



TRƯỜNG ĐẠI HỌC FPT

<<LIBRARY MANAGEMENT SYSTEM>>

Software Design Document

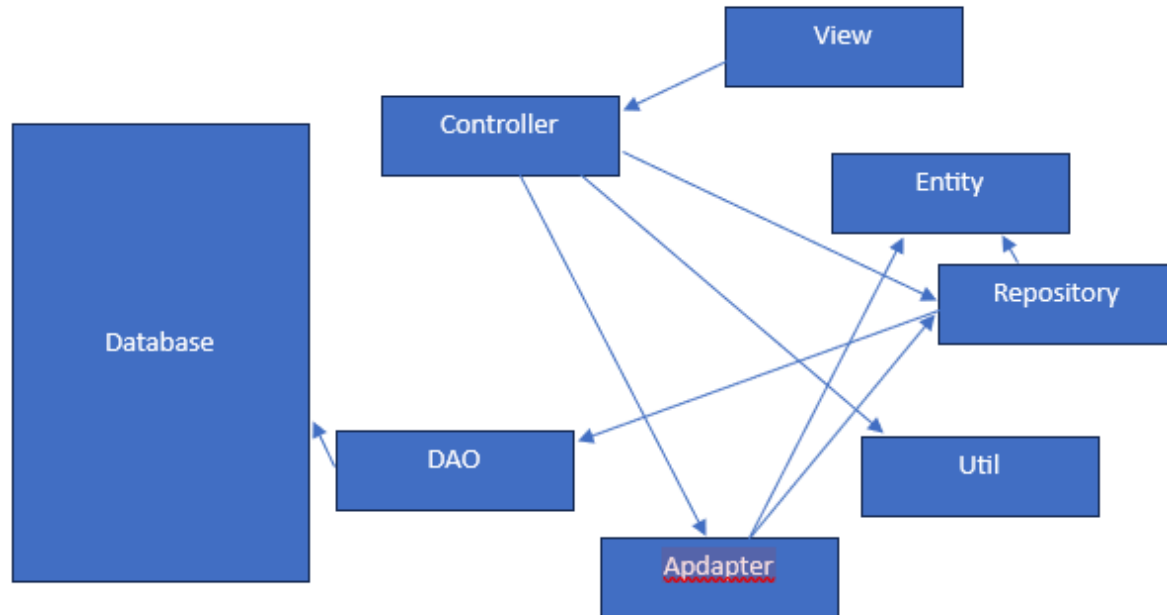
– Hanoi, March 2024–

Table of Contents

I. Overview.....	3
1. Code Packages.....	3
2. Database Schema.....	3
II. Code Designs.....	4
1. <Feature/Function Name>.....	4
a. Class Diagram.....	4
b. Class Specifications.....	4
c. Sequence Diagram(s).....	4
d. Database queries.....	5
2. <Feature/Function Name2>.....	5
III. Database Tables.....	5
1. <Table name >.....	5
2. <Table name>.....	5

I. Overview

1. Code Packages/Namespaces



Package descriptions & package class naming conventions

No	Package	Description
01	entries	Contains classes related to entries, possibly referring to entries in a system or application.
02	View	Contains classes related to the presentation layer, such as activities, fragments, and custom views.
03	Controller	Contains classes that control the flow of data and logic within the application, often acting as intermediaries between the view and the model.
04	Repository	Contains classes responsible for managing data operations, including fetching, storing, and updating data from various sources.
05	Util	Contains utility classes and methods that provide common functionality used throughout the application.

06	Adapter	Contains classes related to adapting data for display, such as RecyclerView adapters.
07	DAO	Contains classes related to data access objects, often used in conjunction with Room or other database libraries.
08	Database	Contains classes related to database management, such as database creation, migration, and configuration.

2. Database Design

2.1 Database Schema

[Provide the tables relationship like example below – following MySQL database naming convention]

Table descriptions & package class naming conventions are as below

No	Table	Description
01	MasterData	Stores master data that is used throughout the application, such as configuration settings or reference data.
02	Logs	Stores logs of system activities or events for monitoring and troubleshooting purposes.
03	Employee	Stores information about employees, such as their name, contact details, and employment history.
04	Order	Stores information about orders placed by customers, including order details and status.
05	Customer	Stores information about customers, including their name, contact details, and billing information.
06	OrderDetail	Stores detailed information about items included in each order, such as quantity, price, and product details.
07	Book	Stores information about books, including title, author, ISBN, and other relevant details.

2.2 Database table

MasterData	
uu_id	
type	
value	
key_col	
description	
created_at	
created_by	
updated_at	
updated_by	
deleted_at	
deleted_by	

Logs	
uu_id	
action_name	
status	
link	
created_at	
created_by	
updated_at	
updated_by	
deleted_at	
deleted_by	

Employee	
uu_id	
name	
dob	
id	
address	
phone	
email	
role	
created_at	
created_by	
updated_at	
updated_by	
deleted_at	
deleted_by	

Order	
uu_id	
customer	
description	
borrowed_day	
due_date	
return_date	
status	
created_at	
created_by	
updated_at	
updated_by	
deleted_at	
deleted_by	

Customer	
uu_id	
name	
dob	
id	
address	
phone	
email	
status	
created_at	
created_by	
updated_at	
updated_by	
deleted_at	
deleted_by	

OrderDetail	
uu_id	
order_uu_id	
book_uu_id	
description	
created_at	
created_by	
updated_at	
updated_by	
deleted_at	
deleted_by	

Book	
uu_id	
name	
book_id	
supplier	
author	
status	
category	
description	
order_id	
created_at	
created_by	
updated_at	
updated_by	
deleted_at	
deleted_by	

1. <Book>

Book			
	Column Name	Data Type	Allow Nulls
🔑	uu_id	uniqueidentifier	<input type="checkbox"/>
	name	varchar(255)	<input type="checkbox"/>
	book_id	int	<input checked="" type="checkbox"/>
	supplier	varchar(255)	<input checked="" type="checkbox"/>
	author	varchar(255)	<input checked="" type="checkbox"/>
	status	varchar(50)	<input checked="" type="checkbox"/>
	category	varchar(50)	<input checked="" type="checkbox"/>
	description	nvarchar(255)	<input checked="" type="checkbox"/>
	order_id	uniqueidentifier	<input checked="" type="checkbox"/>
	created_at	datetime2(7)	<input checked="" type="checkbox"/>
	created_by	varchar(255)	<input checked="" type="checkbox"/>
	updated_at	datetime2(7)	<input checked="" type="checkbox"/>
	updated_by	varchar(255)	<input checked="" type="checkbox"/>
	deleted_at	datetime2(7)	<input checked="" type="checkbox"/>
	deleted_by	varchar(255)	<input checked="" type="checkbox"/>
			<input type="checkbox"/>


2. <Order Detail>

OrderDetail			
	Column Name	Data Type	Allow Nulls
🔍	uu_id	uniqueidentifier	<input type="checkbox"/>
	order_uu_id	uniqueidentifier	<input checked="" type="checkbox"/>
	book_uu_id	uniqueidentifier	<input checked="" type="checkbox"/>
	description	nvarchar(255)	<input checked="" type="checkbox"/>
	created_at	datetime2(7)	<input checked="" type="checkbox"/>
	created_by	varchar(255)	<input checked="" type="checkbox"/>
	updated_at	datetime2(7)	<input checked="" type="checkbox"/>
	updated_by	varchar(255)	<input checked="" type="checkbox"/>
	deleted_at	datetime2(7)	<input checked="" type="checkbox"/>
	deleted_by	varchar(255)	<input checked="" type="checkbox"/>
			<input type="checkbox"/>


2. <Customer>

Customer			
	Column Name	Data Type	Allow Nulls
🔍	uu_id	uniqueidentifier	<input type="checkbox"/>
	name	varchar(255)	<input type="checkbox"/>
	dob	date	<input checked="" type="checkbox"/>
	id	int	<input checked="" type="checkbox"/>
	address	varchar(255)	<input checked="" type="checkbox"/>
	phone	varchar(20)	<input checked="" type="checkbox"/>
	email	varchar(255)	<input checked="" type="checkbox"/>
	status	varchar(20)	<input checked="" type="checkbox"/>
	created_at	datetime2(7)	<input checked="" type="checkbox"/>
	created_by	varchar(255)	<input checked="" type="checkbox"/>
	updated_at	datetime2(7)	<input checked="" type="checkbox"/>
	updated_by	varchar(255)	<input checked="" type="checkbox"/>
	deleted_at	datetime2(7)	<input checked="" type="checkbox"/>
	deleted_by	varchar(255)	<input checked="" type="checkbox"/>
			<input type="checkbox"/>

2. <Order>

Order			
	Column Name	Data Type	Allow Nulls
	uu_id	uniqueidentifier	<input type="checkbox"/>
	customer	uniqueidentifier	<input checked="" type="checkbox"/>
	description	text	<input checked="" type="checkbox"/>
	borrowed_day	date	<input checked="" type="checkbox"/>
	due_date	date	<input checked="" type="checkbox"/>
	return_date	date	<input checked="" type="checkbox"/>
	status	varchar(20)	<input checked="" type="checkbox"/>
	created_at	datetime2(7)	<input checked="" type="checkbox"/>
	created_by	varchar(255)	<input checked="" type="checkbox"/>
	updated_at	datetime2(7)	<input checked="" type="checkbox"/>
	updated_by	varchar(255)	<input checked="" type="checkbox"/>
	deleted_at	datetime2(7)	<input checked="" type="checkbox"/>
	deleted_by	varchar(255)	<input checked="" type="checkbox"/>
			<input type="checkbox"/>

2. <Employee>

Employee			
	Column Name	Data Type	Allow Nulls
	uu_id	uniqueidentifier	<input type="checkbox"/>
	name	varchar(255)	<input type="checkbox"/>
	dob	date	<input checked="" type="checkbox"/>
	id	int	<input checked="" type="checkbox"/>
	address	varchar(255)	<input checked="" type="checkbox"/>
	phone	varchar(20)	<input checked="" type="checkbox"/>
	email	nvarchar(70)	<input checked="" type="checkbox"/>
	role	varchar(20)	<input checked="" type="checkbox"/>
	created_at	datetime2(7)	<input checked="" type="checkbox"/>
	created_by	varchar(255)	<input checked="" type="checkbox"/>
	updated_at	datetime2(7)	<input checked="" type="checkbox"/>
	updated_by	varchar(255)	<input checked="" type="checkbox"/>
	deleted_at	datetime2(7)	<input checked="" type="checkbox"/>
	deleted_by	varchar(255)	<input checked="" type="checkbox"/>
			<input type="checkbox"/>

2. <Master Data>

MasterData			
	Column Name	Data Type	Allow Nulls
🔑	uu_id	uniqueidentifier	<input type="checkbox"/>
	type	varchar(255)	<input type="checkbox"/>
	value	varchar(255)	<input type="checkbox"/>
	key_col	varchar(255)	<input type="checkbox"/>
	description	nvarchar(255)	<input type="checkbox"/>
	created_at	datetime2(7)	<input checked="" type="checkbox"/>
	created_by	varchar(255)	<input checked="" type="checkbox"/>
	updated_at	datetime2(7)	<input checked="" type="checkbox"/>
	updated_by	varchar(255)	<input checked="" type="checkbox"/>
	deleted_at	datetime2(7)	<input checked="" type="checkbox"/>
	deleted_by	varchar(255)	<input checked="" type="checkbox"/>
			<input type="checkbox"/>

2. <Logs>

Logs			
	Column Name	Data Type	Allow Nulls
🔑	uu_id	uniqueidentifier	<input type="checkbox"/>
	action_name	varchar(255)	<input checked="" type="checkbox"/>
	status	varchar(50)	<input checked="" type="checkbox"/>
	link	varchar(255)	<input checked="" type="checkbox"/>
	created_at	datetime2(7)	<input checked="" type="checkbox"/>
	created_by	varchar(255)	<input checked="" type="checkbox"/>
	updated_at	datetime2(7)	<input checked="" type="checkbox"/>
	updated_by	varchar(255)	<input checked="" type="checkbox"/>
	deleted_at	datetime2(7)	<input checked="" type="checkbox"/>
	deleted_by	varchar(255)	<input checked="" type="checkbox"/>
			<input type="checkbox"/>

II. Code Designs

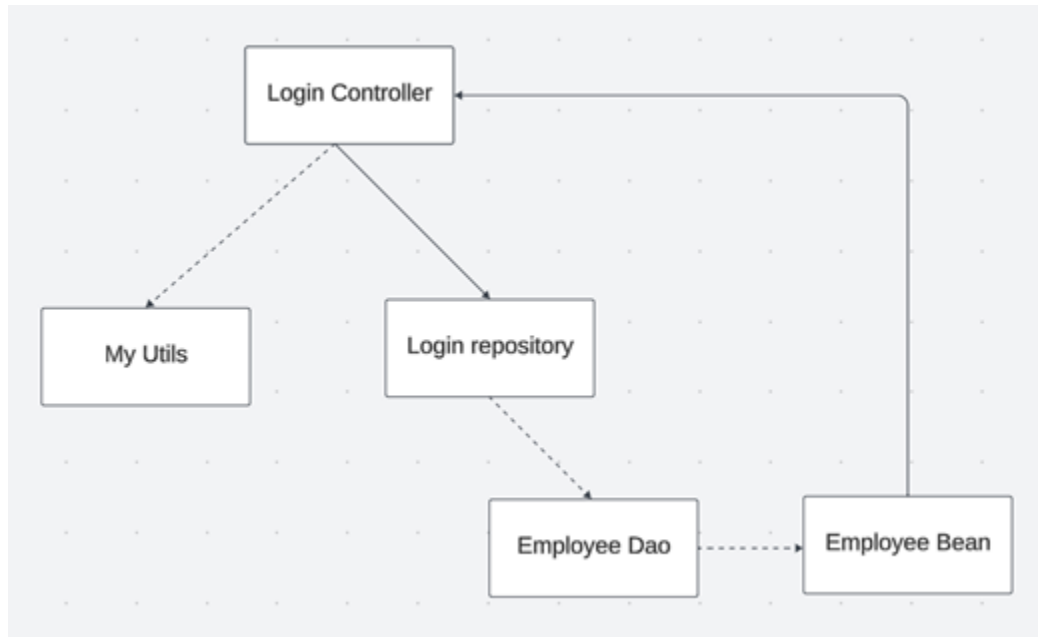
1. Login

a. Screen Design

Table <x>: <Screen Name> Definition

No	Object/Control Name	Type	Required	Length	Description
1	Title	Textview	Yes	10	Show tittle
2	Username	Edittext	Yes	100	Enter username
3	Password	Edittext	Yes	100	Enter password
4	Remember Password	Checkbox	No		Save account info
5	Login	Button			Login to app
6	Error	Textview	Yes	20	Show error when login
7	Forgot password	Textview	No	20	Direct to reset password page when click

b. Class Diagram



c. Class Specifications

EmployeeRepository class

[Provide the detailed description for the class methods]

No	Method	Description
01	EmployeeRepository	<i>Initial repository:</i> <ul style="list-style-type: none"> - <i>Input: context</i> - <i>Output: not</i>
02	getAll ()	<i>Get all employee</i> <ul style="list-style-type: none"> - <i>Input: no</i> - <i>Output: List<Employee></i>
03	getEmployeeByUsernameAndPassword	<i>Get employee by username and password to check login:</i> <i>Input: username, password</i> <i>Output: Employee</i>

04	<code>getEmployeeById()</code>	<i>Get employee by id:</i> <i>Input: id</i> <i>Output: employee</i>
05	<code>updateEmployee</code>	<i>Update employee</i> <i>Input: Employee</i> <i>Output: not</i>

LoginActivity Class

Class Methods

[Provide the detailed description for the class methods]

No	Method	Description
01	<code>onCreate</code>	<i>Build login activity</i>
02	<code>validateAccountAndDirect</code>	<i>Validate account and direct to dashboard if it available:</i> <i>Input: Username, password</i> <i>Output: Not</i>
03	<code>recoverUsernameAndPassword</code>	<i>Direct to Forgot Password page if click it on screen:</i>

d. Sequence Diagram(s)

e. Database queries

```
@Query("SELECT * FROM Employee")
```

```
@Query("SELECT * FROM Employee WHERE uuid = :uuid")
```

```
@Query("SELECT * FROM Employee WHERE email = :username and password =
```

```
:password")
```

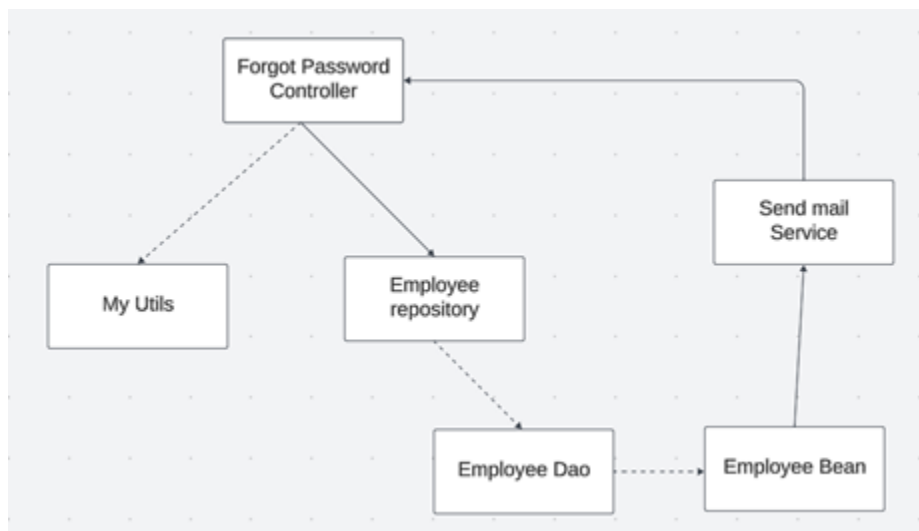
2. Forgot Password

a. Screen Design

Table <x>: <Screen Name> Definition

No	Object/Control Name	Type	Required	Length	Description
1	Title	Textview	Yes	10	Show title
2	Email	Edittext	Yes	100	Enter email
3	Forgot password	Button			Reset Password

b. Class Diagram



c. Class Specifications

EmployeeRepository class

Class Methods

No	Method	Description
01	EmployeeRepository	<i>Initial repository:</i> - <i>Input: context</i> - <i>Output: not</i>
02	getAll ()	<i>Get all employee</i> - <i>Input: no</i> - <i>Output: List<Employee></i>
03	getEmployeeByUsernameAndPassword	<i>Get employee by username and password to check login:</i> <i>Input: username, password</i> <i>Output: Employee</i>
04	getEmployeeById ()	<i>Get employee by id:</i> <i>Input: id</i> <i>Output: employee</i>
05	updateEmployee	<i>Update employee</i> <i>Input: Employee</i> <i>Output: not</i>

ForgotPasswordActivity Class**Class Methods**

No	Method	Description
01	onCreate	<i>Build forgot password activity</i>

02	isValidEmail	<p><i>Validate email available in system:</i></p> <p><i>Input: email</i></p> <p><i>Output: boolean true false</i></p>
03	sendEmail	<p><i>Send mail with new password:</i></p> <p><i>Input: recipientEmail, newPassword</i></p> <p><i>Output: not</i></p>
04	sendEmail	<p><i>Send mail with new password:</i></p> <p><i>Input: recipientEmail, newPassword</i></p> <p><i>Output: not</i></p>
03	generateNewPassword	<p><i>Generate new password:</i></p> <p><i>Input: not</i></p> <p><i>Output: new password</i></p>

d. Sequence Diagram(s)

e. Database queries

```
@Query("SELECT * FROM Employee WHERE email = :email")
Employee getEmployeeByEmail(String email);
```

```
@Update
void updateEmployee(Employee employee);
```

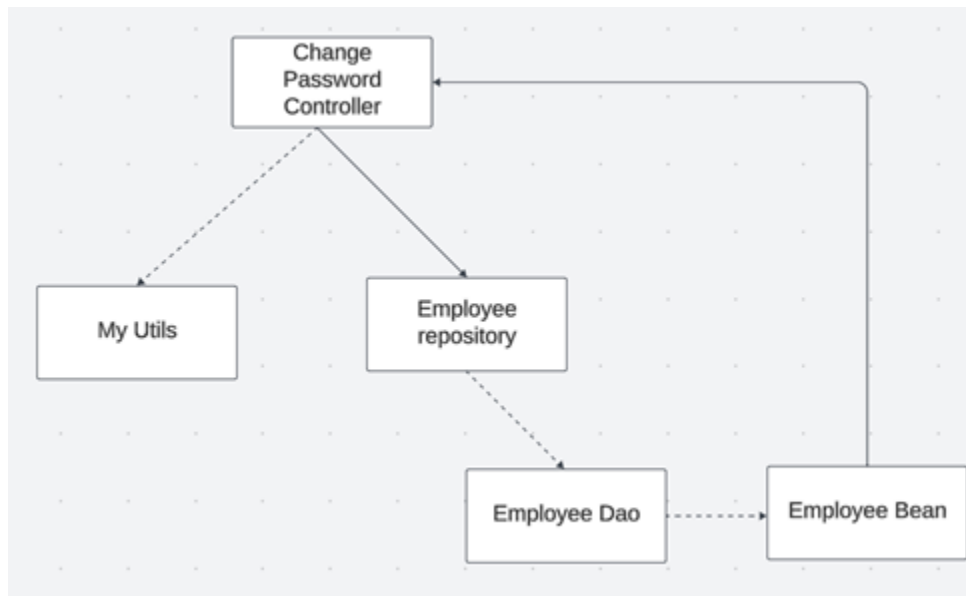
3. Change Password

a. Screen Design

Table <x>: <Screen Name> Definition

No	Object/Control Name	Type	Required	Length	Description
1	Old Password	Edit Text	Yes	100	Enter old pass
2	New Password	Edittext	Yes	100	Enter new pass
3	Confirm password	Edittext	Yes	100	Enter new pass again
4	Change password	Button			Change pass

b. Class Diagram



c. Class Specifications

EmployeeRepository class

Class Methods

No	Method	Description
01	<code>EmployeeRepository</code>	<i>Initial repository:</i> <ul style="list-style-type: none"> - <i>Input: context</i> - <i>Output: not</i>

02	<code>getAll()</code>	<i>Get all employee</i> - <i>Input: no</i> - <i>Output: List<Employee></i>
03	<code>getEmployeeByUsernameAndPassword</code>	<i>Get employee by username and password to check login:</i> <i>Input: username, password</i> <i>Output: Employee</i>
04	<code>getEmployeeById()</code>	<i>Get employee by id:</i> <i>Input: id</i> <i>Output: employee</i>
05	<code>updateEmployee</code>	<i>Update employee</i> <i>Input: Employee</i> <i>Output: not</i>

ChangePasswordActivity Class

Class Methods

No	Method	Description
01	<code>onCreate</code>	<i>Build change password activity</i>
02	<code>validatePasswords</code>	<i>Validate password available in system:</i> <i>Input:</i> String oldPassword, String newPassword, String confirmPassword <i>Output: boolean true false</i>
03	<code>changePassword</code>	<i>Change password</i> <i>Input:</i> newPassword

		<i>Output: not</i>
--	--	--------------------

d. Sequence Diagram(s)

e. Database queries

```
@Query("SELECT * FROM Employee WHERE uuid = :uuid")
Employee getEmployeeById(Long uuid);
```

```
@Update
void updateEmployee(Employee employee);
```

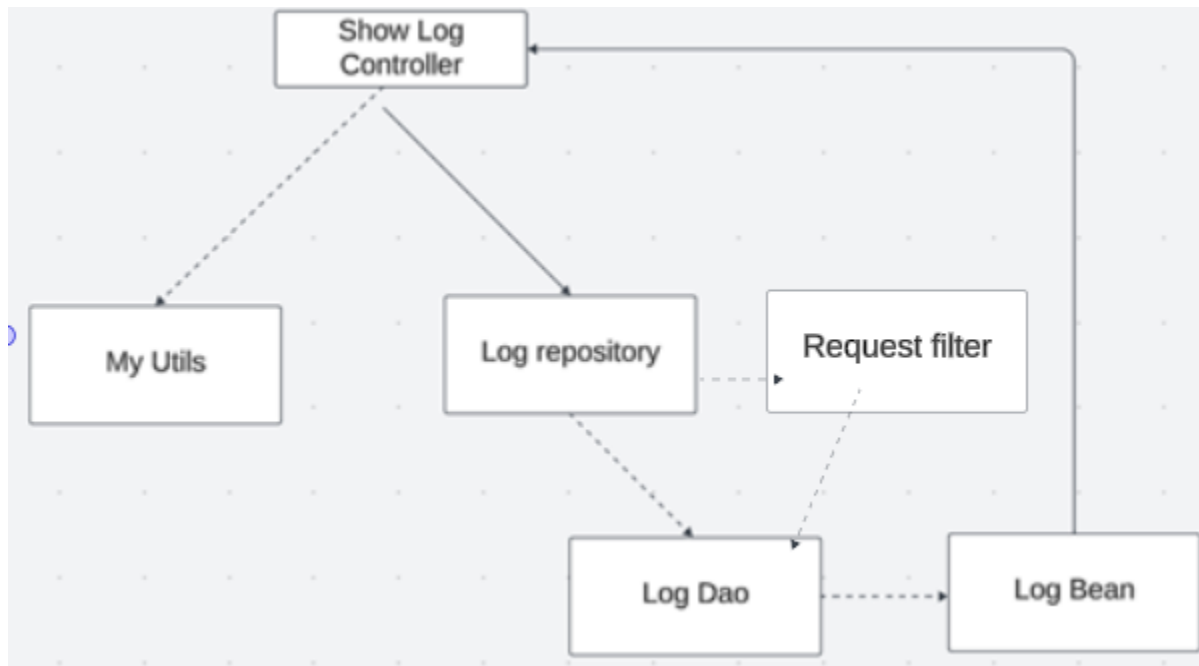
4. Show Log

a. Screen Design

Table <x>: <Screen Name> Definition

No	Object/Control Name	Type	Required	Length	Description
1	History	ViewText	Yes	100	Show title
2	Start Date	DatePicker	No		Start date
3	End Date	DatePicker	No		End date
4	Search	EditText	No	100	Enter action name to search

b. Class Diagram



c. Class Specifications

LogRepository class

Class Methods

No	Method	Description
01	<code>LogRepository</code>	<i>Initial repository:</i> <ul style="list-style-type: none"> - <i>Input: context</i> - <i>Output: not</i>
02	<code>insertLog()</code>	<i>Insert log</i> <i>Input:</i> String actionName, Long createdBy, String before, String after <i>Output: not</i>
03	<code>getEmployeeByUsernameAndPassword</code>	<i>Get employee by username and password to check login:</i> <i>Input: username, password</i> <i>Output: Employee</i>

04	getListLog	<i>Get log list</i> <i>Input: not</i> <i>Output: log list</i>
----	------------	---

ShowLogActivity Class

Class Methods

No	Method	Description
01	onCreate	<i>Build change password activity</i>
02	convertLogObject	<i>Convert form log entity to logShow object to show on screen</i> <i>Input:</i> List<Logs> logEntities <i>Output:</i> List<LogShow> logShowList
03	applyFilter	<i>Apply filter for list</i> <i>Input:</i> List<LogShow> logShowList <i>Output: not</i> List<LogShow> logShowListApplyFilter
04	showStartDatePicker	<i>Show calender to pick</i>
05	showEndDatePicker	<i>Show calender to pick</i>
06	updateLogList	<i>Update list when list update or apply filter</i>

d. Sequence Diagram(s)

e. Database queries

```
@Query("SELECT * FROM Logs")
List<Logs> getAllLogs();

@Insert(onConflict = OnConflictStrategy.REPLACE)
void insertLog(Logs log);
```

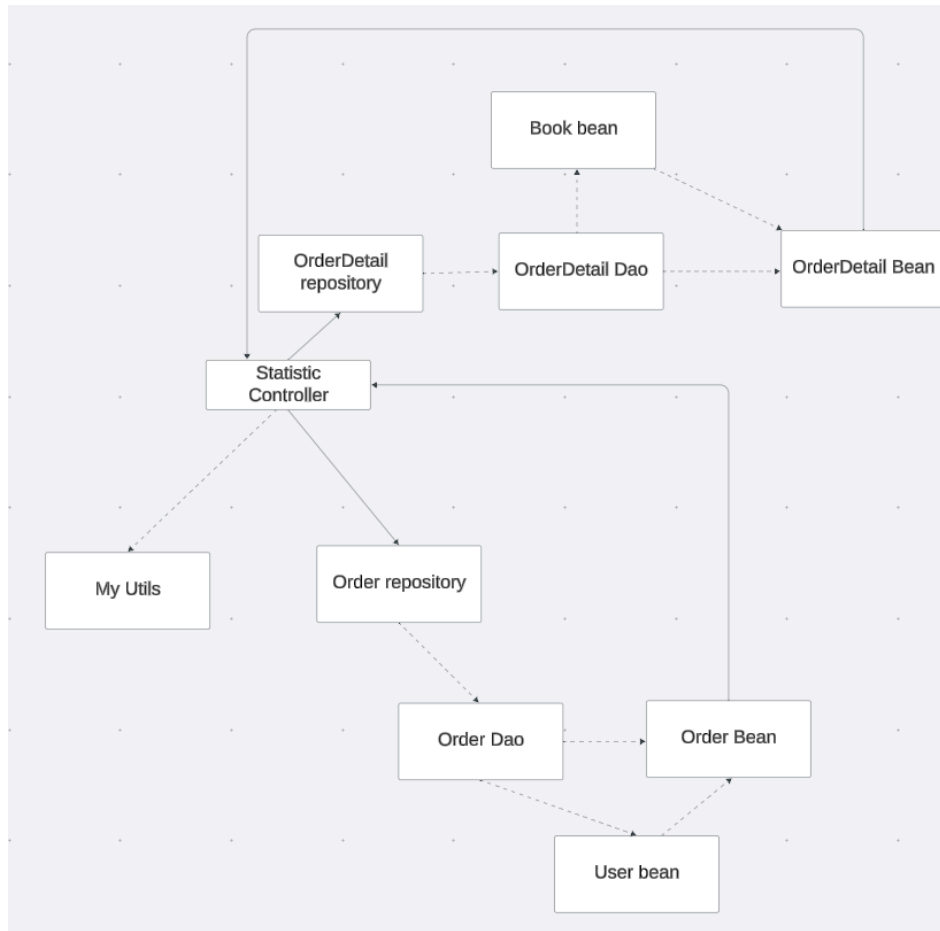
5. Statistic

a. Screen Design

Table <x>: <Screen Name> Definition

No	Object/Control Name	Type	Required	Length	Description
1	List customer	ViewText	Yes		Show list customer when click
2	List book	ViewText	Yes		Show list book when click
3	Table content	Table	Yes		Show info

b. Class Diagram



c. Class Specifications

OrderDetailRepository class

Class Methods

No	Method	Description
01	OrderDetailRepository	<i>Initial repository:</i> <ul style="list-style-type: none"> - <i>Input: context</i> - <i>Output: not</i>
02	getBookOrderDetailSummary	<i>Get list join between book and order detail</i> <p><i>Input: not</i></p> <p><i>Output: List<BookOrderDetailSummary></i></p>

OrderRepository class

Class Methods

No	Method	Description
01	OrderRepository	<i>Initial repository:</i> - <i>Input: context</i> - <i>Output: not</i>
02	getCustomerOrderSummary	<i>Get list join between order and customer</i> <i>Input: not</i> <i>Output: List< CustomerOrderSummary></i>

StatisticActivity Class

Class Methods

No	Method	Description
01	onCreate	<i>Build change password activity</i>
02	displayUserList	<i>Shows a list of loyal users and the number of times they borrowed.</i> <i>Input: Not</i> <i>Output: Not</i>
03	displayBookList	<i>Shows a list of favorite books.</i> <i>Input: Not</i> <i>Output: Not</i>
04	createTableCell	<i>Creates a table cell with the specified content</i> <i>Input: Content</i>

		<i>Output:</i> TextView
--	--	-------------------------

d. Sequence Diagram(s)

e. Database queries

```

@Query("SELECT b.book_name AS book_name, b.uuid as book_uuid,
COUNT(od.book_uuid) AS quantity " +
    "FROM 'OrderDetails' od " +
    "INNER JOIN Book b ON od.book_uuid = b.uuid " +
    "GROUP BY od.book_uuid " +
    "ORDER BY quantity desc"
)
List<BookOrderDetailSummary> getBookOrderDetailSummary();

@Query("SELECT c.name AS customer_name, c.uuid as customer_uuid,
COUNT(o.customer_id) AS quantity " +
    "FROM 'Order' o " +
    "INNER JOIN Customer c ON o.customer_id = c.uuid " +
    "GROUP BY o.customer_id " +
    "ORDER BY quantity desc"
)
List<CustomerOrderSummary> getCustomerOrderSummary();

```

6. Home

a. Screen Design

Table <x>: <Screen Name> Definition

No	Object/Control Name	Type	Required	Length	Description
1	Image background	ImageView	Yes		Wallpaper when opening the app
2	Name Project	TextView	Yes		Display name of app
3	Login	Button	Yes		User chooses to log in to account
4	About us	Button	Yes		User chooses to view information of app.

b. Class Diagram



c. Class Specifications

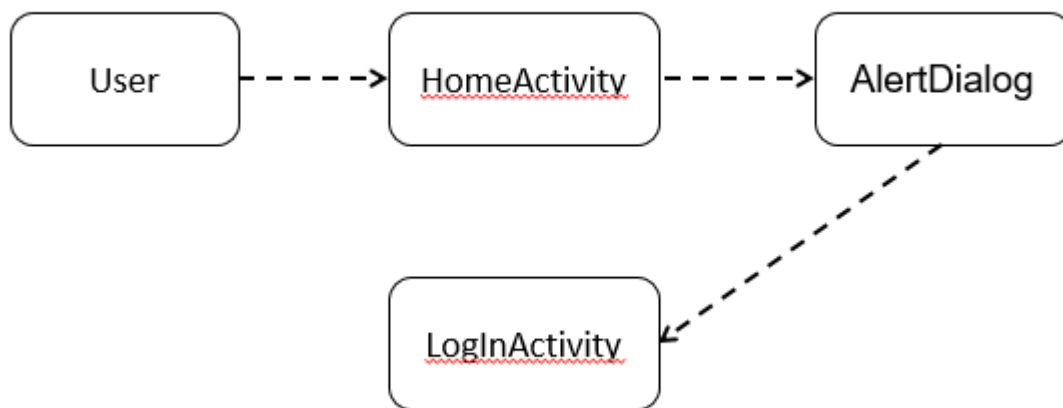
HomeActivity class

Class Methods

No	Method	Description
01	buttonLogInOnClick	Handle the button click event to navigate to the login activity - Input: not - Output: not
02	btnAboutOnClick	Handle the button click event to display the About Us information in an AlertDialog Input: not

		<i>Output: not</i>
--	--	--------------------

d. Sequence Diagram(s)



e. Database queries

*/*Nothing*/*

7. List Customer

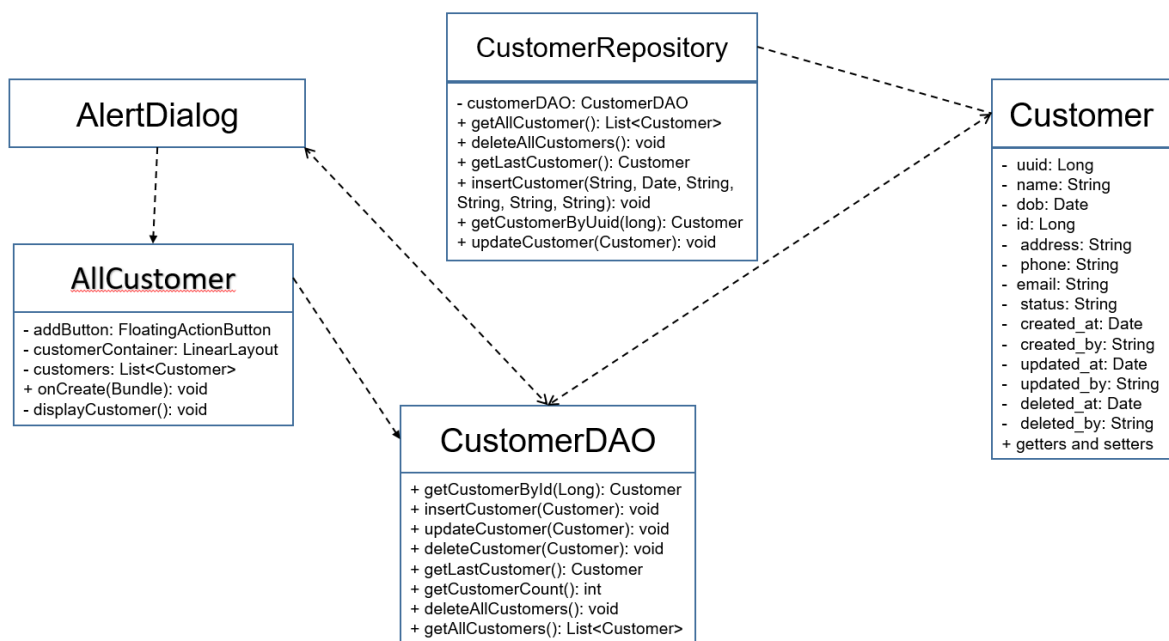
a. Screen Design

Table <x>: <Screen Name> Definition

No	Object/Control Name	Type	Required	Length	Description
1	Header	ActionBar	Yes		Display back and label
2	Add Customer	FloatingActionButton	Yes		User click to add new customer.
3	Delete Customer	ImageButton	Yes		User click to delete customer.

4	Edit Customer	ImageButton	Yes		User click to edit customer.
5	View Customer	ImageButton	Yes		User click to view profile customer.
6	List Customer	ScrollView	Yes		List all customers.
7	Id Customer	TextView	Yes		Show id of customer.
8	Name Customer	TextView	Yes		Show name of customer.
9	Status	TextView	Yes		Show status of customer.

b. Class Diagram



c. Class Specifications

AllCustomer class

Class Methods

No	Method	Description
----	--------	-------------

01	displayCustomer()	<i>Description: Displays the list of customers in the customerContainer.</i> <i>Input: None</i> <i>Output: None</i>
----	-------------------	---

CustomerRepository class

Class Methods

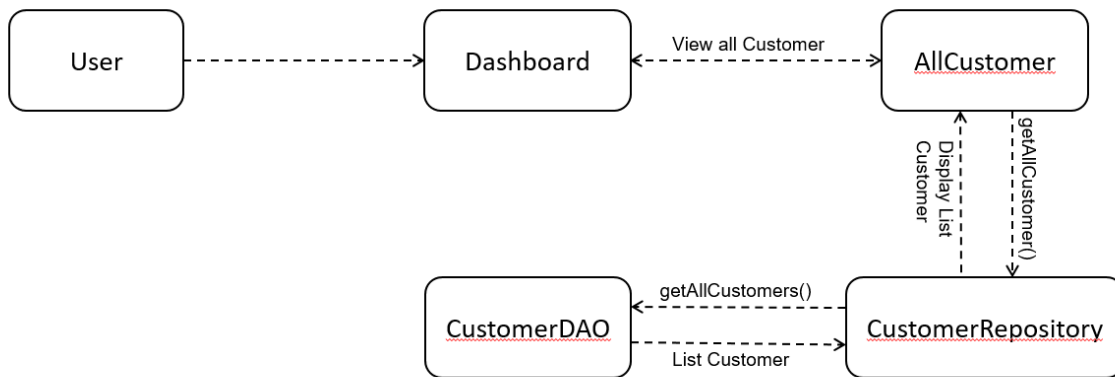
No	Method	Description
01	getAllCustomer(): List<Customer>	<i>Description: Retrieves all customers from the database.</i> <i>Input: None</i> <i>Output: List of Customer objects</i>
02	deleteAllCustomers(): void	<i>Description: Deletes all customers from the database.</i> <i>Input: None</i> <i>Output: None</i>
03	getLastCustomer(): Customer	<i>Description: Retrieves the last customer added to the database.</i> <i>Input: None</i> <i>Output: Customer object</i>
04	insertCustomer(String, Date, String, String, String, String): void	<i>Description: Inserts a new customer into the database.</i> <i>Input: String name, Date dob, String phone, String email, String address, String status</i> <i>Output: None</i>
05	getCustomerByUuid(long): Customer	<i>Description: Retrieves a customer by their UUID.</i> <i>Input: long uuid</i> <i>Output: Customer object</i>
06	updateCustomer(Customer): void	<i>Description: Updates an existing customer in the database.</i> <i>Input: Customer customer</i> <i>Output: None</i>

CustomerDAO class

Class Methods

No	Method	Description
01	<code>getCustomerById(Long) : Customer</code>	<i>Description: Retrieves a customer by their ID.</i> <i>Input: Long id</i> <i>Output: Customer object</i>
02	<code>insertCustomer(Customer) : void</code>	<i>Description: Inserts a new customer into the database.</i> <i>Input: Customer customer</i> <i>Output: None</i>
03	<code>updateCustomer(Customer) : void</code>	<i>Description: Updates an existing customer in the database.</i> <i>Input: Customer customer</i> <i>Output: None</i>
04	<code>deleteCustomer(Customer) : void</code>	<i>Description: Deletes a customer from the database.</i> <i>Input: Customer customer</i> <i>Output: None</i>
05	<code>getLastCustomer() : Customer</code>	<i>Description: Retrieves the last customer added to the database.</i> <i>Input: None</i> <i>Output: Customer object</i>
06	<code>getCustomerCount() : int</code>	<i>Description: Retrieves the number of customers in the database.</i> <i>Input: None</i> <i>Output: int</i>
07	<code>deleteAllCustomers() : void</code>	<i>Description: Deletes all customers from the database.</i> <i>Input: None</i> <i>Output: None</i>
08	<code>getAllCustomers() : List<Customer></code>	<i>Description: Retrieves all customers from the database.</i> <i>Input: None</i> <i>Output: List of Customer objects</i>

d. Sequence Diagram(s)



e. Database queries

```
@Query("SELECT * FROM Customer WHERE uuid = :uuid")
```

```
Customer getCustomerById(Long uuid);
```

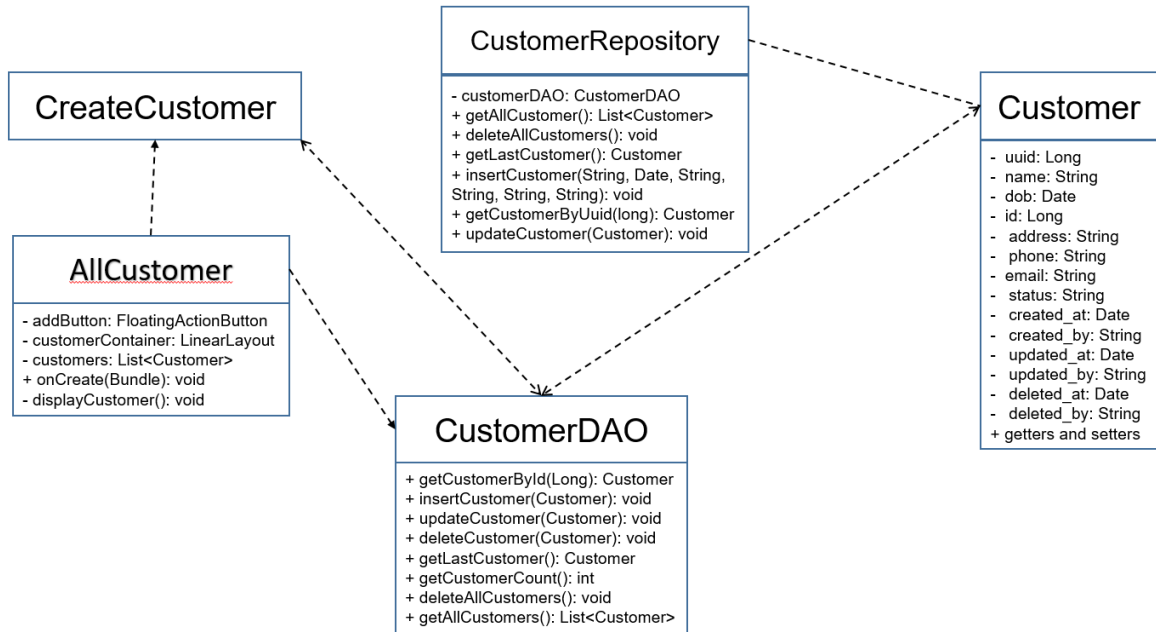
8. Create Customer

a. Screen Design

Table <x>: <Screen Name> Definition

No	Object/Control Name	Type	Required	Length	Description
1	Header	ActionBar	Yes		Display back and label
2	Information field	TextView	Yes		Information fields as name, status, dob, phone,email,address.
3	Add information new customer	EditText	Yes		User add Information of new customer as name, status, dob, phone,email,address.
4	Button add	Button	Yes		User click to confirm add a new customer.

b. Class Diagram



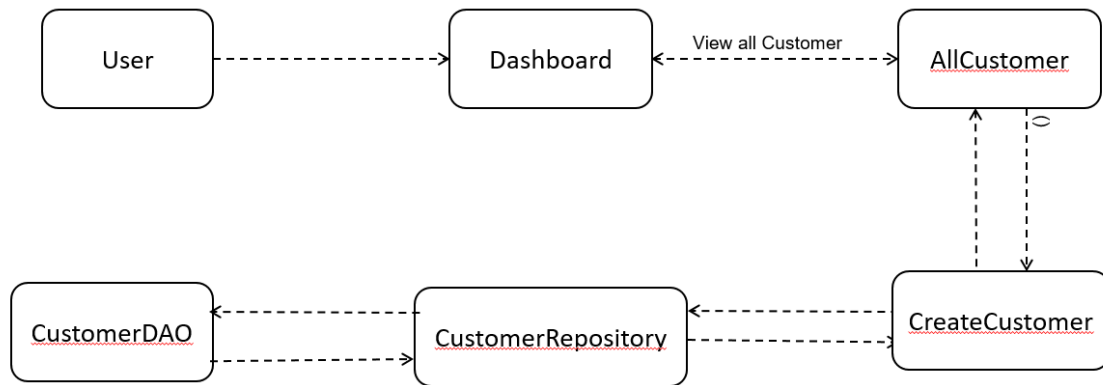
c. Class Specifications

CreateCustomer class

Class Methods

No	Method	Description
01	onClick (addButton)	Handle the onClick event for the "Add" button. Retrieve customer information from the EditText fields, validate the input, and insert the new customer into the database using the CustomerRepository. Display a success message and navigate to the AllCustomer activity upon successful insertion. Display an error message if insertion fails.
02	showDatePickerDialog	Display a DatePickerDialog to select the customer's date of birth (DOB) and set the selected date to the DOB EditText.

d. Sequence Diagram(s)



e. Database queries

@Insert(onConflict = OnConflictStrategy.REPLACE)

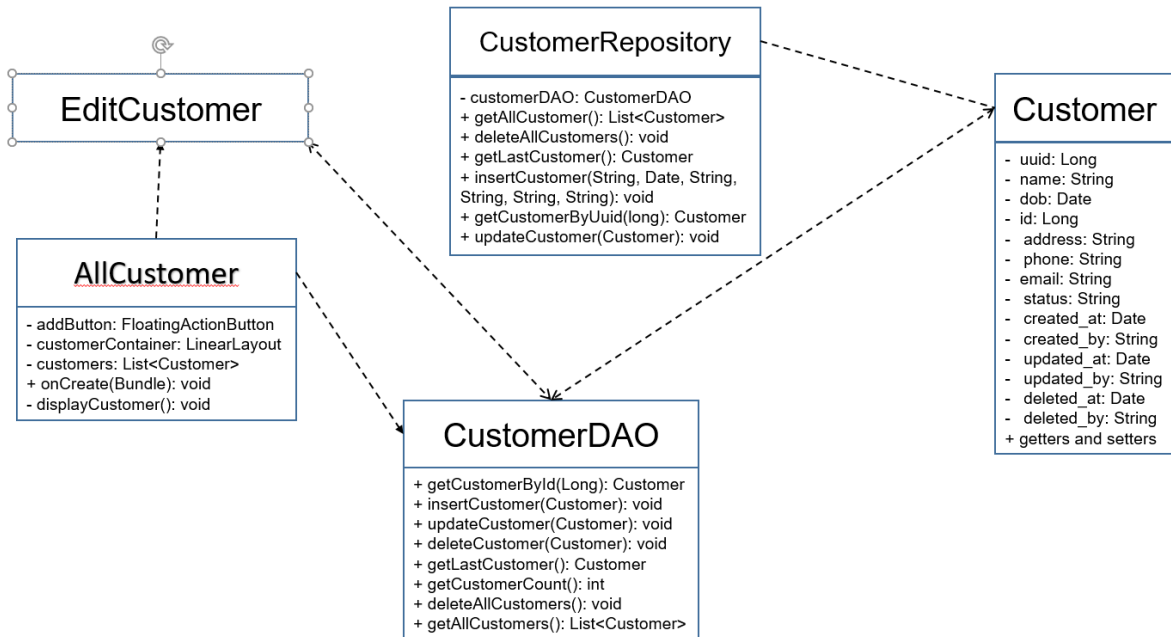
9. Edit Customer

a. Screen Design

Table <x>: <Screen Name> Definition

No	Object/Control Name	Type	Required	Length	Description
1	Header	ActionBar	Yes		Display back and label
2	Information field	TextView	Yes		Information fields as name, status, dob, phone,email,address.
3	Edit information customer	EditText	Yes		User edit information of customer as name, status, dob, phone,email,address.
4	Button Edit	Button	Yes		User click to confirm edit customer.

b. Class Diagram



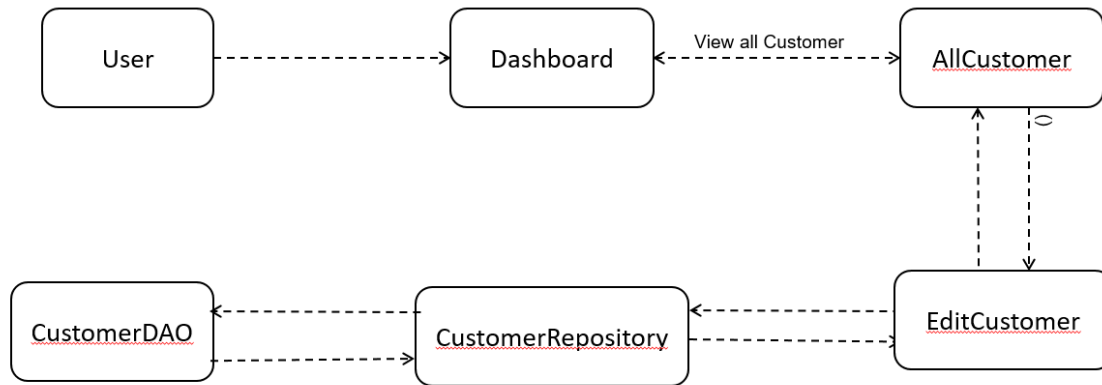
c. Class Specifications

EditCustomer class

Class Methods

No	Method	Description
01	updateCustomer	<p>Retrieve the updated customer information from the EditText fields, validate the input, and update the customer object. Update the customer in the database using the CustomerRepository. Display a success message and navigate to the AllCustomer activity upon successful update. Display an error message if the update fails.</p> <p>Input: Customer</p> <p>Output: None</p>

d. Sequence Diagram(s)



e. Database queries

@Query("SELECT * FROM Employee WHERE uuid = :uuid")

@Update(onConflict = OnConflictStrategy.REPLACE)

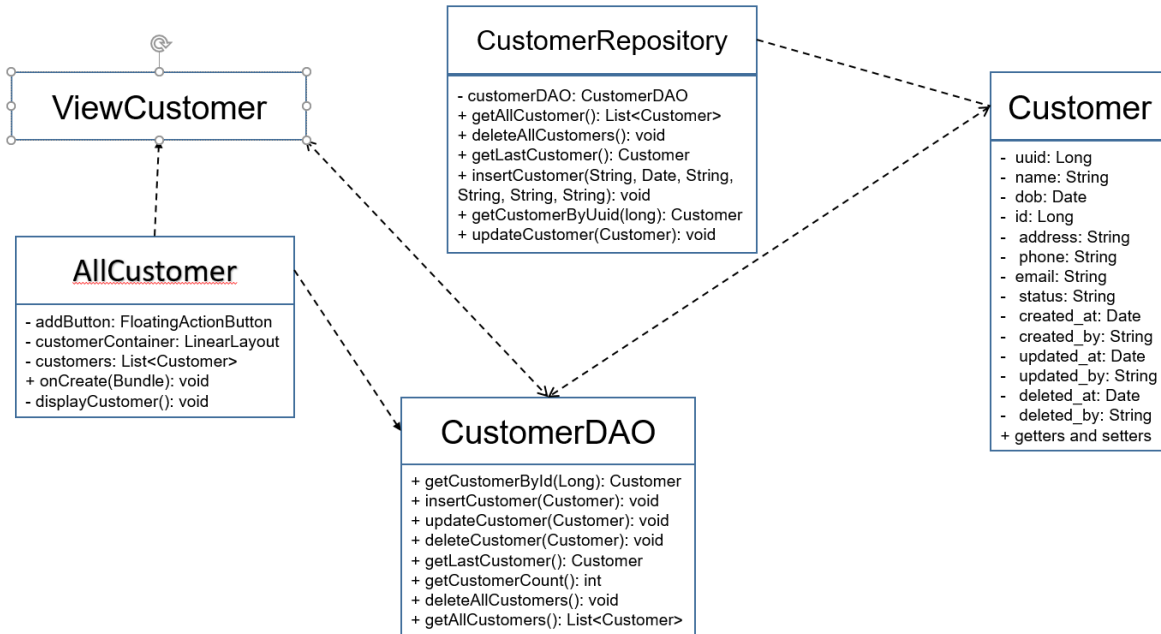
10. View Customer

a. Screen Design

Table <x>: <Screen Name> Definition

No	Object/Control Name	Type	Required	Length	Description
1	Header	ActionBar	Yes		Display back and label
2	Information field	TextView	Yes		Information fields as name, status, dob, phone,email,address.

b. Class Diagram



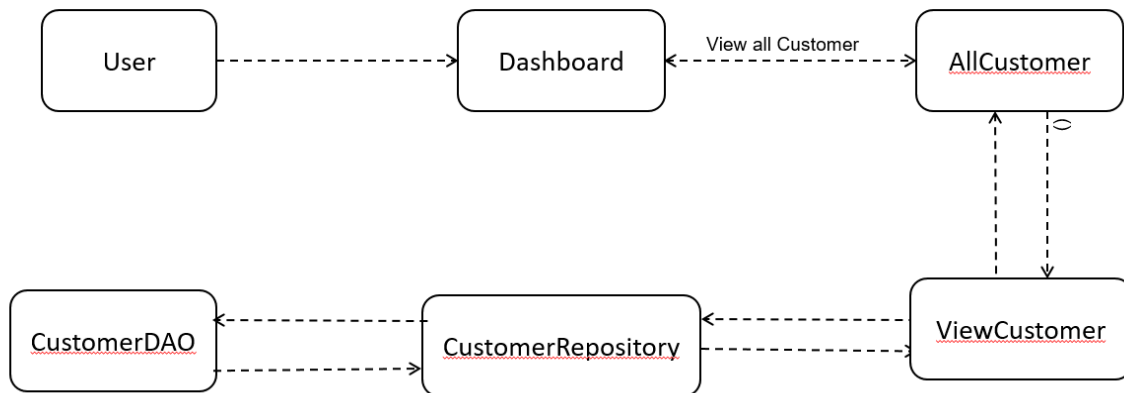
c. Class Specifications

ViewCustomer class

Class Methods

No	Method	Description
01	onCreate	Initialize the activity, set up the layout, and bind the UI components. Retrieve the customer's information using the UUID passed from the intent and populate the TextView fields with the customer's data.

d. Sequence Diagram(s)



e. Database queries

@Query("SELECT * FROM Employee WHERE uuid = :uuid")

11. List Book

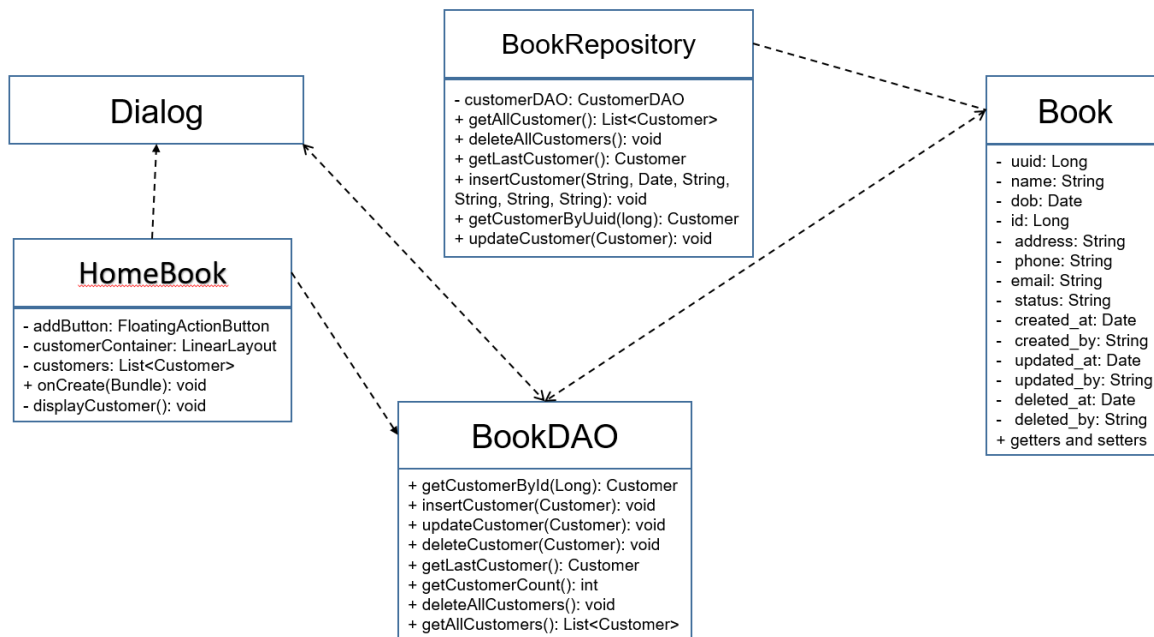
a. Screen Design

Table <x>: <Screen Name> Definition

No	Object/Control Name	Type	Required	Length	Description
1	Header	ActionBar	Yes		Display back and label
2	Add <i>Book</i>	FloatingActionButton	Yes		User click to add new <i>Book</i> .
3	Delete <i>Book</i>	ImageButton	Yes		User click to delete <i>Book</i> .
4	Edit <i>Book</i>	ImageButton	Yes		User click to edit <i>Book</i> .
5	View <i>Book</i>	ImageButton	Yes		User click to view profile <i>Book</i> .
6	List <i>Book</i>	ScrollView	Yes		List all <i>Book</i> .

7	Id <i>Book</i>	TextView	Yes		Show id of <i>Book</i> .
8	Name <i>Book</i>	TextView	Yes		Show name of <i>Book</i> .
9	Status	TextView	Yes		Show status of <i>Book</i> .

b. Class Diagram



c. Class Specifications

HomeBook class

Class Methods

No	Method	Description
01	displayBook ()	<p><i>Description: Displays the list of Books in the BookContainer.</i></p> <p><i>Input: None</i></p> <p><i>Output: None</i></p>

BookRepository class

Class Methods

No	Method	Description
01	<code>getAllBook() : List<Book></code>	Description: Retrieves all Books from the database. Input: None Output: List of Book objects
02	<code>deleteAllBooks() : void</code>	Description: Deletes all Books from the database. Input: None Output: None
03	<code>getLastBook() : Book</code>	Description: Retrieves the last Book added to the database. Input: None Output: Book object
04	<code>insertBook(String, Date, String, String, String, String) : void</code>	Description: Inserts a new Book into the database. Input: String name, Date dob, String phone, String email, String address, String status Output: None
05	<code>getBookByUuid(long) : Book</code>	Description: Retrieves a Book by their UUID. Input: long uuid Output: Book object
06	<code>updateBook(Book) : void</code>	Description: Updates an existing Book in the database. Input: Book Book Output: None

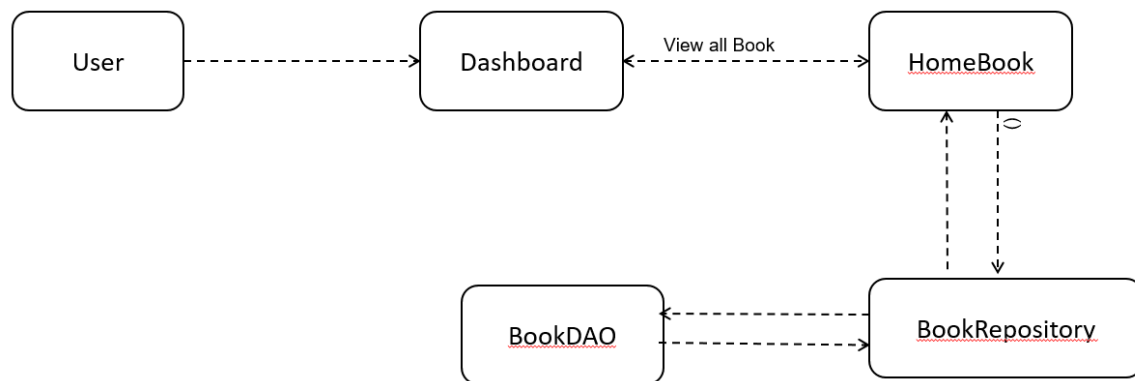
BookDAO class

Class Methods

No	Method	Description
01	<code>getBookById(Long) : Book</code>	Description: Retrieves a book by their ID. Input: Long id Output: book object

02	<code>insertBook(Book) : void</code>	<i>Description: Inserts a new book into the database.</i> <i>Input: book book</i> <i>Output: None</i>
03	<code>updateBook(Book) : void</code>	<i>Description: Updates an existing book in the database.</i> <i>Input: Book book</i> <i>Output: None</i>
04	<code>deleteBook(Book) : void</code>	<i>Description: Deletes a book from the database.</i> <i>Input: Book book</i> <i>Output: None</i>
05	<code>getLastBook() : Book</code>	<i>Description: Retrieves the last book added to the database.</i> <i>Input: None</i> <i>Output: book object</i>
06	<code>getBookCount() : int</code>	<i>Description: Retrieves the number of books in the database.</i> <i>Input: None</i> <i>Output: int</i>
07	<code>deleteAllBooks() : void</code>	<i>Description: Deletes all books from the database.</i> <i>Input: None</i> <i>Output: None</i>
08	<code>getAllBooks() : List<Book></code>	<i>Description: Retrieves all books from the database.</i> <i>Input: None</i> <i>Output: List of Book objects</i>

d. Sequence Diagram(s)



e. Database queries

```
@Query("SELECT * FROM Book WHERE uuid = :uuid")
```

```
Book getbookById(Long uuid);
```

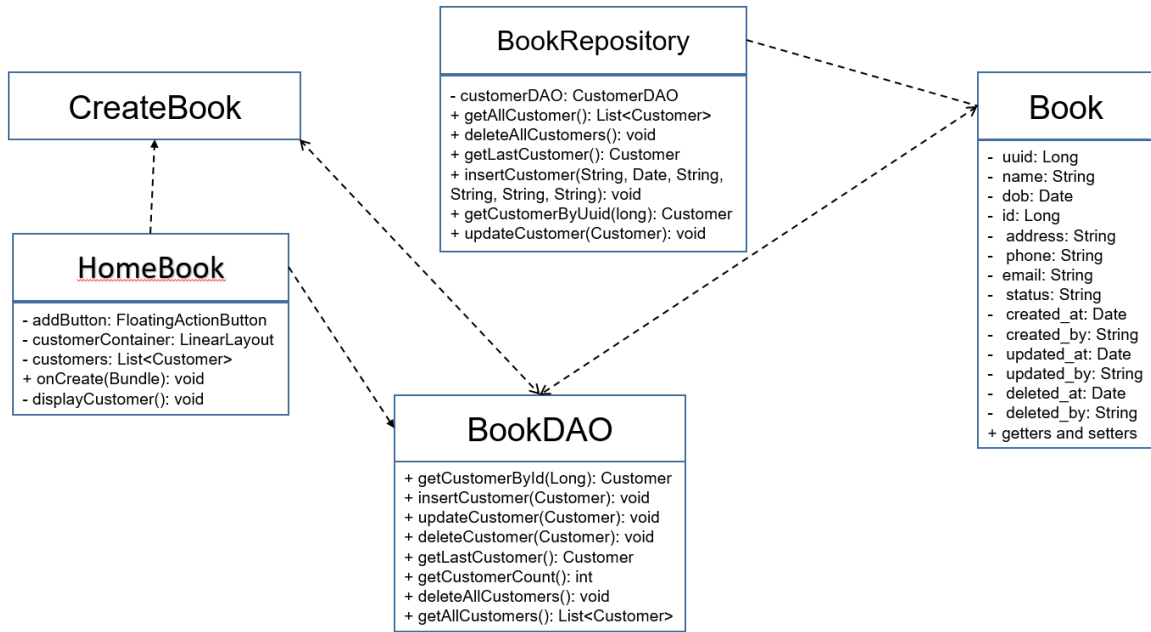
12. Create book

a. Screen Design

Table <x>: <Screen Name> Definition

No	Object/Control Name	Type	Required	Length	Description
1	Header	ActionBar	Yes		Display back and label
2	Information field	TextView	Yes		Information fields as name, status, ...
3	Add information new book	EditText	Yes		User add Information of new book as name, status,
4	Button add	Button	Yes		User click to confirm add a new book.

b. Class Diagram



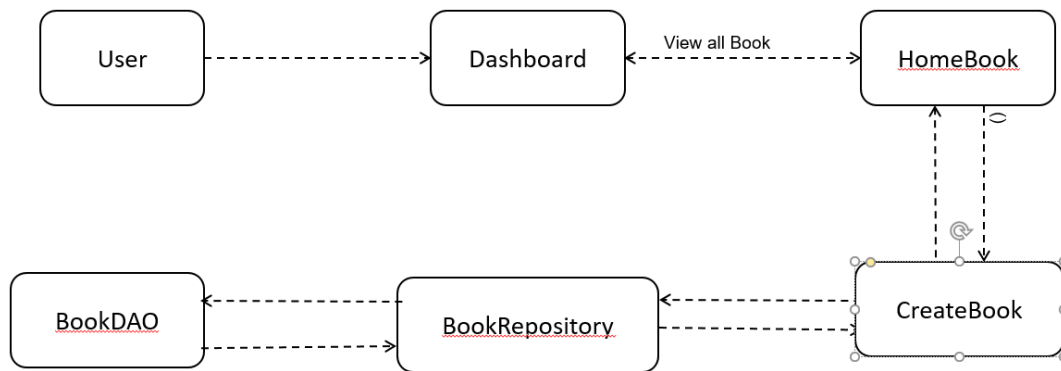
c. Class Specifications

CreateBook class

Class Methods

No	Method	Description
01	onClick (addButton)	Handle the onClick event for the "Add" button. Retrieve book information from the EditText fields, validate the input, and insert the new book into the database using the BookRepository. Display a success message and navigate to the HomeBookr activity upon successful insertion. Display an error message if insertion fails.
02	showDatePickerDialog	Display a DatePickerDialog to select the book's date of birth (DOB) and set the selected date to the DOB EditText.

d. Sequence Diagram(s)



e. Database queries

@Insert(onConflict = OnConflictStrategy.REPLACE)

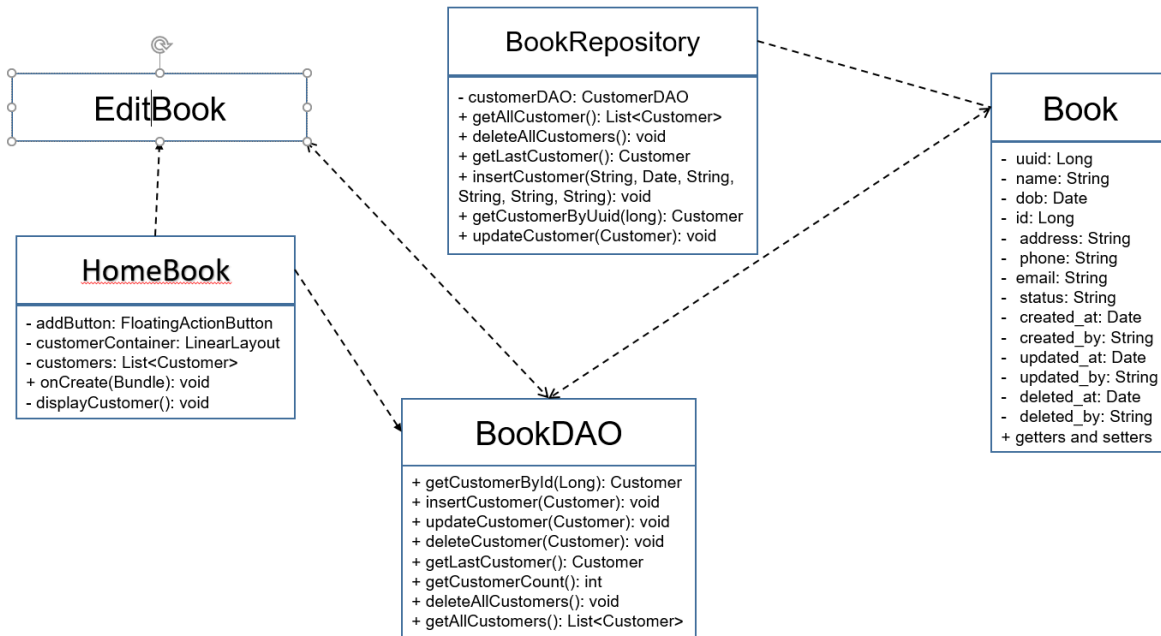
13. Edit Book

a. Screen Design

Table <x>: <Screen Name> Definition

No	Object/Control Name	Type	Required	Length	Description
1	Header	ActionBar	Yes		Display back and label
2	Information field	TextView	Yes		Information fields as name, status, ...
3	Edit information book	EditText	Yes		User edit information of book as name, status,...
4	Button Edit	Button	Yes		User click to confirm edit book.

b. Class Diagram



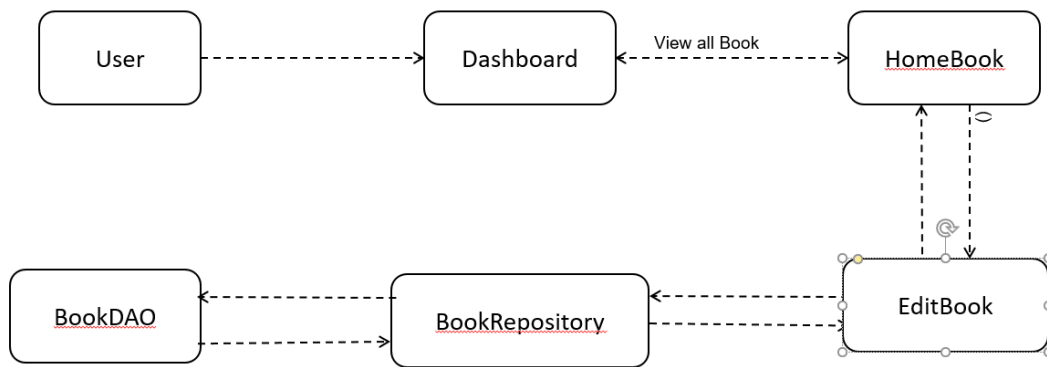
c. Class Specifications

EditBook class

Class Methods

No	Method	Description
01	updateBook	<p>Retrieve the updated book information from the EditText fields, validate the input, and update the book object. Update the book in the database using the Book Repository. Display a success message and navigate to the HomeBook activity upon successful update. Display an error message if the update fails.</p> <p><i>Input:</i> Book</p> <p><i>Output:</i> None</p>

d. Sequence Diagram(s)



e. Database queries

@Query("SELECT * FROM Book WHERE uuid = :uuid")

@Update(onConflict = OnConflictStrategy.REPLACE)

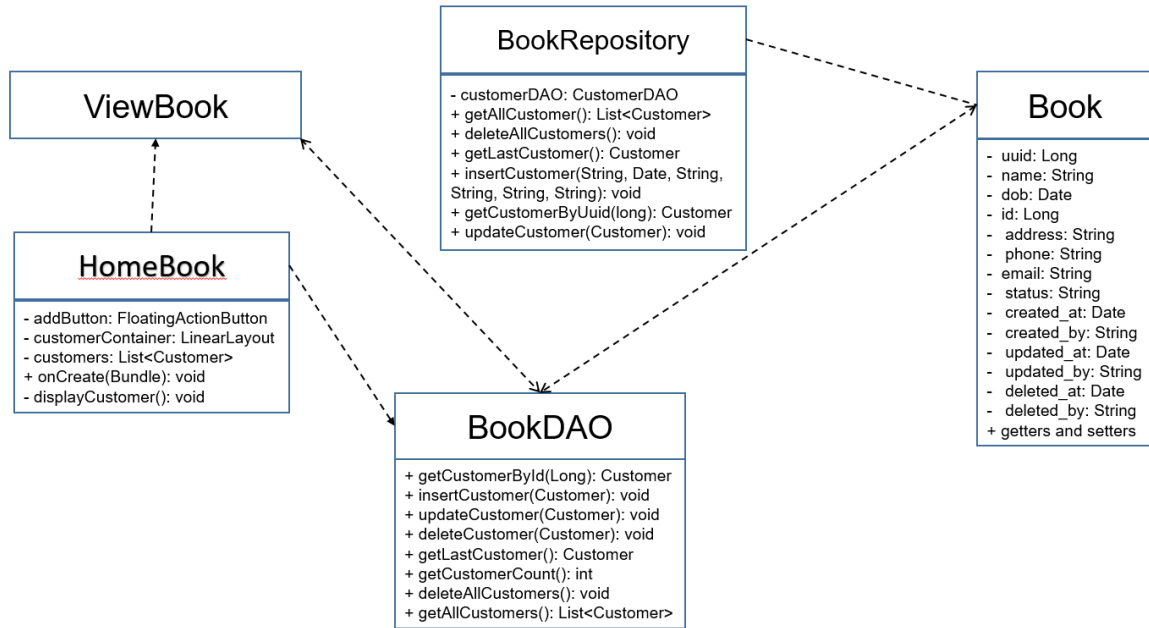
14. View Book

a. Screen Design

Table <x>: <Screen Name> Definition

No	Object/Control Name	Type	Required	Length	Description
1	Header	ActionBar	Yes		Display back and label
2	Information field	TextView	Yes		Information fields as name, status, ...

b. Class Diagram



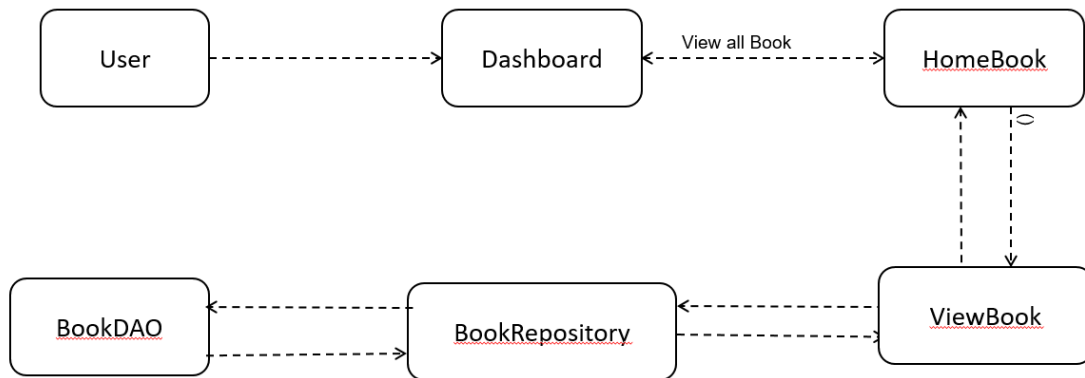
c. Class Specifications

ViewBook class

Class Methods

No	Method	Description
01	onCreate	Initialize the activity, set up the layout, and bind the UI components. Retrieve the book 's information using the UUID passed from the intent and populate the TextView fields with the book 's data.

d. Sequence Diagram(s)



e. Database queries

@Query("SELECT * FROM Book WHERE uuid = :uuid")

15. Employee Profile Detail

a. Screen Design

Table <x>: <Screen Name> Definition

No	Object/Control Name	Type	Required	Length	Description
1	Image View Avatar	ImageView	Yes		Show image
2	Username	Textview	Yes	100	Show user name
3	Date of birth	Textview	Yes	20	Show Date of birth
4	Email	Textview	Yes	20	Show Email
5	Phone	Textview	Yes	20	Show Phone Number
6	Edit Profile	Button	No		Move to edit profile

b. Class Diagram

[This part presents the class diagram for the relevant feature]

c. Class Specifications

EmployeeRepository class

No	Method	Description
01	<code>EmployeeRepository</code>	<i>Initial repository:</i> <ul style="list-style-type: none">- <i>Input: context</i>- <i>Output:</i>
02	<code>getEmployeeById(long id)</code>	<i>Get employee by ID</i> <ul style="list-style-type: none">- <i>Input: id</i>- <i>Output: Employee</i>

Employee Class

Class Methods

No	Method	Description
01	<code>Employee()</code>	<i>Initial an employee object:</i> <ul style="list-style-type: none">- <i>Input: no</i>- <i>Output: object</i>
02	<code>getName()</code>	<i>Get the Employee Name:</i> <ul style="list-style-type: none">- <i>Input: no</i>- <i>Output: name</i>

03	<i>getDob()</i>	<i>Get the Employee Date of Birth:</i> - <i>Input: no</i> - <i>Output: date</i>
04	<i>getEmail()</i>	<i>Get the Employee Email:</i> - <i>Input: no</i> - <i>Output: email</i>
05	<i>getPhone()</i>	<i>Get the Employee Phone Number:</i> - <i>Input: no</i> - <i>Output: phone</i>
06	<i>getAddress()</i>	<i>Get the Employee Address:</i> - <i>Input: no</i> - <i>Output: address</i>

EmployeeProfileActivity Class

Class Methods

[Provide the detailed description for the class methods]

No	Method	Description
01	<code>onCreate()</code>	<i>Build EmployeeProfile activity</i>

d. Sequence Diagram(s)

e. Database queries

@Query("SELECT * FROM Employee WHERE uuid = :uuid")

16. Edit Employee Profile

a. Screen Design

Table <x>: <Screen Name> Definition

No	Object/Control Name	Type	Required	Length	Description
1	Image View Avatar	ImageView	Yes		Show image
2	Change avatar	Button	No		Change avatar
3	Username	EditText	No	100	Edit user name
4	Date of birth	EditText	No	20	Edit Date of birth
5	Email	EditText	No	20	Edit Email
6	Phone	EditText	No	20	Edit Phone Number
7	Save Profile	Button	Yes		Save the profile

b. Class Diagram

[This part presents the class diagram for the relevant feature]

c. Class Specifications

EmployeeRepository class

No	Method	Description
01	EmployeeRepository	<i>Initial repository:</i> - <i>Input: context</i> - <i>Output:</i>

02	<i>getEmployeeById(long id)</i>	<i>Get employee by ID:</i> - <i>Input: id</i> - <i>Output: Employee</i>
03	<i>updateEmployee(Employee employee)</i>	<i>Update Employee:</i> - <i>Input: Employee</i> - <i>Output: no</i>

Employee Class

Class Methods

No	Method	Description
01	<i>Employee()</i>	<i>Initial an employee object:</i> - <i>Input: no</i> - <i>Output: object</i>
02	<i>setName()</i>	<i>Set the Employee Name:</i> - <i>Input: name</i> - <i>Output: no</i>
03	<i>setDob()</i>	<i>Set the Employee Date of Birth:</i> - <i>Input: dob</i> - <i>Output: no</i>

04	<i>setEmail()</i>	<i>Set the Employee Email:</i> - <i>Input: email</i> - <i>Output: no</i>
05	<i>setPhone()</i>	<i>Set the Employee Phone Number:</i> - <i>Input: phone</i> - <i>Output: no</i>
06	<i>setAddress()</i>	<i>Set the Employee Address:</i> - <i>Input: address</i> - <i>Output: no</i>

EmployeeProfileActivity Class

Class Methods

[Provide the detailed description for the class methods]

No	Method	Description
01	<i>onCreate()</i>	<i>Build Edit Profile activity</i>

d. Sequence Diagram(s)

e. Database queries

@Query("SELECT * FROM Employee WHERE uuid = :uuid")

@Update(onConflict = OnConflictStrategy.REPLACE)

17. Edit Employee Profile

a. Screen Design

Table <x>: <Screen Name> Definition

No	Object/Control Name	Type	Required	Length	Description
1	ID	EditText	Yes	100	Input ID
2	Name	EditText	Yes	100	Input Name
3	Address	EditText	Yes	100	Input Address
4	Date of birth	EditText	Yes	20	Input Date of birth
5	Email	EditText	Yes	20	Input Email
6	Phone	EditText	Yes	20	Input Phone Number
7	Staff A	RadioButton	No		Set the role = staffA
8	Staff B	RadioButton	No		Set the role = staffB
9	Manager	RadioButton	No		Set the role = manager

b. Class Diagram

[This part presents the class diagram for the relevant feature]

c. Class Specifications

EmployeeRepository class

No	Method	Description
01	EmployeeRepository	<i>Initial repository:</i> <i>- Input: context</i>

		- <i>Output:</i>
<i>02</i>	<i>addEmployee(Employee employee)</i>	<i>Add new employee to database:</i> - <i>Input: employee</i> - <i>Output: no</i>

Employee Class

Class Methods

No	Method	Description
<i>01</i>	<i>Employee()</i>	<i>Initial an employee object:</i> - <i>Input: no</i> - <i>Output: object</i>
<i>02</i>	<i>setName()</i>	<i>Set the Employee Name:</i> - <i>Input: name</i> - <i>Output: no</i>
<i>03</i>	<i>setDob()</i>	<i>Set the Employee Date of Birth:</i> - <i>Input: dob</i> - <i>Output: no</i>
<i>04</i>	<i>setEmail()</i>	<i>Set the Employee Email:</i> - <i>Input: email</i> - <i>Output: no</i>

05	<i>setPhone()</i>	<i>Set the Employee Phone Number:</i> - <i>Input: phone</i> - <i>Output: no</i>
06	<i>setAddress()</i>	<i>Set the Employee Address:</i> - <i>Input: address</i> - <i>Output: no</i>

Create New Staff Class

Class Methods

[Provide the detailed description for the class methods]

No	Method	Description
01	<i>onCreate()</i>	<i>Build Create New Staff activity</i>

d. Sequence Diagram(s)

e. Database queries

@Insert(onConflict = OnConflictStrategy.REPLACE)

18. Employee Profile Detail

a. Screen Design

Table <x>: <Screen Name> Definition

No	Object/Control Name	Type	Required	Length	Description
----	---------------------	------	----------	--------	-------------

1	Image View Avatar	ImageView	Yes		Show image
2	Username	Textview	Yes	100	Show user name
3	Role	Textview	Yes	20	Show Role
4	Email	Textview	Yes	20	Show Email
5	Phone	Textview	Yes	20	Show Phone Number
6	Edit Staff	Button	No		Move to edit staff

b. Class Diagram

[This part presents the class diagram for the relevant feature]

c. Class Specifications

EmployeeRepository class

No	Method	Description
01	<code>EmployeeRepository</code>	<i>Initial repository:</i> - <i>Input: context</i> - <i>Output:</i>
02	<code>getEmployeeById(long id)</code>	<i>Get employee by ID</i> - <i>Input: id</i> - <i>Output: Employee</i>

Employee Class

Class Methods

No	Method	Description
01	<i>Employee()</i>	<i>Initial an employee object:</i> <ul style="list-style-type: none"> - <i>Input: no</i> - <i>Output: object</i>
02	<i>getName()</i>	<i>Get the Employee Name:</i> <ul style="list-style-type: none"> - <i>Input: no</i> - <i>Output: name</i>
03	<i>getDob()</i>	<i>Get the Employee Date of Birth:</i> <ul style="list-style-type: none"> - <i>Input: no</i> - <i>Output: date</i>
04	<i>getEmail()</i>	<i>Get the Employee Email:</i> <ul style="list-style-type: none"> - <i>Input: no</i> - <i>Output: email</i>
05	<i>getPhone()</i>	<i>Get the Employee Phone Number:</i> <ul style="list-style-type: none"> - <i>Input: no</i> - <i>Output: phone</i>
06	<i>getAddress()</i>	<i>Get the Employee Address:</i> <ul style="list-style-type: none"> - <i>Input: no</i> - <i>Output: address</i>

EmployeeProfileActivity Class

Class Methods

[Provide the detailed description for the class methods]

No	Method	Description
01	<code>onCreate()</code>	<i>Build Staff Profile activity</i>

d. Sequence Diagram(s)

e. Database queries

@Query("SELECT * FROM Employee WHERE uuid = :uuid")

19. Staff List

a. Screen Design

Table <x>: <Screen Name> Definition

No	Object/Control Name	Type	Required	Length	Description
1	Create New Staff	Button	Yes		Create New Staff
2	Staff List	RecyclerView	Yes		Show Staff List

b. Class Diagram

[This part presents the class diagram for the relevant feature]

c. Class Specifications

EmployeeRepository class

No	Method	Description
01	<code>EmployeeRepository</code>	<i>Initial repository:</i> - <i>Input: context</i> - <i>Output:</i>

02	<i>getAll()</i>	<i>Get all employee</i> - <i>Input: no</i> - <i>Output: List<Employee></i>
----	-----------------	--

StaffListAdapter Class

Class Methods

No	Method	Description
01	<i>StaffListAdapter(List<Staff Show> staffShowList, Context context)</i>	<i>Initial an staff list adapter object:</i> - <i>Input: staffShowList, context</i> - <i>Output: staff list adapter</i>

EmployeeProfileActivity Class

Class Methods

[Provide the detailed description for the class methods]

No	Method	Description
01	<i>onCreate()</i>	<i>Build Staff List activity</i>

d. Sequence Diagram(s)

e. Database queries

@Query("SELECT * FROM Employee")

