CMPS 350 Project Phase 2 – UI Design Implementation (10% of the course grade)

Group Id:	t6-ahmed-hani-syed
Group Members:	Ahmed Abdelhamid (201907476) Hani Jassin Jafer (201908619) Syed Zubair (201708606) **Emails:** aa1907476@student.qu.edu.qa; hj1908619@student.qu.edu.qa; sz1708606@student.qu.edu.qa;

Grading Rubric - In the Functionality column please specify either: *Working (completed x%)*, *Not Working (completed x%)* or *Not done*.

Criteria	%	Functionality*	Quality of the implementation	Your Grade
1) Design and implementation of the database schema to manage the data in a MongoDB database (including a schema diagram in the project report).	10%	Working (completed 100%)		
2) Populate the database with the data from the json files.	5%	Working (completed 100%)		
3) Implementation: - Repository Implementation to read/write data from/to MongoDB (30%) - Web API implementation (25%) - Change Web UI to use the Web API (20%)	75%	Working (completed 100%)		
U1 – Login	5	Working (completed 100%)		
U2 - YalaPay Dashboard	10	Working (completed 100%)		
U3 - List/Search/Add/ Update and Delete a Customer: 1. List Customers (5 pts) 2. Search Customers (5 pts) 3. Add Customer (5 pts) 4. Update Customer (5 pts) 5. Delete Customer (5 pts)	25	Working (completed 100%)		
U4 - List/Search/Add/ Update and Delete an Invoice 1. List Invoices (5 pts) 2. Search Invoices (5 pts)	25	Working (completed 100%)		

3. Add Invoice (5 pts) 4. Update Invoice (5 pts) 5. Delete Invoice (5 pts)				
U5 - List/Search/Add/ Update and Delete payments for an invoice 1. List Payments (5 pts) 2. Search Payments (5 pts) 3. Add Payment (5 pts) 4. Update Payment (5 pts) 5. Delete Payment (5 pts)	25	Working (completed 100%)		
U6 - Manage Cashing Cheques 1. Add Cheques Deposit (10 pts) 2. List Cheque Deposits (5 pts) 3. Update Cheques Deposit (15 pts)	30	Working (completed 100%)		
U7 - Invoices Report	10	Working (completed 100%)		
U8 - Cheques Report	10	Working (completed 100%)		
4) Design and Testing Documentation *** Design documentation:** - Web API services diagram - Database schema diagram *** Testing documentation:** with evidence of working implementation using snapshots illustrating the results of your solution testing. *** Discussion of the project contribution** of each team member [-10pts if not done]	10%	Working (completed 100%)		
Total	100			
Copying and/or plagiarism or not being able to explain or answer questions about the implementation	- 100			

[*] **Possible grading for functionality** - *Working* (get 70% of the assigned grade), *Not working* (lose 40% of assigned grade and *Not done* (get 0). The remaining grade is assigned to the quality of the implementation.

In case your implementation is not working then 40% of the grade will be lost and the remaining 60% will be determined based on of the code quality and how close your solution to the working implementation.

Solution quality also includes meaningful naming of identifiers (according to Android naming conventions), no redundant code, simple and efficient design, clean implementation without unnecessary files/code, use of comments where necessary, proper code formatting and indentation.

Marks will be reduced for code duplication, poor/inefficient coding practices, poor naming of identifiers, unclean/untidy submission, and unnecessary complex/poor user interface design.

1. Application Design

- ## Web API services diagram

Deposits

HTTP VERB	URL	FUNCTIONALITY
POST	/api/deposits/	adds new deposit
GET	/api/deposits/	gets the matching deposit or all deposits
PUT	/api/deposits/	Updates matching deposit
DELETE	/api/deposits/	Deletes all deposits
GET	/api/deposits/date/:date	Returns the matching deposits by deposit date
DELETE	/api/deposits/:depositId	deletes the matching deposit
GET	/api/deposits/status/:status	Returns the matching deposit by status

Invoices

HTTP VERB	URL	FUNCTIONALITY
POST	/api/invoices/	adds new invoice
GET	/api/invoices/	gets the matching invoice or all invoices
PUT	/api/invoices/	Updates matching invoice
GET	/api/invoices/customer/ :customerId	Returns the matching invoice holding the customer Id
DELETE	/api/invoices/:invoiceId	deletes the matching invoice
GET	/api/invoices/amount/:amount	Returns the matching invoices by amount
GET	/api/invoices/report	Returns the invoice report
GET	/api/invoices/totalAmount	Returns the total amount of all invoices
GET	/api/invoices/dueDates/:condition	Returns the total amount of all invoices based on their due dates

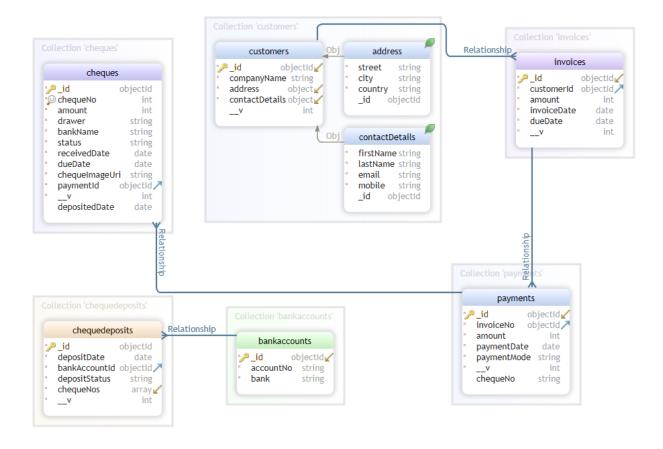| HTTP VERB	URL	FUNCTIONALITY
GET	/api/banks	Returns the bank names
GET	/api/accounts	Return the matching account if id passed, else all accounts if no id passed
GET	api/cheques	Return the matching cheque if id passed, else all cheques if no id passed
POST	api/cheques	Add cheque
PUT	api/cheques	Update cheque
DELETE	api/cheques/:chequeId	Delete cheque by id
GET	api/cheques/awaitingCheques	Return awaiting cheques
GET	api/cheques/payments/:paymentId	Return cheque by payment id
DELETE	api/cheques/payments/:paymentId	Delete cheque by payment id
GET	api/cheques/status/:status	Return total amount of the cheque values which have a specific status
GET	/api/customers	Return matching customer if id passed, else all customers
POST	/api/customers	Add a customer
PUT	/api/customers	Update a customer
DELETE	/api/customers	Delete All customers
DELETE	/api/customers/:customerId	Delete customer with passed Id
GET	/api/customers/companyName/:companyName	Return customer by company name
GET	/api/customers/city/:city	Return customer(s) by city
GET	/api/customers/last/:limit	Return the number of recent customers specified by limit

Payments

HTTP VERB	URL	FUNCTIONALITY
POST	/api/payments/	adds new payment
GET	/api/payments/	gets the matching payment or all payment
PUT	/api/payments/	Updates matching payment
GET	/api/payments/invoices/ :invoiceNo	Returns all the payments that have the matching invoice Id
DELETE	/api/payments/:paymentId	deletes the matching payment
PUT	/api/payments/date/:date	Returns the matching payments by date
GET	/api/payments/amount/:amount	Returns the matching payments by amount
GET	/api/payments/mode/:mode	Returns the matching payments by their payment mode
GET	/api/payments/last/:limit	Returns the number of recent payments specified by the limit

Users

HTTP VERB	URL	FUNCTIONALITY
GET	/api/users/	Returns all the users
GET	/api/users/email/:email	Returns the matching user by email

- **Database schema diagram**

users
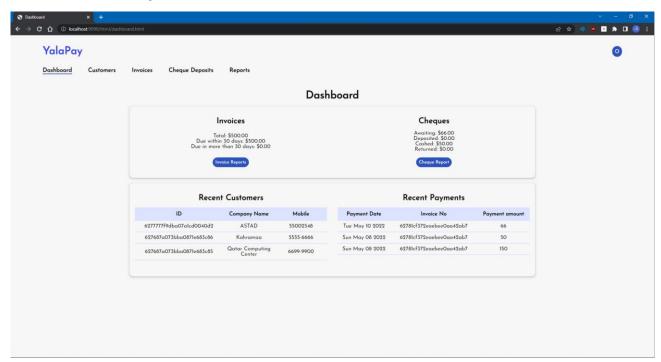|		
⚷	_id	objectId
×	password	string
×	firstName	string
×	email	string
×	lastName	string

Collection 'customers'

customers
⚷	_id	objectId
×	companyName	string
×	address	object
×	contactDetails	object
×	__v	int

Obj

address
street	string
city	string
country	string
_id	objectId

Obj

contactDetails
firstName	string
lastName	string
email	string
mobile	string
_id	objectId

Relationship

Collection 'invoices'

invoices
⚷	_id	objectId
×	customerId	objectId
×	amount	int
×	invoiceDate	date
×	dueDate	date
×	__v	int

Collection 'cheques'

cheques
⚷	_id	objectId
×	chequeNo	int
×	amount	int
×	drawer	string
×	bankName	string
×	status	string
×	receivedDate	date
×	dueDate	date
×	chequeImageUri	string
×	paymentId	objectId
×	__v	int
	depositedDate	date

Relationship

Collection 'payments'

payments
⚷	_id	objectId
×	invoiceNo	objectId
×	amount	int
×	paymentDate	date
×	paymentMode	string
×	__v	int
	chequeNo	string

Relationship

Collection 'chequedeposits'

chequedeposits
⚷	_id	objectId
×	depositDate	date
×	bankAccountId	objectId
×	depositStatus	string
×	chequeNos	array
×	__v	int

Relationship

Collection 'bankaccounts'

bankaccounts
⚷	_id	objectId
×	accountNo	string
×	bank	string

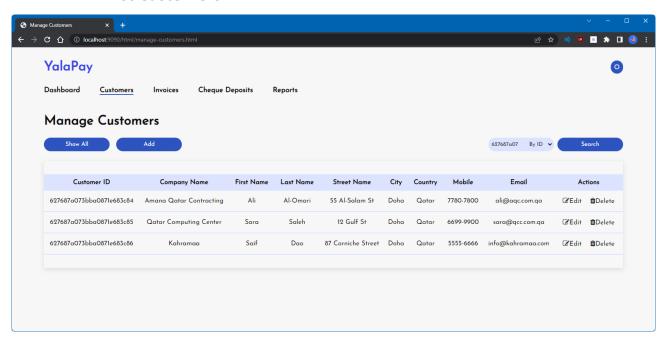## 2. Testing

- ### U1 – Login
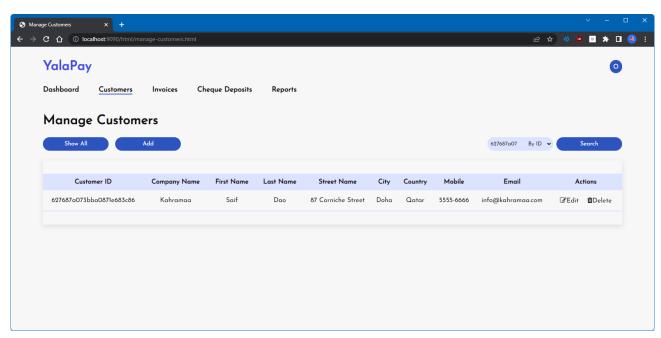

- ### U2 - YalaPay Dashboard

- **U3 - List/Search/Add/ Update and Delete a Customer**

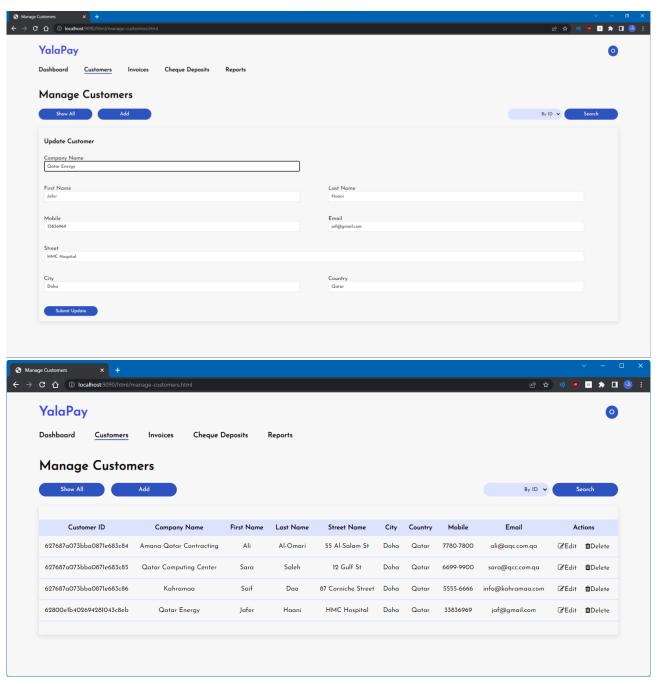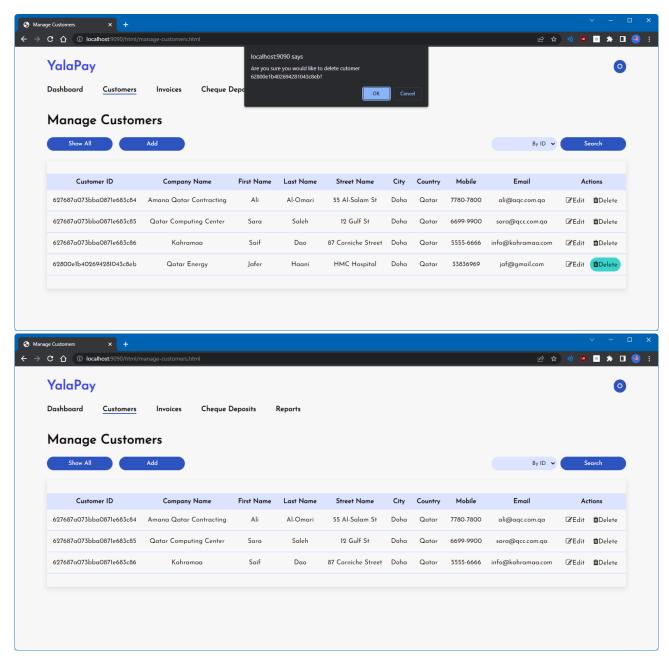## 2..1. List Customers

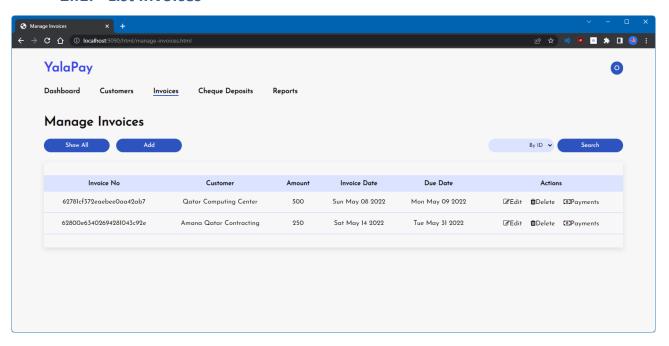## 2..2. Search Customers

2..3. Add Customer

Add Customer

Company Name
QP

First Name
Jafer

Last Name
Haani

Mobile
33836969

Email
jaf@gmail.com

Street
HMC Hospital

City
Doha

Country
Qatar

Submit

YalaPay

Dashboard   **Customers**   Invoices   Cheque Deposits   Reports

Manage Customers

Show All   Add

By ID   Search

Customer ID	Company Name	First Name	Last Name	Street Name	City	Country	Mobile	Email	Actions
627687a073bba0871e683c84	Amana Qatar Contracting	Ali	Al-Omari	55 Al-Salam St	Doha	Qatar	7780-7800	ali@aqc.com.qa	✎Edit 🗑Delete
627687a073bba0871e683c85	Qatar Computing Center	Sara	Saleh	12 Gulf St	Doha	Qatar	6699-9900	sara@qcc.com.qa	✎Edit 🗑Delete
627687a073bba0871e683c86	Kahramaa	Saif	Dao	87 Corniche Street	Doha	Qatar	5555-6666	info@kahramaa.com	✎Edit 🗑Delete
62800e1b402694281043c8eb	QP	Jafer	Haani	HMC Hospital	Doha	Qatar	33836969	jaf@gmail.com	✎Edit 🗑Delete

2..4. Update Customer

2..5.   Delete Customer

- **U4 - List/Search/Add/ Update and Delete an Invoice**

2..1. List Invoices

2..2. Search Invoices

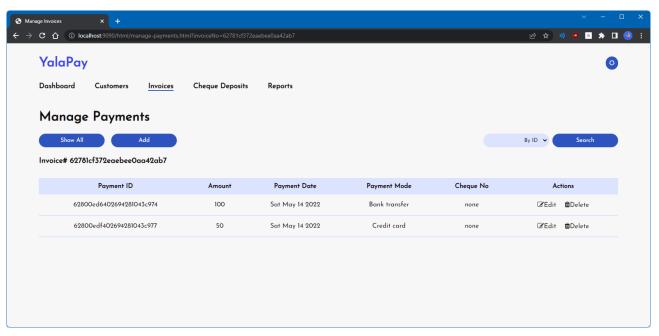## 2..3. Add Invoice

2..4. Update Invoice
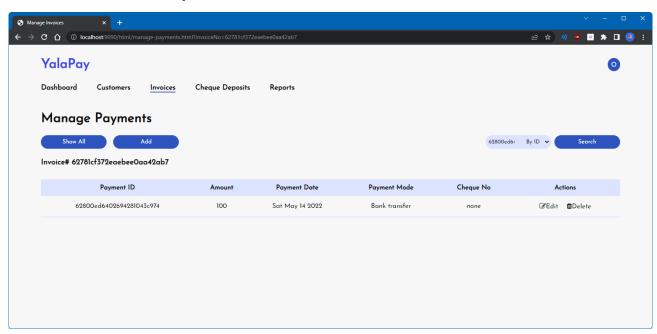
2..5.   Delete Invoice

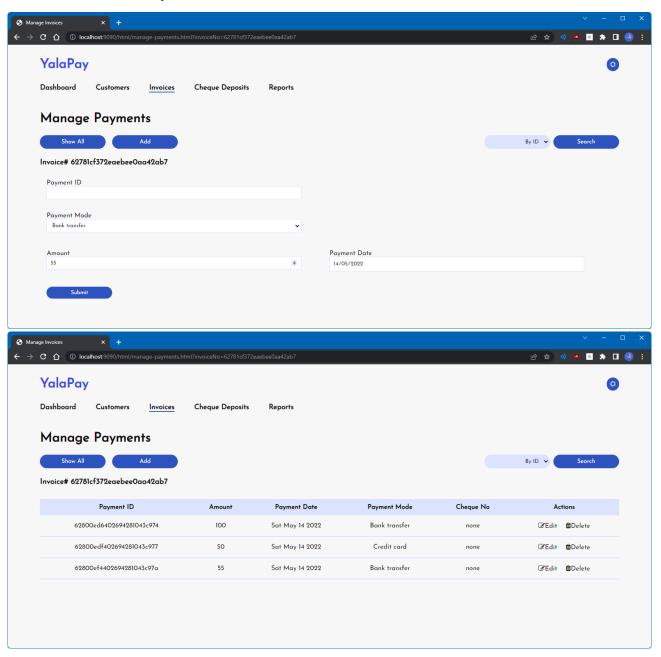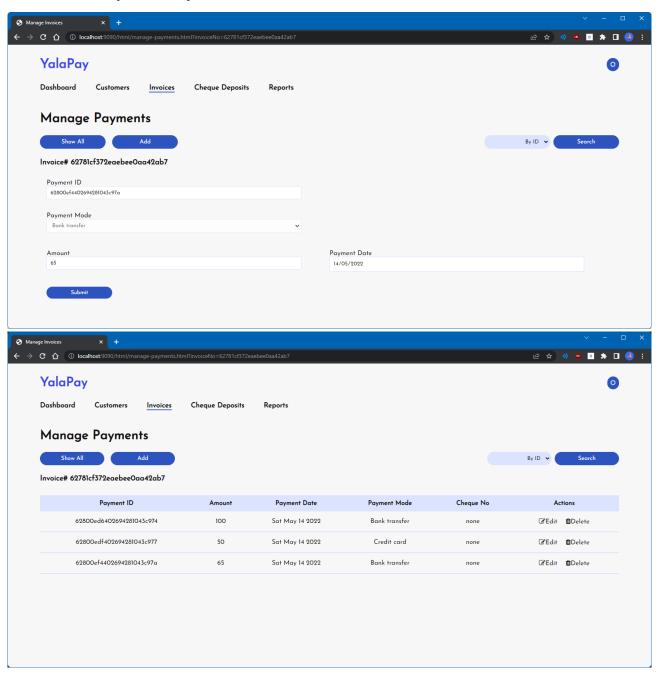- **U5 - List/Search/Add/ Update and Delete payments for an invoice**
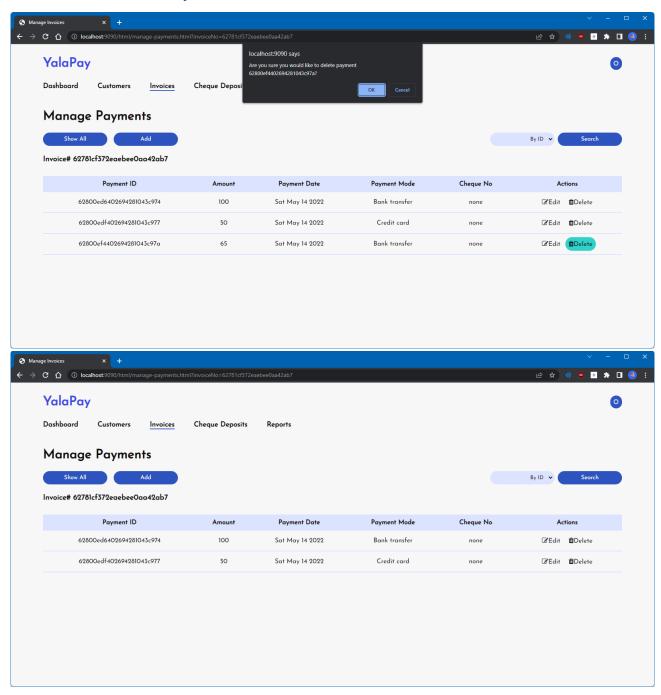
2..1. List Payments

2..2. Search Payments
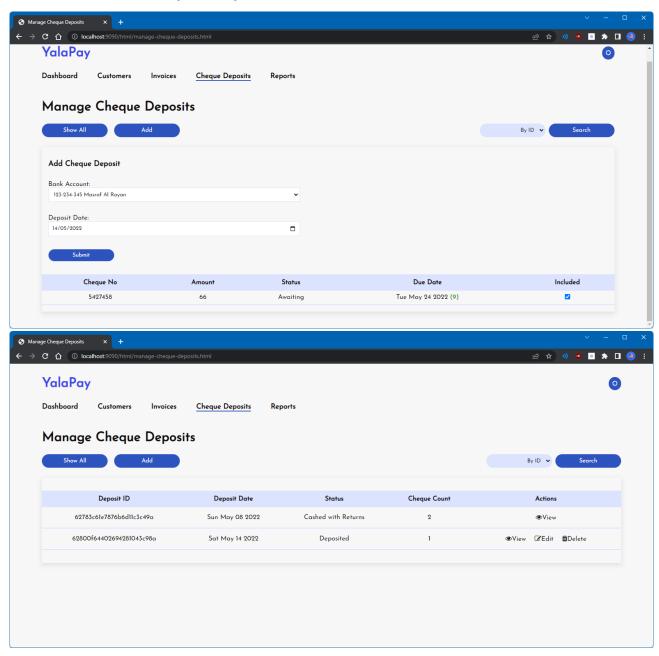
2..3. Add Payment

2..4. Update Payment

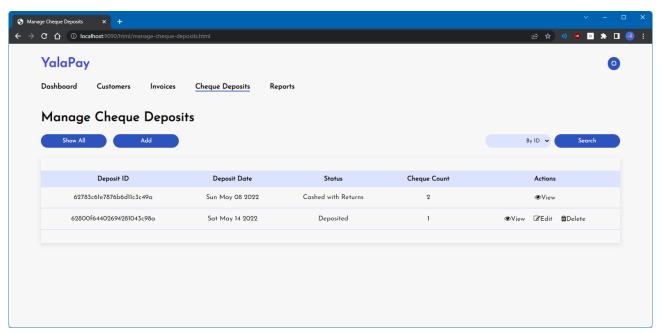## 2..5. Delete Payment
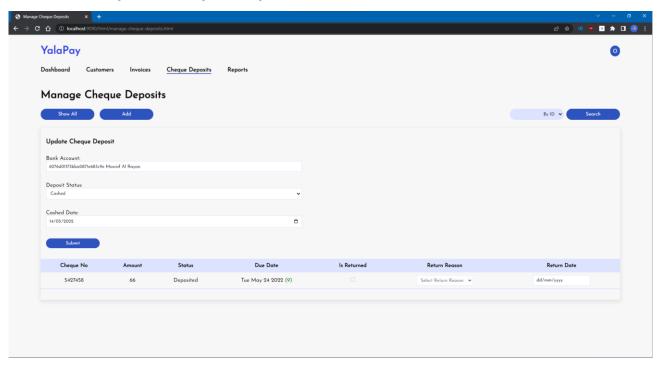
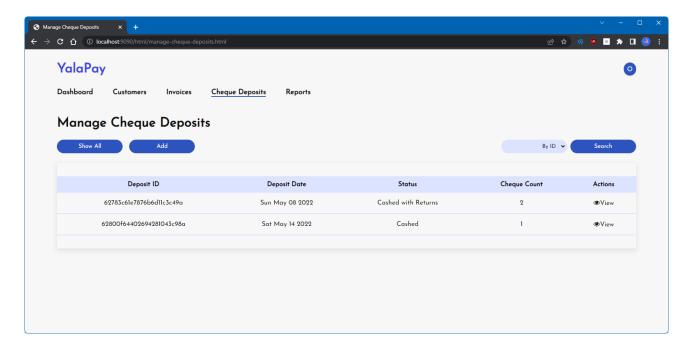- **U6 - Manage Cashing Cheques**

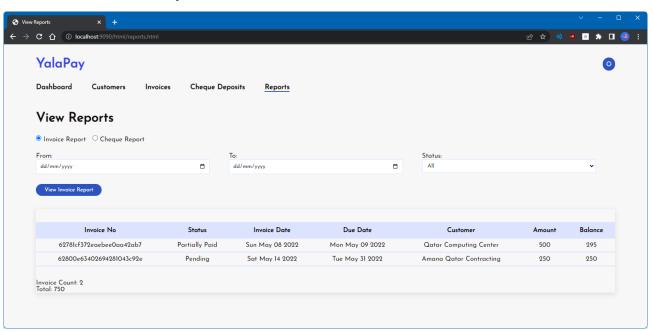## 2..1.   Add Cheques Deposit

2..2. List Cheque Deposits
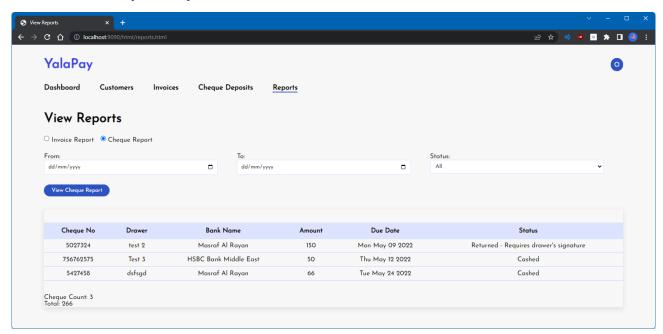

2..3. Update Cheques Deposit

• U7 - Invoices Report

YalaPay

Dashboard   Customers   Invoices   Cheque Deposits   **Reports**

View Reports

● Invoice Report ○ Cheque Report

From:	To:	Status:
dd/mm/yyyy	dd/mm/yyyy	All

View Invoice Report

Invoice No	Status	Invoice Date	Due Date	Customer	Amount	Balance
62781cf372eaebee0aa42ab7	Partially Paid	Sun May 08 2022	Mon May 09 2022	Qatar Computing Center	500	295
62800e63402694281043c92e	Pending	Sat May 14 2022	Tue May 31 2022	Amana Qatar Contracting	250	250

Invoice Count: 2
Total: 750

- **U8 - Cheques Report**

3. Summary of team member contributions

- Ahmed Abdelhamid
  Update relevant areas of code to use MongoDB database for Login, Dashboard, Manage Payment

- Hani Jafer
  Update relevant areas of code to use MongoDB database for Manage Invoice, Cheques-Deposits (update), Reports

- Syed Zubair
  Update relevant areas of code to use MongoDB database for Manage Customers, Cheques-Deposits