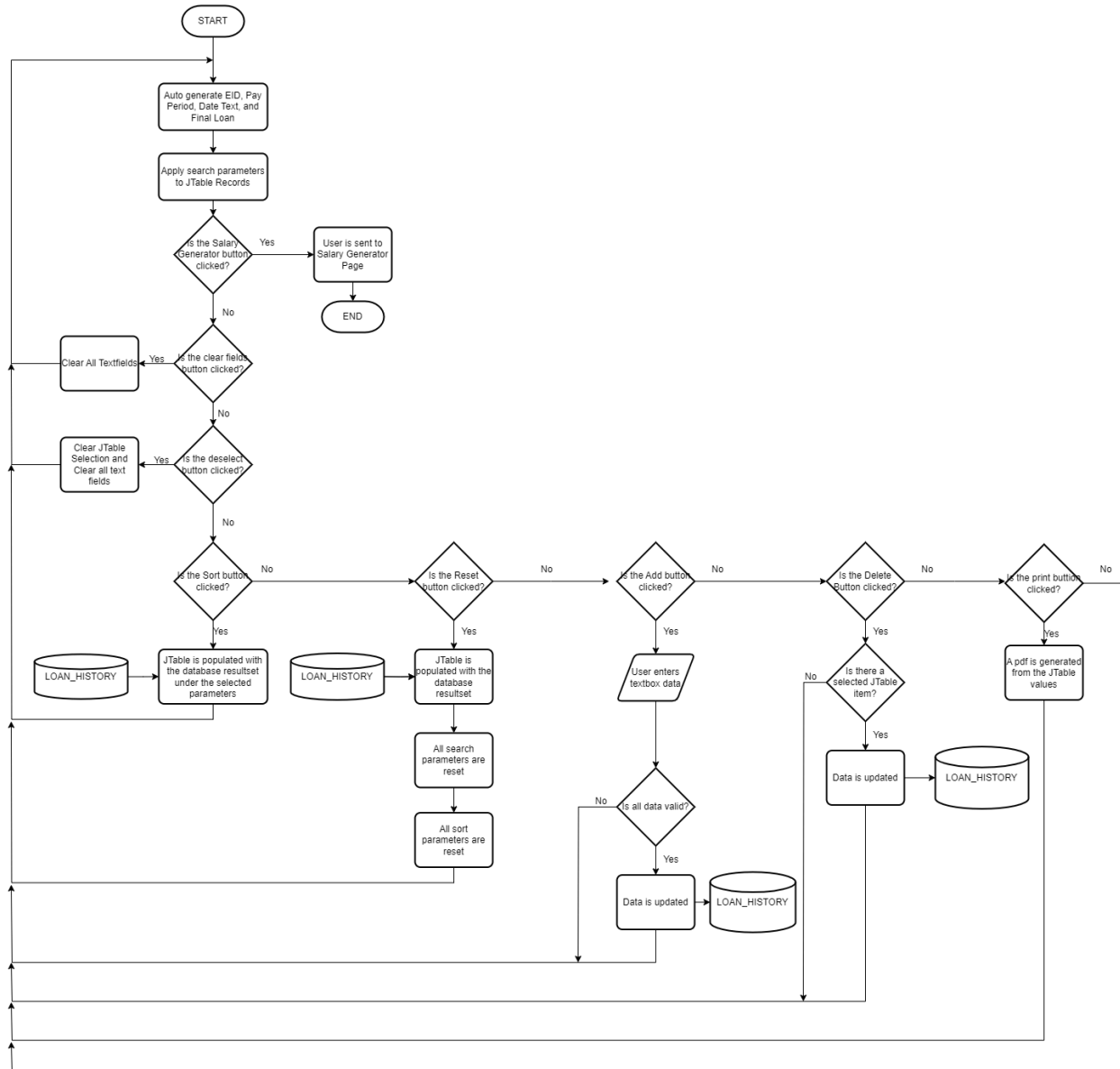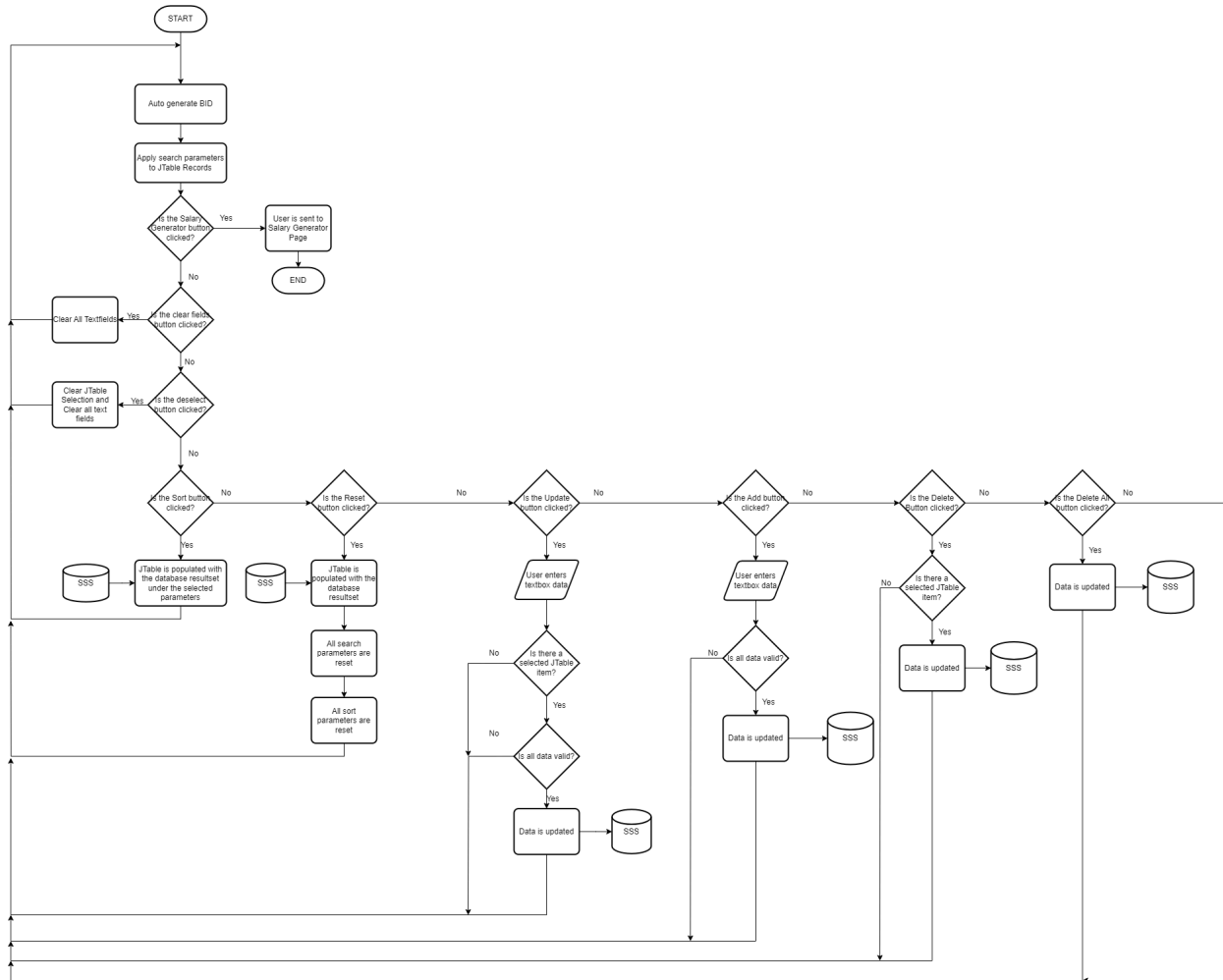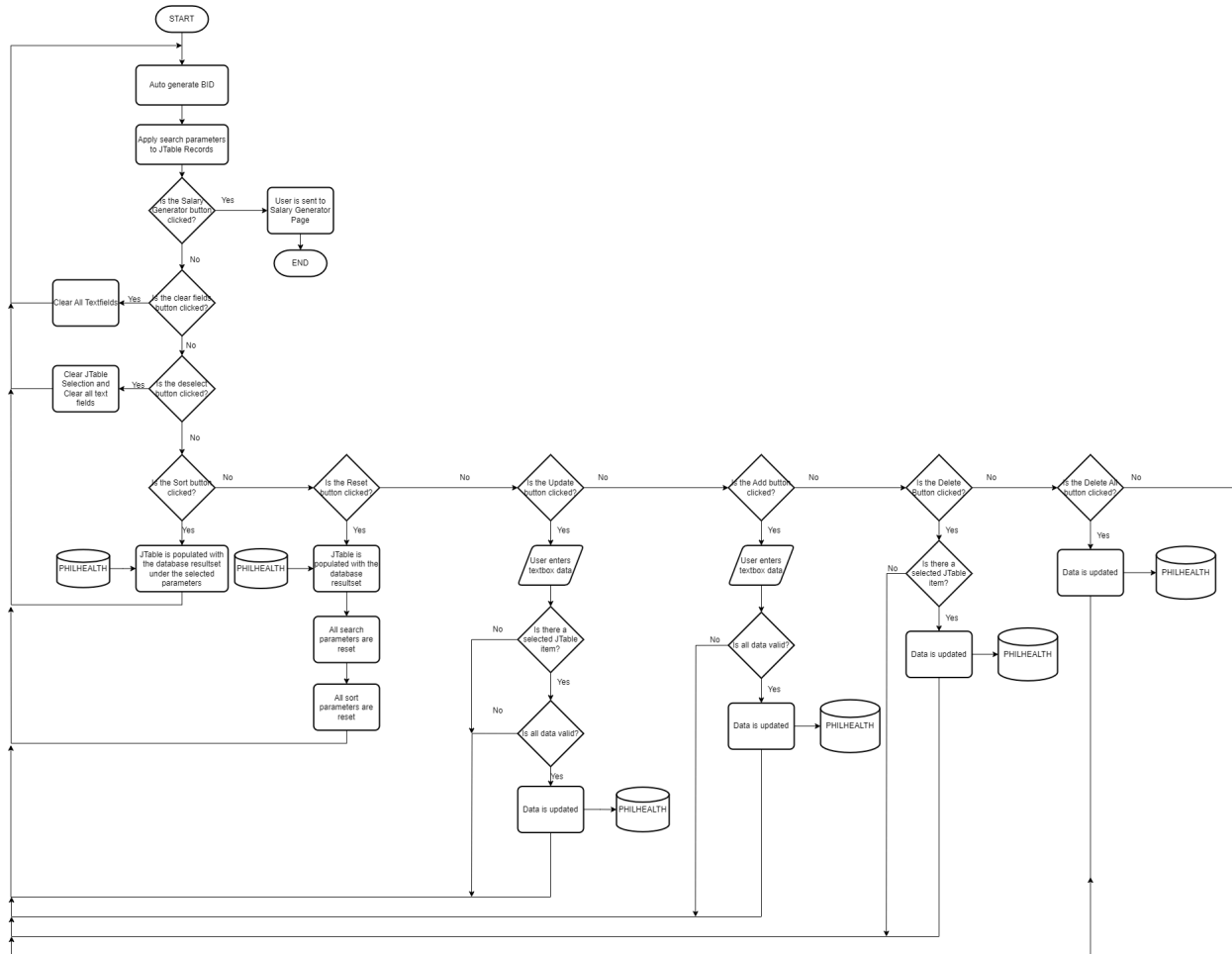# Process Flow Charts

Login Page:

Sign Up Page:

START

Was the Back button pressed? — Yes → User is sent to Sign Up Page

No

User enters email, password, and reenter password

Was the Register button clicked? — No

Yes

Are the fields blank? — Yes

No

Are the fields valid? — No

Yes

Does password and reenter password match? — No

Yes

Does the email already exist in the database? — Yes ← USERS

No

Is there already a user? — Yes ← USERS

No

Add email and password to database → USERS

END

Salary Generator:

```
                              ┌─────────┐
                              │  START  │
                              └────┬────┘
                                   │
                    ┌──────────────────────────────┐
                    │ Auto generate EID, SSS EE,    │
                    │ SSS ER, PH EE, PH ER, HDMF EE,│
                    │ HDMF ER, Final Wage           │
                    └──────────────┬────────────────┘
                                   │
                    ┌──────────────────────────────┐
                    │ Apply Search parameters       │
                    │ to JTable values              │
                    └──────────────┬────────────────┘
```

Is the Edit SSS Button clicked? — Yes → User is sent to Edit SSS Page
No ↓

Is the Edit PH Button clicked? — Yes → User is sent to Edit PH Page
No ↓

Is the Edit HDMF Button clicked? — Yes → User is sent to Edit HDMF Page
No ↓

Is the Log Out Button clicked? — Yes → User is sent to Sign In Page
No ↓

Is the Salary History Button clicked? — Yes → User is sent to Salary History Page
No ↓

Is the Loan History Button Clicked? — Yes → User is sent to Loan History Page
No ↓

Is the Edit Employee List Button clicked? — Yes → User is sent to Edit Employee List Page
No ↓

END

Clear All Textfields ← Yes — Is the clear fields button clicked?
No ↓

Clear JTable Selection and Clear all text fields ← Yes — Is the deselect button clicked?
No ↓

Is the save button clicked? — No → Is the Sort button clicked? — No → Is the Reset button clicked? — No → Is the Update button clicked? — No → Is the print button clicked? — No
Yes ↓

Is the JTable empty? — No

Sort (Yes): [USERID]INFO JOIN SALARY_ALLOCATIONS → JTable is populated with the database resultset under the selected parameters

Reset (Yes): [USERID]INFO JOIN SALARY_ALLOCATIONS → JTable is populated with the database resultset → All search parameters are reset → All sort parameters are reset

Update (Yes): User enters textbox data → Is there a selected JTable item? — No / Yes → Is all data valid? — No / Yes → Data is updated → SALARY_ALLOCATIONS

Print (Yes): A pdf is generated from the JTable values

Save (Yes):
SALARY_ALLOCATIONS → All Salary related data is saved to Salary History → SALARY HISTORY
SALARY_ALLOCATIONS → All Loan related data is saved to Loan History → LOAN HISTORY
SALARY_ALLOCATIONS → All loan data in SALARY_ALLOCATIONS is updated → SALARY HISTORY
→ Clear JTable Selection and Clear all text fields

Manipulate Employee Records:

Salary History:

START

Auto generate EID, Pay Period, Date Text, Final Wage, and Final ER

Search parameters are applied to the JTable records

Is the Salary Generator button clicked? — Yes → User is sent to Salary Generator Page → END

No

Is the clear fields button clicked? — Yes → Clear All Textfields

No

Is the deselect button clicked? — Yes → Clear JTable Selection and Clear all text fields

No

Is the Sort button clicked? — No → Is the Reset button clicked? — No → Is the Add button clicked? — No → Is the Delete Button clicked? — No → Is the print button clicked? — No

Yes

SALARY_HISTORY → JTable is populated with the database resultset under the selected parameters

SALARY_HISTORY → JTable is populated with the database resultset

Yes → User enters textbox data

Is there a selected JTable item? — No

Yes → Data is updated → SALARY_HISTORY

Yes → A pdf is generated from the JTable values

All search parameters are reset

All sort parameters are reset

Is all data valid? — No

Yes → Data is updated → SALARY_HISTORY

Loan History:

Manipulate SSS Allocations:

Manipulate PhilHealth Allocations:

Manipulate HDMF Allocations:

Key algorithms:

Highest Number Algorithm in an Array:

- Used to generate unique IDs for individual table records.
- To be used in Salary Generator, Edit Employee Records, Salary History, Loan History, Edit SSS, Edit PhilHealth, Edit HDMF.

Pseudocode:

```
A = [8, 3, 2, -3, 7]
SIZE = 5
HIGHEST = 0 //storing index
for I from 1 to SIZE-1
        if A[I] > A[HIGHEST] then
                HIGHEST = I
        end if
end loop
output "The highest number in the array is: " A[HIGHEST]
```

Flowchart:

START

A = [8, 3, 2, -3, 7]
SIZE = 5
HIGHEST = 0
I = 1

Is I<=SIZE-1?

No

Yes

Is A[I] >
A[HIGHEST]?

No

I = I + 1

Yes

HIGHEST = I

output "The highest
number in the array
is: " A[HIGHEST]

END
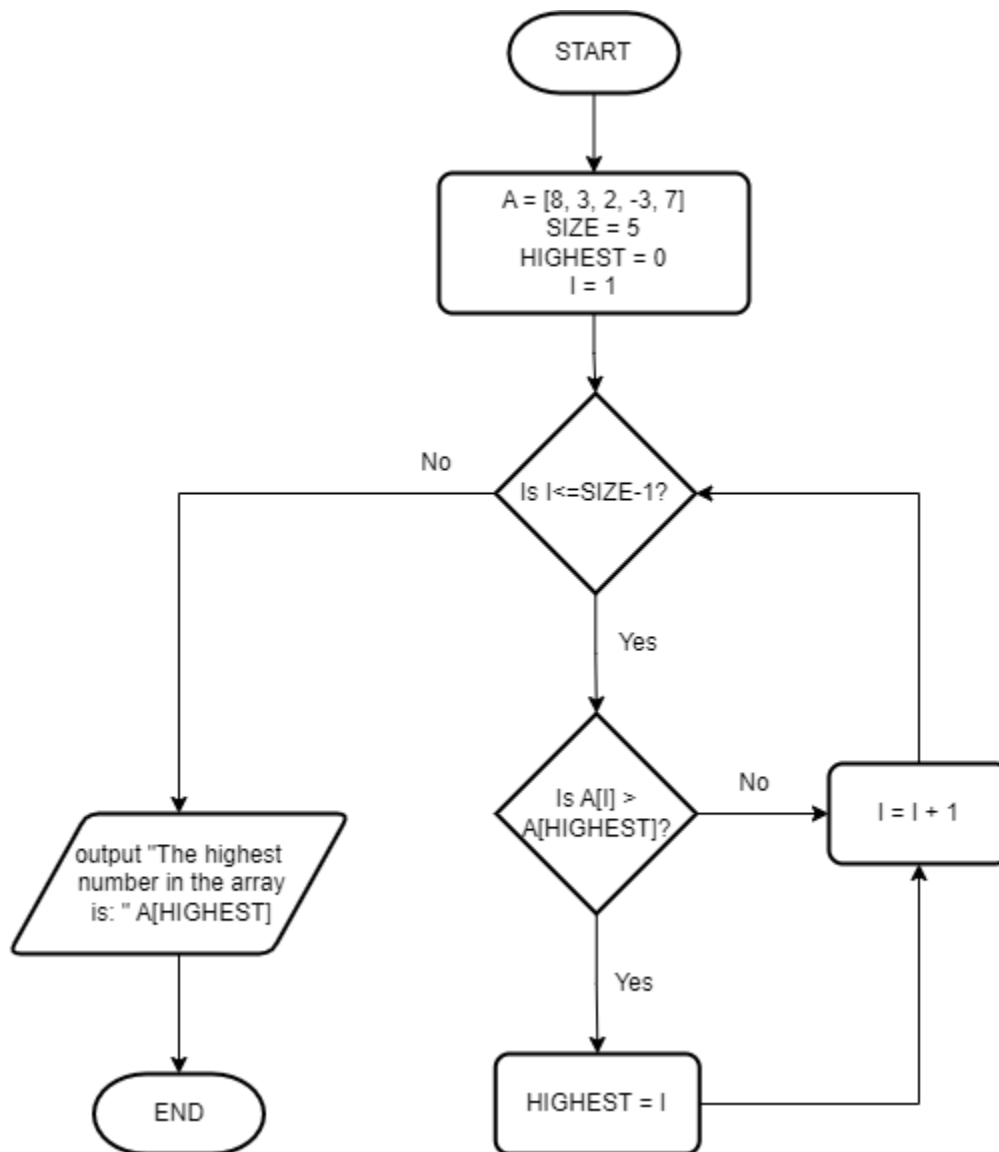
Linear Searching:

- Used to filter table records by those with a certain substring in them.
- To be used in Salary Generator, Edit Employee Records, Salary History, Loan History, Edit SSS, Edit PhilHealth, Edit HDMF.

Pseudocode:

```
TARGET = input()
A = ["Sample string", "Test string", "Hello World", "Johnny"]
SIZE = 4
ArrayList<Integer> FILTERED = new ArrayList<>()
I = 0
TARGETLENGTH = TARGET.length()
for I from 0 to SIZE - 1
        CHECK = A[I]
        CHECKLENGTH = CHECK.length()
        if TARGETLENGTH >= CHECKLENGTH then
                for J from 0 to CHECKLENGTH - TARGETLENGTH
                        if CHECK.substring(J, J + TARGETLENGTH) = TARGET then
                                FILTERED.add(CHECK)
                        end if
                end loop
        end if
end loop
```

Flowchart:

```
                    ┌─────────┐
                    │  START  │
                    └────┬────┘
                         │
                  ┌──────▼──────┐
                 /  TARGET =     /
                /   input()     /
               └───────┬───────┘
                       │
        ┌──────────────▼──────────────────────┐
        │ A = ["Sample string", "Test string", │
        │        "Hello World", "Johnny"]      │
        │ SIZE = 4                             │
        │ ArrayList<Integer> FILTERED =        │
        │            new ArrayList<>()         │
        │ I = 0                                │
        │ TARGETLENGTH = TARGET.length()       │
        └──────────────┬──────────────────────┘
                       │
                       ▼           No
                 ◇ Is I <= SIZE-1? ─────────────►  ┌─────┐
                       │                            │ END │
                       │ Yes                        └─────┘
                       ▼
               ┌──────────────────┐
               │ J = 0            │
               │ CHECK = A[I]     │
               │ CHECKLENGTH =    │
               │   CHECK.length() │
               └────────┬─────────┘
                        ▼
          No   ◇ Is TARGETLENGTH >=
       ◄───────   CHECKLENGTH?  ◇
                        │ Yes
                        ▼
          No   ◇ Is J <= CHECKLENGTH -
       ◄───────   TARGETLENGTH? ◇ ◄──────┐
                        │ Yes            │
                        ▼                │
       ◇ Is CHECK.substring(J, J +   No  │  ┌──────────┐
          TARGETLENGTH) = TARGET? ──────►  │ J = J + 1│
                        │ Yes                └────▲─────┘
                        ▼                         │
              ┌──────────────────┐                │
              │ FILTERED.add(     ──────────────────┘
              │      CHECK)      │
              └──────────────────┘

   ┌──────────┐
   │ I = I + 1│
   └──────────┘
```

27
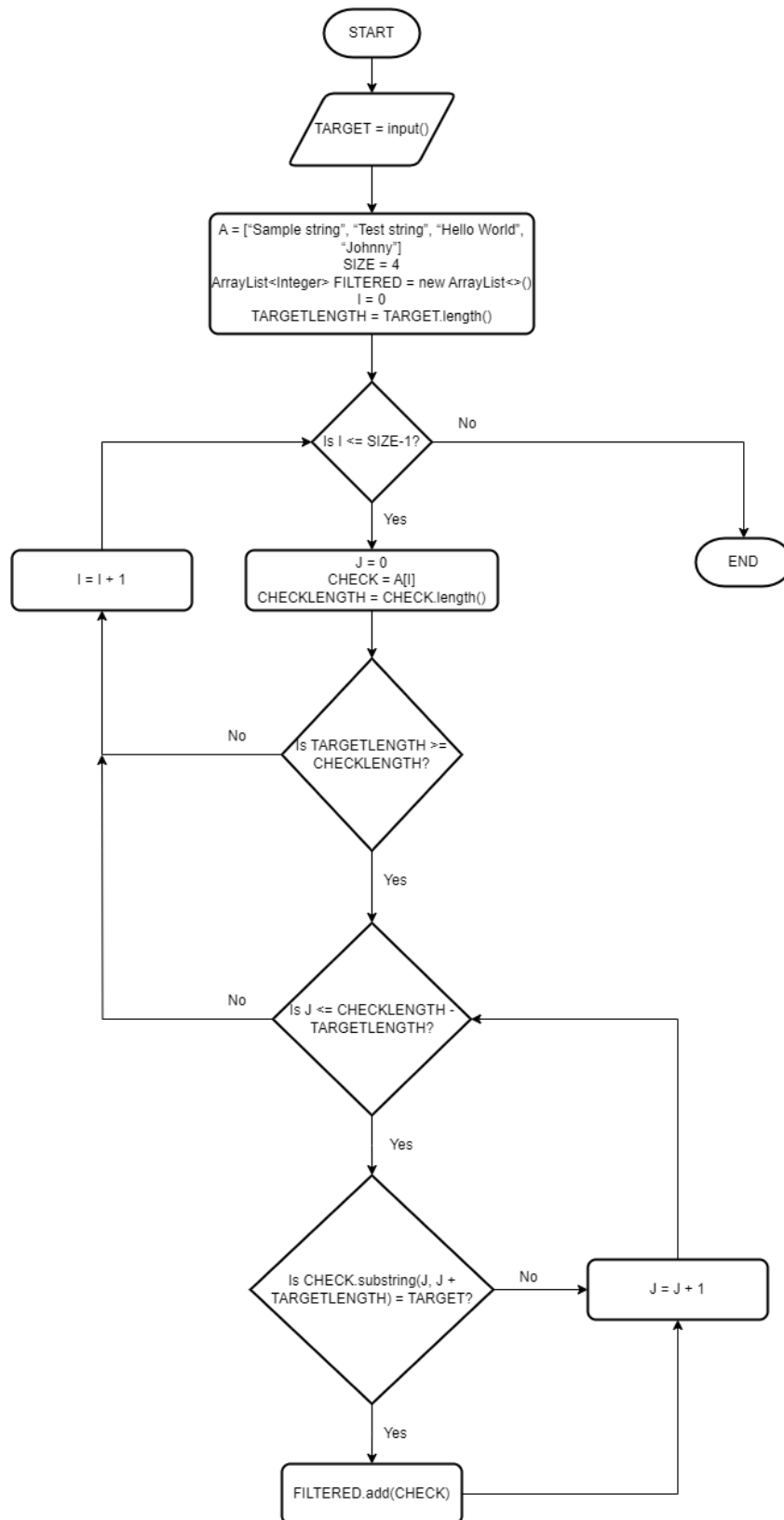
Bubble Sorting:

- Used to sort jtable records by a certain column ascending or descending.
- To be used in Salary Generator, Edit Employee Records, Salary History, Loan History, Edit SSS, Edit PhilHealth, Edit HDMF.

Pseudocode:

```
//Bubble Sort; Condition can be flipped depending on Ascending or Descending
//It is in ascending order.
A = [2, 4, 1, -7, 0]
SIZE = 5
FLAG = true
while FLAG = true
        FLAG = false
        I = 0
        for I from 0 to SIZE - 2
                if A[I] > A[I + 1] then
                        FLAG = true
                        temp = A[I + 1]
                        A[I + 1] = A[I]
                        A[I] = temp
                end if
        end loop
end loop
```

Flowchart:

START

A = [2, 4, 1, -7, 0]
SIZE = 5
FLAG = true

Is FLAG = true? — No → END

Yes

FLAG = false
I = 0

Is I<=SIZE-2? — No

Yes

Is A[I] > A[I+1]? — No → I = I + 1

Yes

FLAG = true
temp = A[I + 1]
A[I + 1] = A[I]
A[I] = temp
I = I + 1
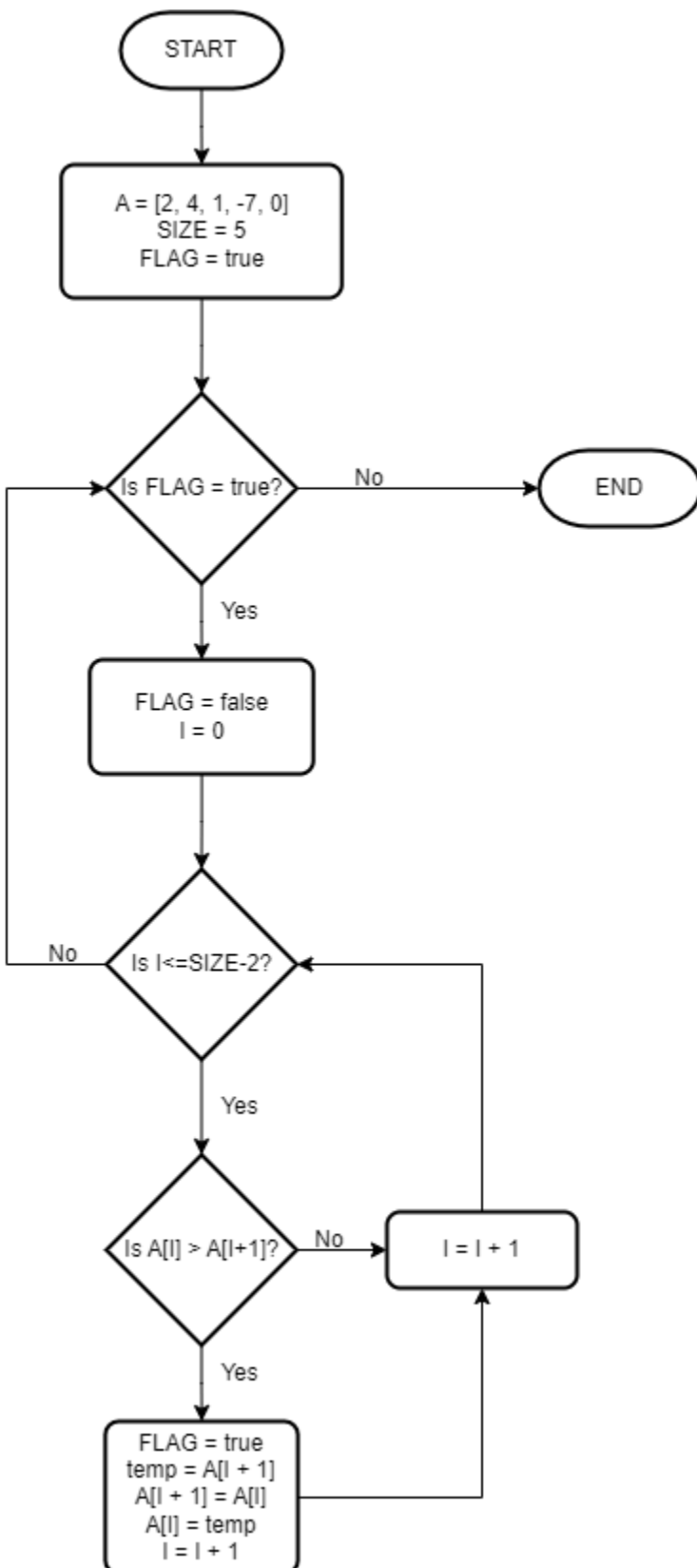
Email Validation:

- Used to validate emails.
- To be used in Login Page and Sign Up Page.

Pseudocode:

```
EMAIL = input.trim().toUpperCase()
LENGTH = EMAIL.length()
ATCOUNT = 0
ATPOS = 0
FIRSTDOTPOS = -1
DOTPOS = 0
FINALDOTPOS = 0
INVALIDCHARS = false
I = 0

for I from 0 to LENGTH - 1
        CHARACTER = EMAIL.charAt(I)
        if CHARACTER = '@'
                ATCOUNT = ATCOUNT + 1
                ATPOS = I
        end if
        if CHARACTER = '.' and ATPOS != 0 and DOTPOS = 0
                DOTPOS = I //the first '.' after the email
        end if
        if CHARACTER = '.' and ATPOS = 0 AND FIRSTDOTPOS = -1
                FIRSTDOTPOS = I //the first '.' before the email
        end if
        if CHARACTER = '.' and ATPOS != 0 and DOTPOS != 0
                FIRSTDOTPOS = I //the final '.'
        end if
        ASCII = (int)CHARACTER
        if ASCII<=45 or ASCII=47 or (ASCII>=58 and ASCII<=63) or ASCII>=91
                INVALIDCHARS = true
        end if
end loop

if LENGTH = 0
        JOptionPane.showMessageDialog(null, "Email Field Empty. Please enter an email.")
else if FIRSTDOTPOS = 0
        JOptionPane.showMessageDialog(null, "Invalid Email. The '.' cannot be at the first
position.")
else if FINALDOTPOS = LENGTH - 1
```

JOptionPane.showMessageDialog(null, "Invalid Email. The '.' cannot be at the first position.")
else if ATCOUNT = 0
JOptionPane.showMessageDialog(null, "Invalid Email. The email is missing an '@' symbol.")
else if ATCOUNT>=2
JOptionPane.showMessageDialog(null, "Invalid Email. The email has too many '@' symbols.")
else if ATPOS = 0
JOptionPane.showMessageDialog(null, "Invalid Email. The email is lacking a prefix before the '@' symbol.")
else if DOTPOS - ATPOS <=1
JOptionPane.showMessageDialog(null, "Invalid Email. The email is lacking a valid domain name.")
else if INVALIDCHARS = true
JOptionPane.showMessageDialog(null, "Invalid Email. The email contains invalid characters.")
else if LENGTH>100
JOptionPane.showMessageDialog(null, "Invalid Email. The email is longer than 100 characters.")
end if

Flowchart:

```
                          START
                            │
         ┌──────────────────────────────────────┐
         │ EMAIL = input.trim().toUpperCase()    │
         │ LENGTH = EMAIL.length()               │
         │ ATCOUNT = 0                           │
         │ ATPOS = 0                             │
         │ FIRSTDOTPOS = -1                      │
         │ DOTPOS = 0                            │
         │ FINALDOTPOS = 0                       │
         │ INVALIDCHARS = false                  │
         │ I = 0                                 │
         └──────────────────────────────────────┘
                            │
            ┌───────────────┴──────────────┐
            │ Is I<LENGTH? │──── (continue) ──► Is LENGTH = 0?
            └──────────────┘
                  │ YES
       CHARACTER = EMAIL.CharAt(I)
                  │
         Is CHARACTER =='@'? ──YES──► ATCOUNT = ATCOUNT + 1, ATPOS = I
                  │ NO
         Is CHARACTER ='.' and ATPOS!=0 and DOTPOS=0? ──YES──► DOTPOS = I
                  │ NO
         Is CHARACTER ='.' and ATPOS==0 and FIRSTDOTPOS=-1? ──YES──► FIRSTDOTPOS = I
                  │ NO
         Is CHARACTER ='.' and ATPOS!=0 and DOTPOS!=0? ──YES──► FINALDOTPOS = I
                  │ NO
         ASCII = (int)CHARACTER
                  │
         Is ASCII<=45 or ASCII=47 or (ASCII>=58 and ASCII<=63) or ASCII>=91? ──YES──► INVALIDCHARS = true
                  │ NO
              I = I + 1
```

Right column (after Is I<LENGTH? NO):

- Is LENGTH = 0? ──YES──► output "Email field empty. Please enter an email."
- │ NO
- Is FIRSTDOTPOS = 0? ──YES──► output "Invalid Email. The '.' cannot be at the first position."
- │ NO
- Is FINALDOTPOS = LENGTH - 1? ──YES──► output "Invalid Email. A '.' cannot be at the final position."
- │ NO
- Is ATCOUNT = 0? ──YES──► output "Invalid Email. The email is missing an '@' symbol."
- │ NO
- Is ATCOUNT>=2? ──YES──► output "Invalid Email. The email has too many '@' symbols."
- │ NO
- Is ATPOS = 0? ──YES──► output "Invalid Email. The email is lacking a prefix before the '@' symbol."
- │ NO
- Is DOTPOS - ATPOS<=1? ──YES──► output "Invalid Email. The email is lacking a valid domain name."
- │ NO
- Is INVALIDCHARS = true? ──YES──► output "Invalid Email. The email contains invalid characters."
- │ NO
- Is LENGTH>100? ──YES──► output "Invalid Email. The email is longer than 100 characters."
- │ NO
- output "Email is valid."
- │
- END

Test plan:

Form Name	Testing	Example of the Test	Success Criteria
Form 1 (Login)	● Register button works	● Clicking the Register button sends the user to the sign up page	1, 11
	● Password box hides password	● Text typed into the password box appears as asterisks	
	● JRadioButton toggles showing password	● Activating the JRadioButton shows the password	
	● Login button sends user to the main menu, but not if credentials are incorrect	● A valid username and password sends the user to the main menu. Invalid throws an error message. Checks the database.	
Form 2 (Sign Up)	● Back button works	● Clicking the back button sends the user to the login page	1, 11
	● Password boxes hide password	● Text typed into the password boxes appear as asterisks	
	● JRadioButton toggles showing password	● Activating the respective JRadioButton shows their respective passwords	
	● Register button sends user to login page	● If a user hasn't been made yet, then the system adds a new record to the USERS table. If one has been made already, an error message is shown.	
Form 3 (Salary Generator)	● Log Out button works	● Clicking the log out button sends the user to the login page	5, 6, 7, 8, 9, 10, 11
	● Edit SSS button works	● Clicking the edit SSS button sends the user to the Manipulate SSS Allocations Page.	
	● Edit PH button works	● Clicking the edit PH button sends the user to the	

		Manipulate PhilHealth Allocations Page.	
	● Edit HDMF button works	● Clicking the edit HDMF button sends the user to the Manipulate HDMF Allocations Page.	
	● Salary History button works	● Clicking the Salary History button sends the user to the Salary History page.	
	● Loan History button works	● Clicking the Loan History button sends the user to the Loan History page.	
	● Edit Employee Records button works	● Clicking the edit employee records button sends the user to the Manipulate Employee Records page.	
	● Clear Fields button works	● Clicking the Clear Fields button clears all text fields	
	● Deselect button works	● Clicking the Deselect button clears all text fields and deselect the JTable selection	
	● Save button works	● Clicking the save button saves all of SALARY_ALLOCATIONS's salary-related values in SALARY_HISTORY and all loan-related values to LOAN_HISTORY, throwing an error if the JTable is empty.	
	● Sort button works	● Clicking the sort button sorts the table by the designated parameters.	
	● Reset button works	● Clicking the reset button sorts the table with no parameters.	
	● Update button works	● Clicking the update button updates the values of the record in the SALARY_ALLOCATIONS table if all JTextField values are valid and a record is selected. Otherwise an error message is	

		shown.	
	● Print button works	● The print button exports all JTable information in pdf form.	
	● All readonly values appear.	● All readonly values such as primary keys and foreign keys are automatically filled and calculated by the system.	
	● Social Security JComboBoxes are auto filled with values.	● JComboBoxes for Social Security are automatically filled with choices pulled from the SSS, PhilHealth, and HDMF tables.	
	● Startup of the window executes normally	● When opened, a reminder is given to the user to do payroll if they haven't done it for that half-month yet.	
Form 4 (Manipulate Employee Records)	● Salary Generator button works	● Clicking the salary generator button sends the user to the Salary Generator page.	2, 3, 4, 11
	● Clear Fields button works	● Clicking the Clear Fields button clears all text fields	
	● Deselect button works	● Clicking the Deselect button deselects the JTable selection	
	● Sort button works	● Clicking the sort button sorts the table by the designated parameters.	
	● Reset button works	● Clicking the reset button sorts the table with no parameters.	
	● Update button works	● Clicking the update button updates the values of the record in the INFO table if all JTextField values are valid and a record is selected. Otherwise an error message is shown.	
	● Add button works	● Clicking the add button adds a new record to INFO if all required JTextField values are valid and nonempty. Otherwise an error message is shown. It also adds a new empty record	

		to SALARY_ALLOCATIONS.	
	● Delete button works	● Clicking the delete button deletes the record from INFO if a record has been selected from the JTable. Otherwise an error message is shown. It then deletes the corresponding record in the SALARY_ALLOCATIONS table.	
	● Delete All button works	● Clicking the delete all button deletes all the records from the INFO table and all corresponding records in the SALARY_ALLOCATIONS table.	
	● All readonly values appear.	● All readonly values such as primary keys and foreign keys are automatically filled and calculated by the system.	
Form 5 (Manipulate Salary History)	● Salary Generator button works	● Clicking the salary generator button sends the user to the Salary Generator page.	9, 11
	● Clear Fields button works	● Clicking the Clear Fields button clears all text fields	
	● Deselect button works	● Clicking the Deselect button deselects the JTable selection	
	● Sort button works	● Clicking the sort button sorts the table by the designated parameters.	
	● Reset button works	● Clicking the reset button sorts the table with no parameters.	
	● Add button works	● Clicking the add button adds a new record to the SALARY_HISTORY table if all the JTextField values are valid and nonempty. Otherwise an error message is shown.	
	● Delete button works	● Clicking the Delete button removes the record from the SALARY_HISTORY table if a	

		record has been selected from the JTable. Otherwise an error message is shown.	
	● Print button works	● The print button exports all JTable information in pdf form.	
	● All readonly values appear.	● All readonly values such as primary keys and foreign keys are automatically filled and calculated by the system.	
Form 6 (Manipulate Loan History)	● Salary Generator button works	● Clicking the salary generator button sends the user to the Salary Generator page.	9, 11
	● Clear Fields button works	● Clicking the Clear Fields button clears all text fields	
	● Deselect button works	● Clicking the Deselect button deselects the JTable selection	
	● Sort button works	● Clicking the sort button sorts the table by the designated parameters.	
	● Reset button works	● Clicking the reset button sorts the table with no parameters.	
	● Add button works	● Clicking the add button adds a new record to the LOAN_HISTORY table if all the JTextField values are valid and nonempty. Otherwise an error message is shown.	
	● Delete button works	● Clicking the Delete button removes the record from the LOAN_HISTORY table if a record has been selected from the JTable. Otherwise an error message is shown.	
	● Print button works	● The print button exports all JTable information in pdf form.	
	● All readonly values appear.	● All readonly values such as primary keys and foreign keys are automatically filled and calculated by the system.	

Form 7 (Manipulate SSS Allocations)	● Salary Generator button works	● Clicking the salary generator button sends the user to the Salary Generator page.	11
	● Clear Fields button works	● Clicking the Clear Fields button clears all text fields	
	● Deselect button works	● Clicking the Deselect button deselects the JTable selection	
	● Sort button works	● Clicking the sort button sorts the table by the designated parameters.	
	● Reset button works	● Clicking the reset button sorts the table with no parameters.	
	● Update button works	● Clicking the update button updates the values of the record in the SSS table if all JTextField values are valid and a record is selected. Otherwise an error message is shown.	
	● Add button works	● Clicking the add button adds a new record to the SSS table if all the JTextField values are valid and nonempty. Otherwise an error message is shown.	
	● Delete button works	● Clicking the Delete button removes the record from the SSS table if a record has been selected from the JTable. Otherwise an error message is shown.	
	● Delete all button works	● Clicking the delete all button deletes all the records from the SSS table.	
	● All readonly values appear.	● All readonly values such as primary keys and foreign keys are automatically filled and calculated by the system.	
Form 8 (Manipulate PhilHealth Allocations)	● Salary Generator button works	● Clicking the salary generator button sends the user to the Salary Generator page.	11

	● Clear Fields button works	● Clicking the Clear Fields button clears all text fields	
	● Deselect button works	● Clicking the Deselect button deselects the JTable selection	
	● Sort button works	● Clicking the sort button sorts the table by the designated parameters.	
	● Reset button works	● Clicking the reset button sorts the table with no parameters.	
	● Update button works	● Clicking the update button updates the values of the record in the PHILHEALTH table if all JTextField values are valid and a record is selected. Otherwise an error message is shown.	
	● Add button works	● Clicking the add button adds a new record to the PHILHEALTH table if all the JTextField values are valid and nonempty. Otherwise an error message is shown.	
	● Delete button works	● Clicking the Delete button removes the record from the PHILHEALTH table if a record has been selected from the JTable. Otherwise an error message is shown.	
	● Delete all button works	● Clicking the delete all button deletes all the records from the PHILHEALTH table.	
	● All readonly values appear.	● All readonly values such as primary keys and foreign keys are automatically filled and calculated by the system.	
Form 9 (Manipulate HDMF Allocations)	● Salary Generator button works	● Clicking the salary generator button sends the user to the Salary Generator page.	11
	● Clear Fields button works	● Clicking the Clear Fields button clears all text fields	

	● Deselect button works	● Clicking the Deselect button deselects the JTable selection	
	● Sort button works	● Clicking the sort button sorts the table by the designated parameters.	
	● Reset button works	● Clicking the reset button sorts the table with no parameters.	
	● Update button works	● Clicking the update button updates the values of the record in the HDMF table if all JTextField values are valid and a record is selected. Otherwise an error message is shown.	
	● Add button works	● Clicking the add button adds a new record to the HDMF table if all the JTextField values are valid and nonempty. Otherwise an error message is shown.	
	● Delete button works	● Clicking the Delete button removes the record from the HDMF table if a record has been selected from the JTable. Otherwise an error message is shown.	
	● Delete all button works	● Clicking the delete all button deletes all the records from the HDMF table.	
	● All readonly values appear.	● All readonly values such as primary keys and foreign keys are automatically filled and calculated by the system.	