

Code:

```
paymentbook.py
1 from flask import Flask, render_template, request, flash, redirect, url_for, session
2 from flask_mysqldb import MySQL
3 import MySQLdb.cursors
4 from flask_bcrypt import Bcrypt
5 from flask_mail import Mail, Message
6 import config
7 import re
8
9 app = Flask(__name__)
10
11 #connecting app to database
12 app.config['MYSQL_HOST'] = "127.0.0.1"
13 app.config['MYSQL_USER'] = "root"
14 app.config['MYSQL_PASSWORD'] = "Ibm@1234"
15 app.config['MYSQL_DB'] = "rocky"
16 app.config['MAIL_SERVER'] = 'smtp.gmail.com'
17 app.config['MAIL_PORT'] = 465
18 app.config['MAIL_USERNAME'] = 'paymentbook.ad@gmail.com'
19 app.config['MAIL_PASSWORD'] = '89199322918'
20 app.config['MAIL_USE_SSL'] = True
21 app.config['MAIL_USE_TLS'] = False
22
23 mysql = MySQL(app)
24 app.secret_key = 'x'
25
26 #password hashing
27 bcrypt = Bcrypt(app)
28
29 #for sending email
30 mail = Mail(app)
```

```
paymentbook.py
34 @app.route('/')
35 def home():
36     if 'loggedin' in session:
37         if session['is_retailer'] == 0:
38             cursor = mysql.connection.cursor()
39             cursor.execute("SELECT purchase_id,user_id,item_name,price,amount_paid,purchase_date FROM Purchase WHERE user_id = % s", [session['user_id']])
40             purchase_details = cursor.fetchall()
41             cursor.execute("SELECT COUNT(*) FROM Purchase WHERE user_id = % s", [session['user_id']])
42             purchase_count = cursor.fetchone()[0]
43             cursor.execute("SELECT COUNT(*) FROM Payments WHERE purchase_id IN (SELECT purchase_id FROM Purchase WHERE user_id = % s)", [session['user_id']])
44             payment_count = cursor.fetchone()[0]
45             cursor.execute("SELECT COUNT(*) FROM Purchase WHERE amount_paid<price AND user_id = % s", [session['user_id']])
46             pending_count = cursor.fetchone()[0]
47         else:
48             cursor = mysql.connection.cursor()
49             cursor.execute("SELECT purchase_id,user_id,item_name,price,amount_paid,purchase_date FROM Purchase")
50             purchase_details = cursor.fetchall()
51             cursor.execute("SELECT COUNT(*) FROM Purchase")
52             purchase_count = cursor.fetchone()[0]
53             cursor.execute("SELECT COUNT(*) FROM Payments")
54             payment_count = cursor.fetchone()[0]
55             cursor.execute("SELECT COUNT(*) FROM Purchase WHERE amount_paid<price")
56             pending_count = cursor.fetchone()[0]
57
58         return render_template('home.html', purchase_details=purchase_details, username=session['username'], role=session['is_retailer'], purchase_count=purchase_count, payment_count=payment_count)
59
60     else:
61         return redirect(url_for('intro'))
62
63
64 @app.route('/register', methods=['POST','GET'])
```

The screenshot shows the PyCharm IDE interface with the following details:

- File Path:** C:\Users\RAKESH CHANDU\Downloads\payment book app\paymentbook.py
- Code Content:** The code is a Python script named `paymentbook.py`. It contains a `register()` function that handles user registration. It uses MySQL for database operations and bcrypt for password hashing. It also sends an email confirmation message using the `Message` class from the `smtplib` module.
- Toolbars and Status Bar:** The top bar shows standard file menu options like File, Edit, View, Navigate, Code, Refactor, Run, Tools, VCS, Window, Help, and a Python path indicator. The bottom status bar shows "Breakpoint reached (33 minutes ago)" and "8999 CRLF UTF-8 4 spaces Python 3.10 (2)".

The screenshot shows the PyCharm IDE interface with the following details:

- File Path:** C:\Users\RAKESH CHANDU\Downloads\payment book app\paymentbook.py
- Code Content:** The code is a Python script using Flask. It defines a login route that checks if a user exists and their password is correct. If successful, it logs the user in (storing user ID, name, retailer status, and email in session) and redirects to the home page. If unsuccessful, it renders the login.html template with an error message.

```
File Edit View Navigate Code Refactor Run Tools VCS Window Help Python - C:\Users\RAKESH CHANDU\Downloads\payment book app\paymentbook.py
C:\Users\RAKESH CHANDU Downloads\payment book app\paymentbook.py
Project paymentbook.py
98     mail.send(msg)
99
100    return redirect(url_for('Login'))
101
102    return render_template('register.html', message=message)
103
104 @app.route('/login', methods=['POST', 'GET'])
105 def login():
106     message=None
107     if request.method == 'POST':
108         email = request.form['email']
109         password = request.form['password']
110         cursor = mysql.connection.cursor()
111         cursor.execute('SELECT * FROM Users WHERE email = % s', [email])
112         user = cursor.fetchone()
113         print(user)
114
115         if user and bcrypt.check_password_hash(user[3], password):
116             session['user_id'] = user[0]
117             session['username'] = user[1]
118             session['is_retailer'] = user[4]
119             session['email'] = user[2]
120             session['loggedin'] = True
121             return redirect(url_for('home'))
122         else:
123             message="Log in Unsuccessful. Please check username and password"
124
125
126     return render_template("login.html", message=message)
127
128
129
register() : if request.method == 'POST' : else
Version Control Run Debug Python Packages TODO Python Console Problems Terminal Services
Breakpoint reached (34 minutes ago)
```

```
129
130     @app.route('/logout', methods=['POST', 'GET'])
131     def logout():
132         session.pop('loggedin', None)
133         session.pop('user_id', None)
134         session.pop('username', None)
135         session.pop('is_retailer', None)
136         session.pop('email', None)
137         return redirect(url_for('intro'))
138
139
140     @app.route('/addpurchase', methods=['POST', 'GET'])
141     def addpurchase():
142         if 'loggedin' in session:
143             if session['is_retailer'] == 1:
144                 if request.method == 'POST':
145                     customer = request.form['customer']
146                     item = request.form['item']
147                     price = request.form['price']
148                     purchase_date = request.form['purchase_date']
149                     amount_paid = request.form['amount_paid']
150                     pending_amount = int(price) - int(amount_paid)
151                     cursor = mysql.connection.cursor()
152                     cursor.execute("INSERT INTO Purchase(user_id, item_name, price, purchase_date, amount_paid) VALUES(%s, %s, %s, %s, %s)", [customer, item, price, purchase_date, amount_paid])
153                     mysql.connection.commit()
154                     if pending_amount != 0:
155
156                         cursor.execute("SELECT email, username FROM Users WHERE user_id = %s", [customer])
157                         user_details = cursor.fetchone()
158                         recipient = user_details[0]
159                         username = user_details[1]
160                         msg = Message('Pending Payment Alert!!!', sender=config.email,
161
register() > if request.method == 'POST' > else
162
163             recipients=[recipient])
164             msg.body = f'''
165             Hey {username}, we hope you are doing well.
166             The payment is still pending for the purchase {item} made on {purchase_date}.
167             We have given long due time to you to pay the pending payment.
168             within two days without fail.
169             Pending Payment Details :
170             Item : {item}
171             Price : {price}
172             Purchase Date : {purchase_date}
173             Amount Paid : {amount_paid}
174             Pending Amount : {pending_amount}
175
176             ...
177
178             mail.send(msg)
179             return redirect(url_for('home'))
180
181             cursor = mysql.connection.cursor()
182             cursor.execute("SELECT * FROM Users WHERE is_admin = 0")
183             customers = cursor.fetchall()
184
185             return render_template('addpurchase.html', customers=customers)
186
187             return redirect(url_for('home'))
188         else:
189             return redirect(url_for('intro'))
190
191
192     @app.route('/updatepurchase/<int:id>', methods=['POST', 'GET'])
193     def updatepurchase(id):
register() > if request.method == 'POST' > else
```

```
161             recipients=[recipient])
162             msg.body = f'''
163             Hey {username}, we hope you are doing well.
164             The payment is still pending for the purchase {item} made on {purchase_date}.
165             We have given long due time to you to pay the pending payment.
166             within two days without fail.
167             Pending Payment Details :
168             Item : {item}
169             Price : {price}
170             Purchase Date : {purchase_date}
171             Amount Paid : {amount_paid}
172             Pending Amount : {pending_amount}
173
174             ...
175
176             mail.send(msg)
177             return redirect(url_for('home'))
178
179             cursor = mysql.connection.cursor()
180             cursor.execute("SELECT * FROM Users WHERE is_admin = 0")
181             customers = cursor.fetchall()
182
183             return render_template('addpurchase.html', customers=customers)
184
185             return redirect(url_for('home'))
186         else:
187             return redirect(url_for('intro'))
188
189
190     @app.route('/updatepurchase/<int:id>', methods=['POST', 'GET'])
191     def updatepurchase(id):
register() > if request.method == 'POST' > else
```

paymentbook.py

```
194     if 'loggedin' in session:
195         if session['is_retailer'] == 1:
196
197             if request.method == 'POST':
198
199                 item = request.form['item']
200                 price = request.form['price']
201                 purchase_date = request.form['purchase_date']
202                 amount_paid = request.form['amount_paid']
203                 cursor = mysql.connection.cursor()
204                 cursor.execute("UPDATE Purchase SET item_name = %s, price = %s, amount_paid = %s, purchase_date = %s WHERE purchase_id = %s", [item, price, amount_paid, purchase_date, id])
205                 mysql.connection.commit()
206
207                 return redirect(url_for('home'))
208
209
210             cursor = mysql.connection.cursor()
211             cursor.execute("SELECT * FROM Purchase WHERE purchase_id = %s", [id])
212             purchase = cursor.fetchone()
213
214             return render_template('updatepurchase.html', purchase=purchase)
215         else:
216             return redirect(url_for('intro'))
217
218
219     @app.route('/deletepurchase/<int:id>', methods=['POST', 'GET'])
220     def deletepurchase(id):
221         if 'loggedin' in session:
222             if session['is_retailer'] == 1:
223                 cursor = mysql.connection.cursor()
224                 cursor.execute("DELETE FROM Purchase WHERE purchase_id = %s", [id])
225                 cursor.execute("DELETE FROM Payments WHERE purchase_id = %s", [id])
226
227                 mysql.connection.commit()
228                 return redirect(url_for('home'))
229             else:
230                 return redirect(url_for('intro'))
231
232
233     @app.route('/displaypurchase')
234     def displaypurchase():
235         if 'loggedin' in session:
236             if session['is_retailer'] == 0:
237                 cursor = mysql.connection.cursor()
238                 cursor.execute("SELECT purchase_id, user_id, item_name, price, amount_paid, purchase_date FROM Purchase WHERE user_id = %s", [session['user_id']])
239                 purchase_details = cursor.fetchall()
240                 print(purchase_details)
241             else:
242                 cursor = mysql.connection.cursor()
243                 cursor.execute("SELECT purchase_id, user_id, item_name, price, amount_paid, purchase_date FROM Purchase")
244                 purchase_details = cursor.fetchall()
245                 print(purchase_details)
246
247
248             return render_template('displaypurchase.html', purchase_details=purchase_details, username=session['username'])
249
250         else:
251             return redirect(url_for('intro'))
252
253
254     @app.route('/addpayment', methods=['POST', 'GET'])
255     def addpayment():
256         if 'loggedin' in session:
```

paymentbook.py

```
226             mysql.connection.commit()
227             return redirect(url_for('home'))
228         else:
229             return redirect(url_for('intro'))
230
231
232
233     @app.route('/displaypurchase')
234     def displaypurchase():
235         if 'loggedin' in session:
236             if session['is_retailer'] == 0:
237                 cursor = mysql.connection.cursor()
238                 cursor.execute("SELECT purchase_id, user_id, item_name, price, amount_paid, purchase_date FROM Purchase WHERE user_id = %s", [session['user_id']])
239                 purchase_details = cursor.fetchall()
240                 print(purchase_details)
241             else:
242                 cursor = mysql.connection.cursor()
243                 cursor.execute("SELECT purchase_id, user_id, item_name, price, amount_paid, purchase_date FROM Purchase")
244                 purchase_details = cursor.fetchall()
245                 print(purchase_details)
246
247
248             return render_template('displaypurchase.html', purchase_details=purchase_details, username=session['username'])
249
250         else:
251             return redirect(url_for('intro'))
252
253
254
255     @app.route('/addpayment', methods=['POST', 'GET'])
256     def addpayment():
257         if 'loggedin' in session:
```

A screenshot of a Python code editor (Rocky) showing a file named `paymentbook.py`. The code handles a POST request to process a payment. It retrieves purchase ID, amount paid, and payment date from the form. It then inserts the data into a MySQL database table `Payments` and updates the `Purchase` table to reflect the payment. It fetches details of the purchased item and user. An email message is constructed to inform the user about the purchase and pending payments. The message body includes the item name, price, purchase date, total amount paid, and pending amount.

```
258     if session['is_retailer']==1:
259         if request.method == 'POST':
260             purchase_id = request.form['purchase_id']
261             amount_paid = request.form['amount_paid']
262             payment_date = request.form['payment_date']
263             cursor = mysql.connection.cursor()
264             cursor.execute('INSERT INTO Payments(purchase_id, amount_paid, payment_date) VALUES(%s,%s,%s)', [purchase_id,amountpaid,payment_date])
265             cursor.execute('UPDATE Purchase SET amount_paid=amount_paid + %s WHERE purchase_id = %s',[amountpaid,purchase_id])
266             mysql.connection.commit()
267             cursor.execute("SELECT item_name, price, amount_paid, user_id, purchase_date FROM Purchase WHERE purchase_id = %s", [purchase_id])
268             details = cursor.fetchone()
269             print(details)
270             item = details[0]
271             price = details[1]
272             tot_amount_paid = details[2]
273             userid = details[3]
274             purchase_date = details[4]
275             cursor.execute("SELECT email, username FROM Users WHERE user_id = %s", [userid])
276             user = cursor.fetchone()
277             print(user)
278             recipient = user[0]
279             username = user[1]
280             pending_amount = price-tot_amount_paid
281             msg = Message('Payment Detail', sender=config.email,
282                           recipients=[recipient])
283             msg.body = f'''
284                 Hey {username}, we hope you are doing well.
285                 You made a payment on {payment_date} for the purchase item - {item} purchased on {purchase_date}.
286                 Please pay the balance pending payments as soon as possible. If you paid all the payments, then leave it.
287
288                 Payment Details:
289                 Item : {item}
290             '''
291             register()
292             if request.method == 'POST' :
293                 msg.send()
294             else:
295                 return render_template('addpayment.html')
296
297     else:
298         return redirect(url_for('intro'))
299
300     @app.route('/pendingpayments')
301     def pendingpayments():
302         if 'loggedin' in session:
303             if session['is_retailer'] == 0:
304                 cursor = mysql.connection.cursor()
305                 cursor.execute("SELECT purchase_id,user_id,item_name,price,amount_paid,purchase_date FROM Purchase WHERE amount_paid<price AND user_id = %s", [session['user_id']])
306                 pending_payments = cursor.fetchall()
307                 print('Pending Payments : ', pending_payments)
308             else:
309                 cursor = mysql.connection.cursor()
310                 cursor.execute("SELECT purchase_id,user_id,item_name,price,amount_paid,purchase_date FROM Purchase WHERE amount_paid<price")
311                 pending_payments = cursor.fetchall()
312                 print('Pending Payments : ', pending_payments)
313
314             register()
315             if request.method == 'POST' :
316                 msg.send()
317             else:
318                 return render_template('addpayment.html')
319
320     else:
321         return redirect(url_for('intro'))
```

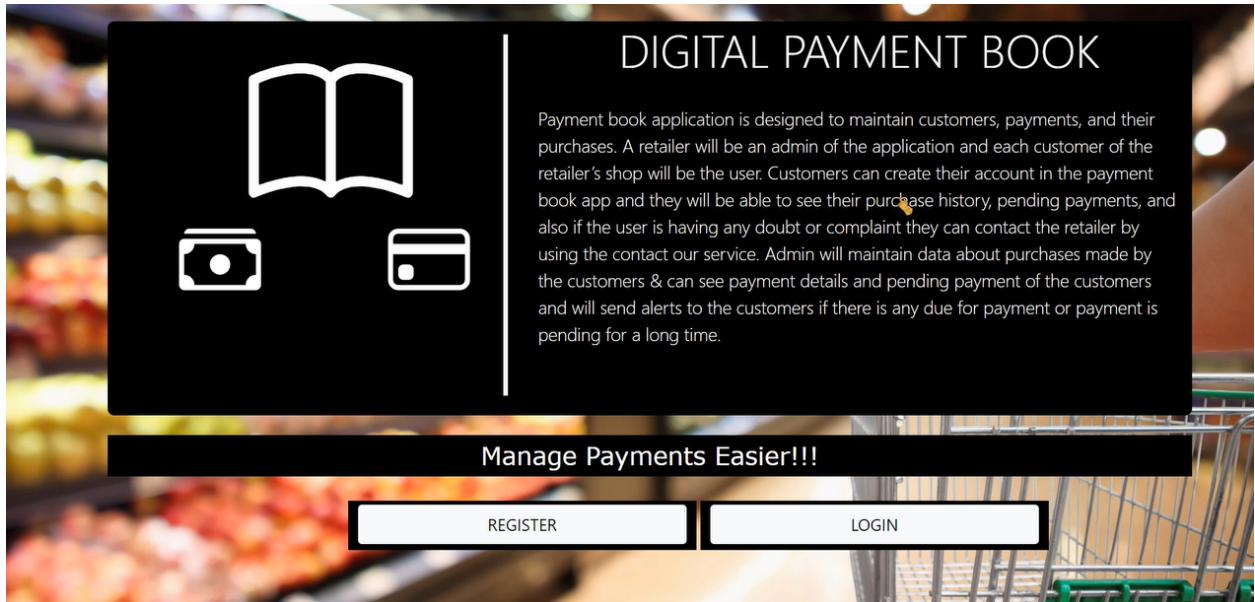
A screenshot of a Python code editor (Rocky) showing the same `paymentbook.py` file as the previous screenshot, but with additional logic added. The new code handles the case where the user is a retailer (session['is_retailer'] == 0). It fetches pending payments for the retailer's user ID and prints the results. The rest of the code remains the same, handling the POST method to process payments and sending emails.

```
298     else:
299         cursor = mysql.connection.cursor()
300         cursor.execute("SELECT purchase_id,user_id,item_name,price,amount_paid,purchase_date FROM Purchase WHERE amount_paid<price AND user_id = %s", [session['user_id']])
301         pending_payments = cursor.fetchall()
302         print('Pending Payments : ', pending_payments)
303
304     register()
305     if request.method == 'POST' :
306         msg.send()
307     else:
308         return render_template('addpayment.html')
309
310     @app.route('/pendingpayments')
311     def pendingpayments():
312         if 'loggedin' in session:
313             if session['is_retailer'] == 0:
314                 cursor = mysql.connection.cursor()
315                 cursor.execute("SELECT purchase_id,user_id,item_name,price,amount_paid,purchase_date FROM Purchase WHERE amount_paid<price AND user_id = %s", [session['user_id']])
316                 pending_payments = cursor.fetchall()
317                 print('Pending Payments : ', pending_payments)
318             else:
319                 cursor = mysql.connection.cursor()
320                 cursor.execute("SELECT purchase_id,user_id,item_name,price,amount_paid,purchase_date FROM Purchase WHERE amount_paid<price")
321                 pending_payments = cursor.fetchall()
322                 print('Pending Payments : ', pending_payments)
323
324             register()
325             if request.method == 'POST' :
326                 msg.send()
327             else:
328                 return render_template('addpayment.html')
329
330     else:
331         return redirect(url_for('intro'))
```

```
323     return render_template('pending_payments.html', pending_payments=pending_payments, username=session['username'], role=session['is_retailer'])
324 else:
325     return redirect(url_for('intro'))
326
327 @app.route('/paymentdetails/')
328 def paymentdetails():
329     if 'loggedin' in session:
330         if session['is_retailer'] == 0:
331             cursor = mysql.connection.cursor()
332             cursor.execute("SELECT * FROM Payments WHERE purchase_id IN (SELECT purchase_id FROM Purchase WHERE user_id = % s)", [session['user_id']])
333             payment_details = cursor.fetchall()
334         else:
335             cursor = mysql.connection.cursor()
336             cursor.execute("SELECT * FROM Payments")
337             payment_details = cursor.fetchall()
338
339     return render_template('paymentdetails.html', payments=payment_details, username=session['username'])
340 else:
341     return redirect(url_for('intro'))
342
343 @app.route('/contact')
344 def contact():
345     if 'loggedin' in session:
346         if session['is_retailer'] == 0:
347             return render_template('contact.html', username=session['username'])
348         else:
349             return redirect(url_for('intro'))
350
351 @app.route('/pendingemail/<int:id>')
352 def pendingemail(id):
353     if 'loggedin' in session:
354         if register() && request.method == 'POST':
355             if id == 1:
356                 pending_email = PendingEmail.query.get(id)
357                 pending_email.status = 'sent'
358                 db.session.commit()
359
360                 mail.send(msg)
361                 return redirect(url_for('home'))
362             else:
363                 redirect(url_for('intro'))
364
365 @app.route('/intro')
366 def intro():
367     return render_template('intro.html')
368
369 @app.route('/terms')
370 def terms():
371     return render_template('terms.html')
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396 if __name__ == '__main__':
397     app.run(port=3006, host="127.0.0.1")
```

```
371     Pending Payment Details :
372         Item : {purchase[0]}
373         Price : {purchase[1]}
374         Purchase Date : {purchase[2]}
375         Amount Paid : {purchase[3]}
376         Pending Amount : {pending_amount}
377
378         ...
379
380         Thankyou.
381
382         mail.send(msg)
383         return redirect(url_for('home'))
384     else:
385         redirect(url_for('intro'))
386
387 @app.route('/intro')
388 def intro():
389     return render_template('intro.html')
390
391 @app.route('/terms')
392 def terms():
393     return render_template('terms.html')
394
395
396 if __name__ == '__main__':
397     app.run(port=3006, host="127.0.0.1")
```

Outputs:



The image shows the home screen of a mobile application titled "DIGITAL PAYMENT BOOK". The background is a blurred photograph of a grocery store aisle with fresh produce. On the left, there is a black rectangular overlay containing three white icons: an open book, a wallet with a dollar sign, and a credit card. To the right of this overlay, the app's title is displayed in large, bold, white capital letters. Below the title is a detailed description of the app's purpose and features. At the bottom of the screen, there is a dark horizontal bar with the text "Manage Payments Easier!!!". Below this bar are two white buttons with black text: "REGISTER" on the left and "LOGIN" on the right.

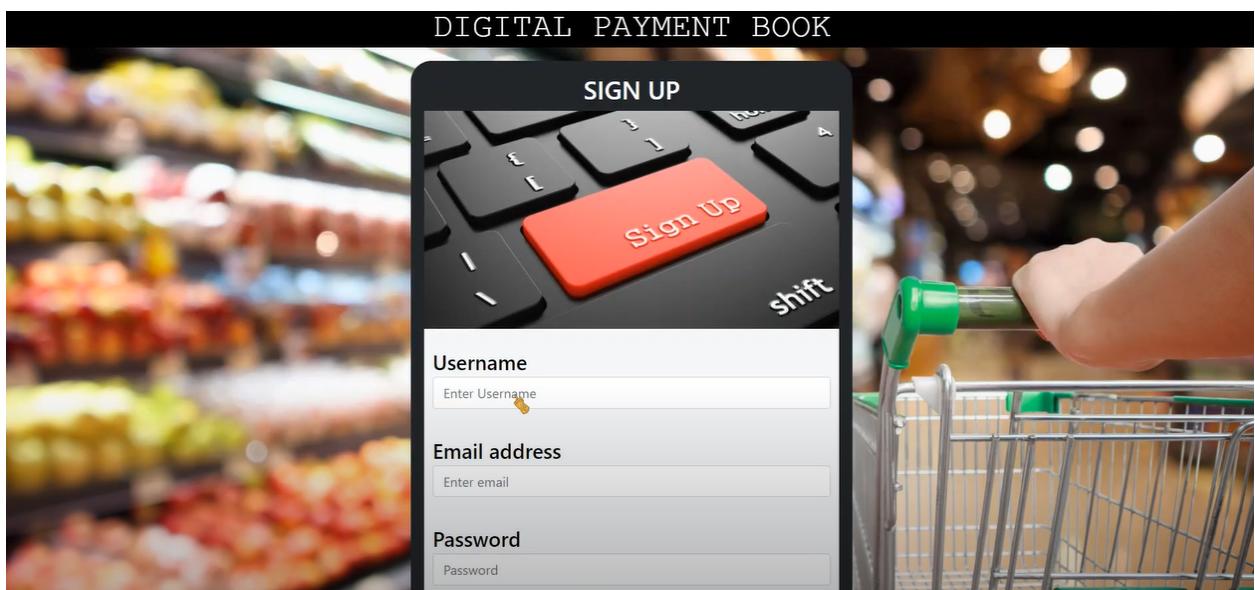
DIGITAL PAYMENT BOOK

Payment book application is designed to maintain customers, payments, and their purchases. A retailer will be an admin of the application and each customer of the retailer's shop will be the user. Customers can create their account in the payment book app and they will be able to see their purchase history, pending payments, and also if the user is having any doubt or complaint they can contact the retailer by using the contact our service. Admin will maintain data about purchases made by the customers & can see payment details and pending payment of the customers and will send alerts to the customers if there is any due for payment or payment is pending for a long time.

Manage Payments Easier!!!

REGISTER

LOGIN



The image shows the "SIGN UP" screen of the "DIGITAL PAYMENT BOOK" application. The background is a blurred photograph of a grocery store aisle. The central focus is a virtual keyboard with a red "Sign Up" button. Below the keyboard, there are three input fields with labels: "Username", "Email address", and "Password". Each field has a placeholder text and a small orange cursor icon indicating interactivity. The overall design is clean and modern, with a black header bar at the top.

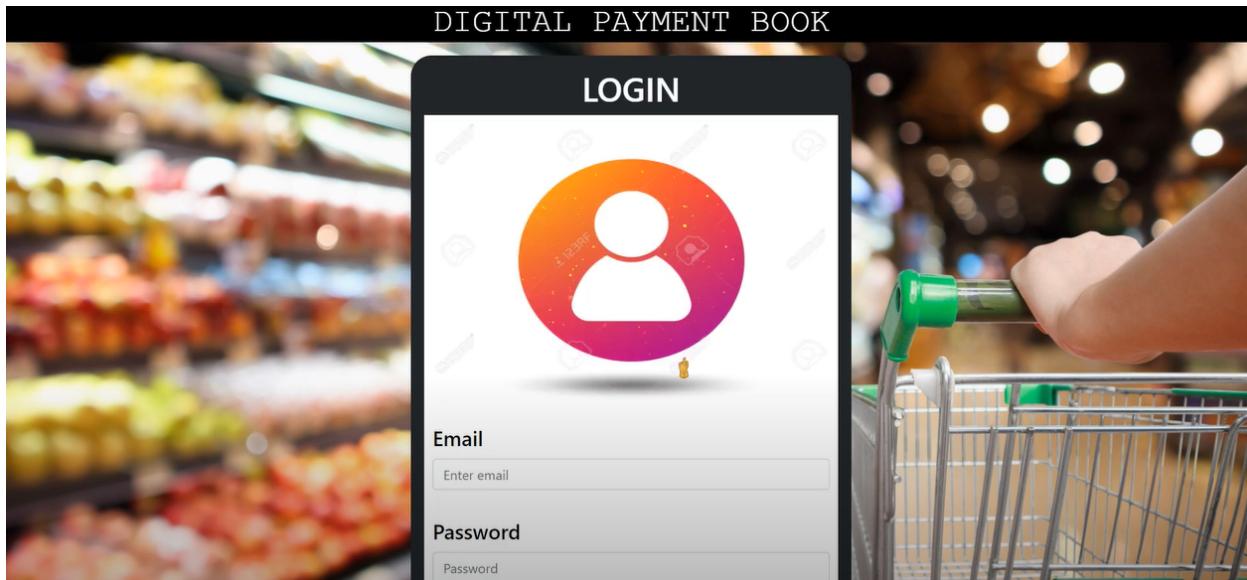
DIGITAL PAYMENT BOOK

SIGN UP

Username
Enter Username

Email address
Enter email

Password
Password



PURCHASE HISTORY

Purchase Id	User ID	Item Name	Price	Amount Paid	Purchase Date
-------------	---------	-----------	-------	-------------	---------------

The grid displays three items:

- PENDING:** Shows a large red "PENDING" stamp. Below it is text: "See Pending Payments" and "Click the button below to see pending payments details." A blue "See Pending Payments" button is at the bottom.
- Payment Details:** Shows a hand holding a credit card over a laptop keyboard. Below it is text: "See Payment Details" and "Click the below button to see payment details." A blue "See Payment Details" button is at the bottom.
- Contact Us:** Shows two speech bubbles, one blue and one red, containing the words "CONTACT" and "US". Below it is text: "Contact Our Service" and "Click the below button to contact us." A blue "Contact Us" button is at the bottom.

DIGITAL PAYMENT BOOK

Welcome Admin!!!

Logout

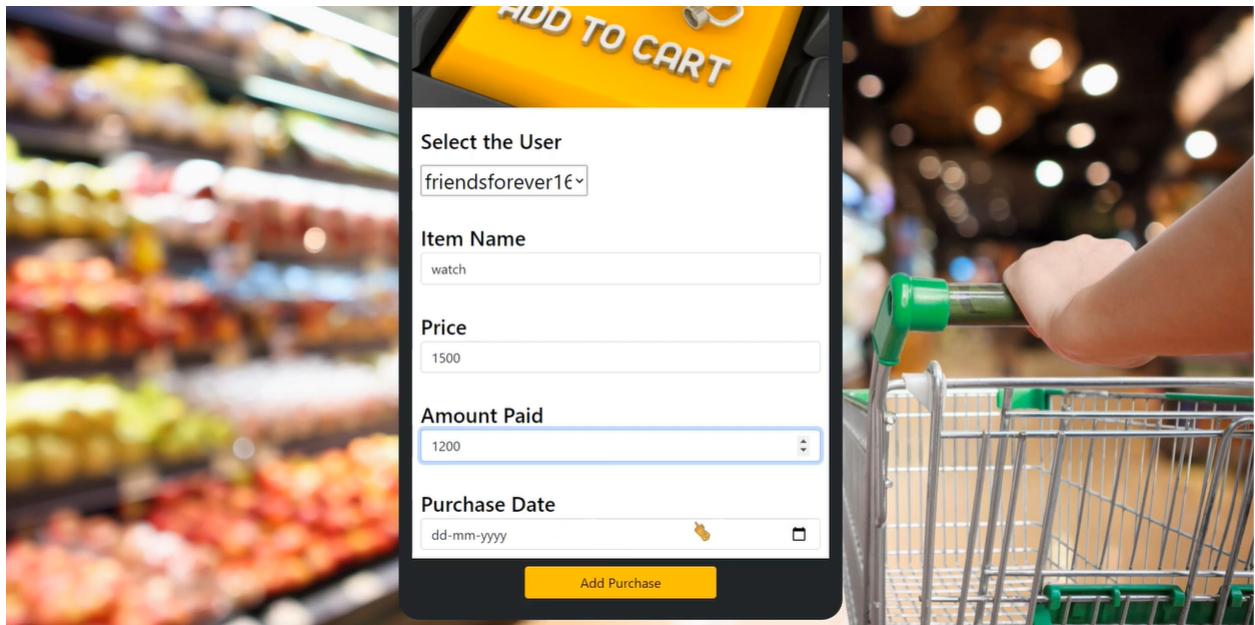
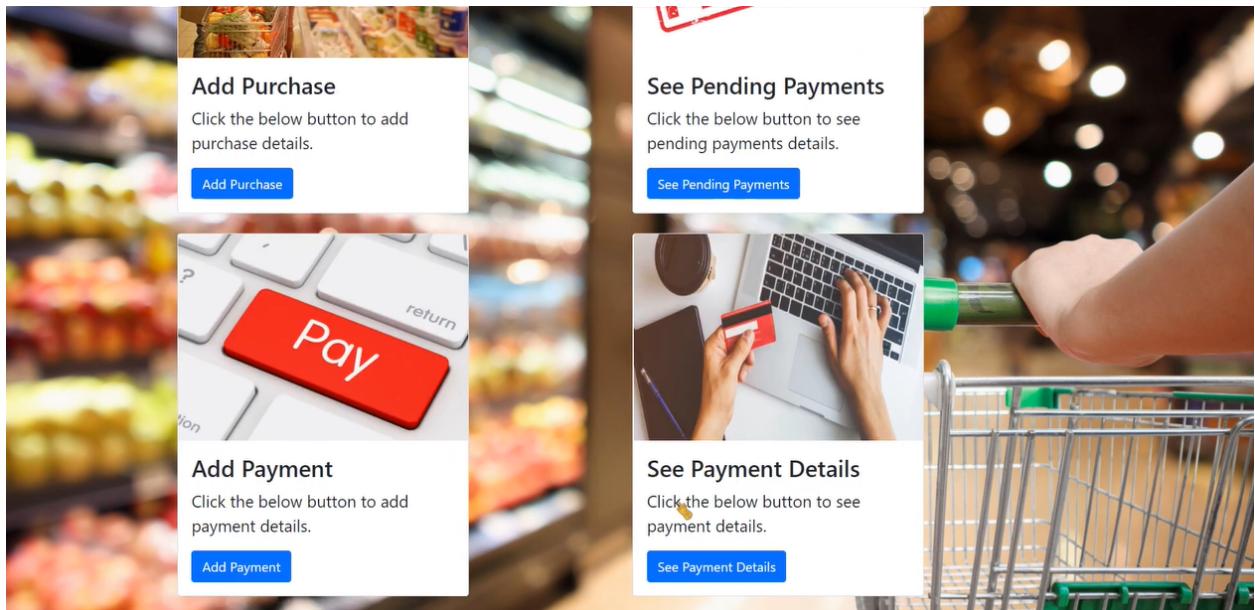
TOTAL PURCHASES
11

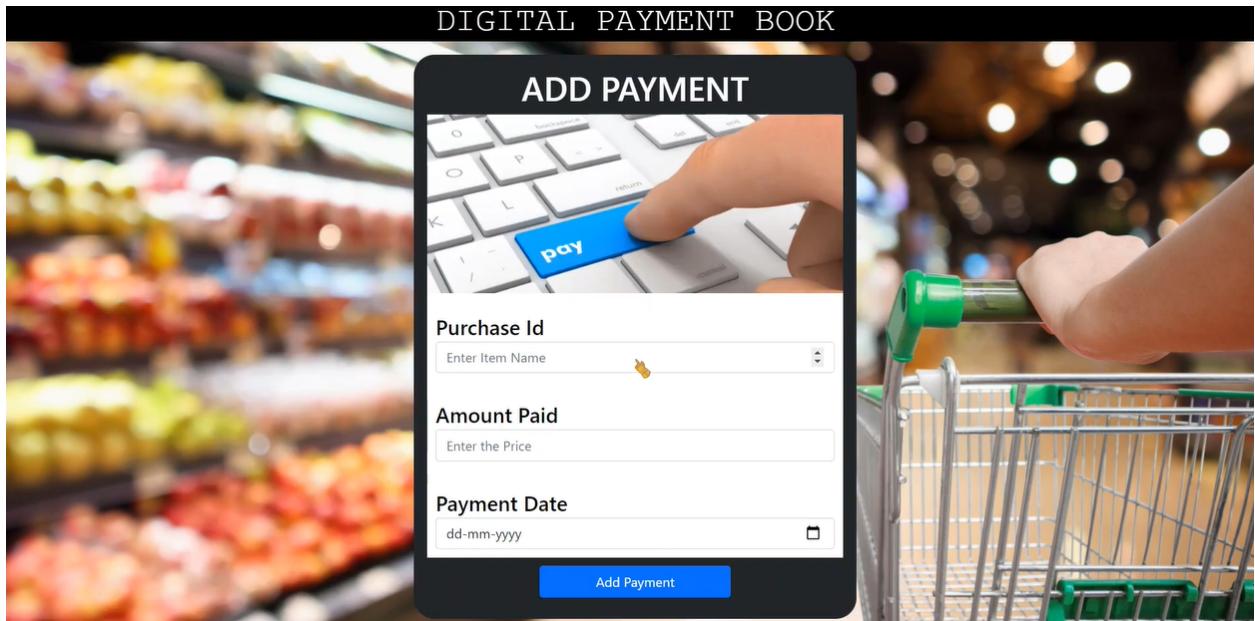
TOTAL PAYMENTS
10

TOTAL PENDING PAYMENTS
9

PURCHASE HISTORY

Purchase Id	User ID	Item Name	Price	Amount Paid	Purchase Date
-------------	---------	-----------	-------	-------------	---------------





Welcome {{username}}!!!

[Go Back to Home](#)

Contact Our Service

For any queries and information, contact us via email or mobile phone

Email : paymentbook.ad@gmail.com

Phone no : 7674897856