ระบบการเช่ารถยนต์

เสนอ

รศ.ตร อนิราช มิ่งขวัญ

จัดทำโดย

นายภูวดล เวชชนะ	**รหัส 6806022510131 Sec 2**
นายพงศ์พันธ์ หมันเทศมัน	**รหัส 6806022510645 Sec 2**
นายกิตติทัษน์ พุ่มเจริญ	**รหัส 6806022510190 Sec 2**
นายอาณัฐ อรรถวิเวก	**รหัส 6806022510220 Sec 2**
นายภูวดล เวชชนะ	**รหัส 6806022510131 Sec 2**

โครงงานนี้เป็นส่วนหนึ่งของการศึกษาหลักสูตรวิศวกรรมศาสตรบัณฑิต

สาขาวิศวกรรมสารสนเทศและเครือข่าย ภาควิชาเทคโนโลยีสารสนเทศ คณะเทคโนโลยีและการจัดการอุตสาหกรรม มหาวิทยาลัยเทคโนโลยีพระจอมเกล้าพระนครเหนือ ปีการศึกษา 2568

คำนำ

รายงานฉบับนี้จัดทำขึ้นเพื่อสรุปผลการศึกษาและการพัฒนาระบบการเช่ารถยนต์ ซึ่งเป็นส่วนหนึ่งของการเรียนในวิชา Computer Programming วิชานี้มุ่งเน้นการพัฒนาทักษะการเขียนโปรแกรมและการประยุกต์ใช้ความรู้ด้าน Programming ในการสร้างระบบที่สามารถนำไปใช้งานได้จริง ระบบการเช่ารถยนต์นี้เป็นตัวอย่างของการนำแนวคิดการเขียนโปรแกรมมาใช้แก้ไขปัญหาในชีวิตประจำวันของธุรกิจ เนื้อหาภายในรายงานจะครอบคลุมขั้นตอนการออกแบบและพัฒนาระบบตั้งแต่การวิเคราะห์ความต้องการ (Requirements Analysis) การเขียนอัลกอริทึม และการใช้ภาษาคอมพิวเตอร์ในการพัฒนาระบบ พร้อมทั้งอธิบายกระบวนการทำงานของแต่ละฟังก์ชันภายในระบบ

รายงานฉบับนี้ได้จัดทำขึ้นเพื่อเป็นการสรุปประสบการณ์และความรู้ที่ได้จากการเรียนและพัฒนาระบบจริง หวังเป็นอย่างยิ่งว่าจะเป็นประโยชน์ทั้งต่อผู้ศึกษาและผู้ที่สนใจในการนำความรู้ทาง Programming ไปประยุกต์ใช้ในการพัฒนาระบบจัดการต่าง ๆ ในอนาคต

สารบัญ

สารบัญ (ต่อ)

สารบัญรูปภาพ

หน้า

สารบัญรูปภาพ (ต่อ)

หน้า

สารบัญรูปภาพ (ต่อ)

หน้า

สารบัญตาราง

บทที่ 1
บทนำ

1.1 วัตถุประสงค์ของโครงงาน

1.1.1 เพื่อพัฒนาระบบการเช่ารถยนต์ได้อย่างมีประสิทธิภาพ

1.1.2 เพื่อฝึกฝนทักษะการเขียนโปรแกรมด้วย Python

1.1.3 เพื่อฝึกกระบวนการคิดอย่างเป็นระบบ

1.1.4 เพื่อเรียนรู้วิธีการจัดการข้อมูลและไฟล

1.2 ขอบเขตของโครงงาน

1.2.1 ระบบจัดการเช่ารถยนต์มีฟังก์ชันพื้นฐาน 14 ฟังก์ชัน ดังนี้

1. add_customer() – เพิ่มข้อมูลลูกค้าใหม่
2. add_car() – เพิ่มข้อมูลรถใหม่
3. add_contract() – เปิดสัญญาเช่า
4. update_customer() – อัปเดตข้อมูลลูกค้า
5. update_car() – อัปเดตข้อมูลรถ (เช็ก consistency กับสัญญา)
6. return_car() – ปิดสัญญา / คืนรถ
7. delete_customer() – ลบลูกค้า (soft delete)
8. delete_car() – ลบรถ (soft delete)
9. delete_contract() – ลบสัญญา (soft delete)
10. view_single() – แสดงข้อมูลทีละเรคคอร์ด (ลูกค้า/รถ/สัญญา)
11. view_all() – แสดงข้อมูลทั้งหมดในตาราง
12. view_filter() – แสดงข้อมูลแบบมีการค้นหา/กรอง
13. view_stats() – แสดงสถิติ (จำนวนรถในแต่ละสถานะ, สัญญาที่เปิดอยู่)
14. generate_report() – สร้างไฟล์รายงานสรุป

1.2.2 ระบบจัดการเช่ารถยนต์ประกอบด้วย 5 ไฟล์ ได้แก่

1. car_rental_binio.py
2. cars.bin
3. contracts.bin
4. customers.bin
5. report.txt

1.3 ประโยชน์ที่ได้รับ

1.3.1 พัฒนาทักษะการเขียนโปรแกรม

1.3.2 เพิ่มประสิทธิภาพในการจัดการเช่ารถยนต์

1.3.3 ฝึกการคิดวิเคราะห์และแก้ไขปัญหา

1.3.4 ความเข้าใจในการทำงานร่วมกับฐานข้อมูล

1.4 เครื่องมือที่คาดว่าจะต้องใช้

1.4.1 ภาษาโปรแกรมที่ใช้ในการพัฒนาระบบ คือ Python สำหรับการพัฒนาโปรแกรมพื้นฐานที่เข้าใจง่ายและมีไลบรารีหลากหลายที่ช่วยจัดการข้อมูลได้ดี

1.4.2 Visual Studio Code เป็นโปรแกรม Text Editor ที่ออกแบบมาเพื่อให้เป็นเครื่องมือที่ช่วยในการเขียนโค้ด โดยสามารถรองรับภาษาโปรแกรมต่าง ๆ เช่น Python, JavaScript และอื่น ๆ

1.4.3 GitHub แพลตฟอร์มที่ช่วยในการเก็บโค้ดและทำงานร่วมกับผู้อื่นในการทำ Project

บทที่ 2

ระบบการเช่ารถ

2.1สเปกไฟล์และระเบียน

ระบบการเช่ารถประกอบด้วยฐานข้อมูลหลัก 3 ส่วน ได้แก่ ลูกค้า (Customers), รถ (Cars) และสัญญาเช่า (Contracts) โดยแต่ละฐานมีบทบาทและฟิลด์สำคัญดังนี้

2.1.1 ลูกค้า (Customers)

หน้าที่: เก็บข้อมูลระบุตัวตนและการติดต่อของผู้เช่า

คีย์หลัก: cus_id (เลขรันอัตโนมัติ)

#	ฟิลด์	ชนิด (struct)	ขนาด (bytes)	ตัวอย่าง
1	flag (0=Deleted,1=Active)	B	1	1
2	cus_id (PK)	I	4	1001
3	id_card	13s	13	"1103700123456"
4	name	50s	50	"Somchai Boonmee"
5	phone	10s	10	"0812345678"
6	birth_ymd (YYYYMMDD)	I	4	19990217
7	gender (0=unk,1=male,2=female)	B	1	1
–	padding	45x	45	—

หมายเหตุ: สตริงเข้ารหัส UTF-8 ตัด/เติม NULL ให้พอดีความยาว

ตารางที่ 2-1 *Customer.bin*

2.1.1.1 flag (B, *1* byte)

ใช้เป็นตัวบอกสถานะของเรคคอร์ด ค่า 1 หมายถึง Active (ข้อมูลใช้งานได้) ค่า 0 หมายถึง Deleted (ถูกลบแล้ว แต่ยังคงอยู่ในไฟล์เพื่อใช้ระบบ Free-list) ฟิลด์นี้ช่วยให้ระบบรองรับ Soft Delete โดยไม่ต้องลบข้อมูลจริงออกจากไฟล์

2.1.1.2 cus_id (I, 4 bytes)

รหัสลูกค้า (Primary Key) กำหนดโดยระบบ อัตโนมัติ (Auto-increment) ใช้เป็นคีย์หลักในการอ้างอิงลูกค้าจากตารางอื่น เช่น สัญญาเช่า (Contracts)

2.1.1.3 id_card (13s, 13 bytes)

เลขบัตรประชาชน หรือเลขบัตรประจำตัวอื่น ๆ ของลูกค้า ใช้ตรวจสอบความถูกต้องและป้องกันข้อมูลซ้ำซ้อน กำหนดความยาว 13 ตัวอักษร (String)

2.1.1.4 name (50s, 50 bytes)

เก็บชื่อลูกค้า (ภาษาไทย/อังกฤษ)ความยาวสูงสุด 50 ตัวอักษร (UTF-8)ใช้แสดงผลและค้นหา

2.1.1.5 phone (10s, 10 bytes)

เก็บหมายเลขโทรศัพท์ของลูกค้า ความยาวสูงสุด 10 หลัก ใช้ในการติดต่อกลับหรือยืนยันตัวตน

2.1.1.6 birth_ymd (I, 4 bytes)

วันเกิดของลูกค้า เก็บในรูปแบบ YYYYMMDD เช่น 19990217 ใช้ในฟังก์ชันตรวจสอบอายุ, โปรโมชั่น หรือการยืนยันตัวตน ถ้าไม่ระบุ เก็บค่าเป็น 0

2.1.1.7 gender (B, 1 byte)

เพศของลูกค้า กำหนดให้รหัส: 0 = unknown, 1 = male, 2 = female ใช้เพื่อการวิเคราะห์ข้อมูลหรือรายงานสถิติ

2.1.1.8 padding (45x, 45 bytes)

พื้นที่เผื่อไว้สำหรับการขยายโครงสร้างในอนาคตป้องกันปัญหาการเปลี่ยนแปลงโครงสร้างไฟล์ที่อาจกระทบการอ่าน/เขียนข้อมูลค่าเริ่มต้นเป็น NULL (ว่างทั้งหมด)

2.1.2 รถ (Cars)

หน้าที่: เก็บข้อมูลรถ, อัตราค่าเช่า และสถานะปัจจุบัน

คีย์หลัก: car_id (เลขรันอัตโนมัติ)

#	ฟิลด์	ชนิด (struct)	ขนาด(bytes)	ตัวอย่าง
1	flag (0=Deleted,1=Active)	B	1	1
2	car_id (PK)	I	4	1001
3	license_plate	12s	12	"ABC-1234"
4	brand	12s	12	"Toyota"
5	model	16s	16	"Camry"
6	year	H	2	2021
7	rate_cents (บาท ×100/วัน)	I	4	150000
8	odometer_km	I	4	42850
9	status (0=available,1=rented, 2=maintenance,3=retired)	B	1	0
10	updated_at (Unix ts)	I	4	1693142400
–	padding	68x	68	—

อัตราค่าเช่า: เก็บเป็นสตางค์/วัน (เช่น 1,500.00 บาท ⇒ 150000)

ตารางที่ 2-2 *car.bin*

2.1.2 อธิบายการทำงานของแต่ละฟิลด์ใน (Cars)

2.1.2.1 flag (B, 1 byte)

ใช้ระบุสถานะเรคคอร์ด 1 = Active (ใช้งานได้) 0 = Deleted (ถูกลบออก แต่ยังคงอยู่ในไฟล์สำหรับ Free-list) รองรับ Soft Delete เพื่อไม่ให้เสียตำแหน่งข้อมูล

2.1.2.2 car_id (I, 4 bytes)

เลขรหัสประจำรถ (Primary Key) ระบบกำหนดอัตโนมัติ (Auto-increment) ใช้เป็นคีย์หลักในการอ้างอิงกับตารางสัญญาเช่า (Contracts)

2.1.2.3 license_plate (12s, 12 bytes)

ทะเบียนรถ (เช่น "ABC-1234") ใช้ตรวจสอบและระบุตัวตนของรถ กำหนดความยาว 12 ตัวอักษร (String, UTF-8)

2.1.2.4 brand (12s, 12 bytes)

ยี่ห้อรถ เช่น "Toyota", "Honda" ความยาวสูงสุด 12 ตัวอักษร

2.1.2.5 model (16s, 16 bytes)

รุ่นรถ เช่น "Camry", "Civic"ความยาวสูงสุด 16 ตัวอักษร

2.1.2.6 year (H, 2 bytes)

ปีที่ผลิตรถ (เช่น 2021) ใช้ในการระบุรุ่น/อายุการใช้งานรถ ใช้ตัวเลข 2 byte (short int)

2.1.2.7 rate_cents (I, 4 bytes)

อัตราค่าเช่าต่อวัน (เก็บในหน่วย สตางค์) เช่น 1,500.00 บาท/วัน จะเก็บเป็น 150000 ใช้ชนิด Integer 4 byte เพื่อรองรับการคำนวณแม่นยำ

2.1.2.8 odometer_km (I, 4 bytes)

เลขไมล์ของรถ (หน่วยกิโลเมตร) ใช้ติดตามการใช้งานจริง และประกอบการบำรุงรักษา

2.1.2.9 status (B, 1 byte)

สถานะปัจจุบันของรถ

0 = available (ว่างให้เช่า)

1 = rented (กำลังถูกเช่า)

2 = maintenance (อยู่ระหว่างซ่อมบำรุง)

3 = retired (ปลดระวาง, เลิกใช้งาน)

ใช้ตรวจสอบความพร้อมของรถก่อนเปิดสัญญา

2.1.2.10 updated_at (I, 4 bytes)

เวลาที่อัปเดตข้อมูลล่าสุด (เก็บเป็น Unix Timestamp) ใช้ตรวจสอบประวัติการแก้ไข เช่น การอัปเดตสถานะหรือเลขไมล์

2.1.2.11 padding (68x, 68 bytes)

พื้นที่สำรองสำหรับการขยายโครงสร้างไฟล์ในอนาคต ค่าเริ่มต้นเป็น NULL ทำให้ขนาดเรคคอร์ดคงที่ และรองรับการปรับปรุงโครงสร้างภายหลังโดยไม่กระทบระบบเดิม

2.1.3 สัญญาเช่า (Contracts)

หน้าที่: บันทึกความสัมพันธ์ “ลูกค้า–รถ” ในแต่ละดีลการเช่า รวมถึงช่วงเวลาและยอดเงิน

คีย์หลัก: rent_id (เลขรันอัตโนมัติ)

#	ฟิลด์	ชนิด (struct)	ขนาด (bytes)	ตัวอย่าง
1	flag (0=Deleted,1=Active)	B	1	1
2	rent_id (PK)	I	4	2001
3	cus_id (FK ⟶ customers.cus_id)	I	4	1001
4	car_id (FK ⟶ cars.car_id)	I	4	1001
5	rent_ymd (YYYYMMDD)	I	4	20250422
6	return_ymd (0=ยังไม่คืน)	I	4	20250425
7	total_cents (บาท ×100)	I	4	450000
8	returned (0=open,1=closed)	B	1	1
–	padding	38x	38	—

ตารางที่ 2-3 *Contract.bin*

2.1.3 อธิบายการทำงานของแต่ละฟิลด์ใน Contracts

2.1.3.1 flag (B, 1 byte)

ระบุสถานะเรคคอร์ด 1 = Active (ใช้งานได้), 0 = Deleted (ลบแบบ Soft Delete)

รองรับการจัดการ Free-list โดยไม่ต้องลบข้อมูลออกจากไฟล์จริง

2.1.3.2 rent_id (I, 4 bytes)

รหัสสัญญาเช่า (Primary Key) เลขรันอัตโนมัติ (Auto-increment)

ใช้อ้างอิงในการแก้ไข/ปิดสัญญา

2.1.3.3 cus_id (I, 4 bytes)

รหัสลูกค้า (Foreign Key customers.cus_id) ใช้เชื่อมโยงสัญญากับข้อมูลผู้เช่า

2.1.3.4 car_id (I, 4 bytes)

รหัสรถ (Foreign Key cars.car_id) ใช้เชื่อมโยงสัญญากับข้อมูลรถที่เช่า

2.1.3.5 rent_ymd (I, 4 bytes)

วันที่เริ่มเช่า (เก็บเป็นรูปแบบ YYYYMMDD) เช่น 20250422 22 เม.ย. 2025

ใช้เป็นวันเริ่มต้นในการคำนวณค่าเช่า

2.1.3.6 return_ymd (I, 4 bytes)

วันที่คืนรถ (เก็บเป็น YYYYMMDD) ถ้ายังไม่คืน เก็บค่า 0

2.1.3.7 total_cents (I, 4 bytes)

ยอดค่าเช่ารวม (เก็บในหน่วย สตางค์)เช่น 4,500.00 บาท เก็บเป็น 450000

คำนวณจาก (จำนวนวัน × ค่าเช่ารถต่อวัน)

2.1.3.8 returned (B, 1 byte)

สถานะสัญญา

0 = open (ยังเช่าอยู่)

1 = closed (ปิดสัญญาแล้ว/คืนรถแล้ว) ใช้ตรวจสอบความถูกต้องก่อนเปิดหรือปิดสัญญาใหม่

2.1.3.9 padding (38x, 38 bytes)

พื้นที่สำรองสำหรับขยายโครงสร้างไฟล์ในอนาคต ค่าดีฟอลต์เป็น NULL ทำให้โครงสร้างเรคคอร์ดมีความยืดหยุ่นและขนาดคงที่

2.2 รายการเช่ารถ (Rental Report)

```
Car Rent System — Rental Report
Generated At : 2025-10-01 15:43:13 (+0700)

รายการรถที่มีคนเช่า (ทั้งที่คืนแล้วและยังไม่คืน)

Renter              | Plate     | Brand     | Model       |    Rate | Rent Time     | Days |     Amount | Status
-------------------------------------------------------------------------------------------------------------
Sudarat Boonmee     | TH-0010   | Mitsu     | Mirage      |    1800 | 06-22->06-27  |    5 |    9000.00 | คืนแล้ว
Sudarat Inthra      | TH-0004   | Nissan    | Note        |    1200 | 07-22->10-01  |   71 |   85200.00 | เช่าอยู่
Sudarat Inthra      | TH-0014   | Nissan    | Note        |    1200 | 07-22->10-01  |   71 |   85200.00 | เช่าอยู่
Warin Chaiyakul     | TH-0008   | Nissan    | Note        |    1200 | 08-18->10-01  |   44 |   52800.00 | เช่าอยู่
Warin Chaiyakul     | TH-0018   | Nissan    | Note        |    1200 | 08-18->09-30  |   43 |   51600.00 | คืนแล้ว
Napat Inthra        | TH-0003   | Honda     | Jazz        |    1200 | 09-03->09-04  |    1 |    1200.00 | คืนแล้ว
Napat Inthra        | TH-0013   | Honda     | Jazz        |    1200 | 09-03->09-04  |    1 |    1200.00 | คืนแล้ว
Siriporn Chanthara  | TH-0002   | Mazda     | CX-30       |    2000 | 09-13->09-18  |    5 |   10000.00 | คืนแล้ว
Siriporn Chanthara  | TH-0012   | Mazda     | CX-30       |    2000 | 09-13->09-18  |    5 |   10000.00 | คืนแล้ว
Puvadon Wetchana    | TH-1234   | Honda     | wave-120    |     200 | 10-01->10-01  |    1 |     200.00 | เช่าอยู่
Puvadon Wetchana    | TH-0020   | Mitsu     | Mirage      |    1800 | 10-01->10-01  |    1 |    1800.00 | คืนแล้ว
Puvadon Wetchana    | TG-4321   | Mazda     | 2           |    2000 | 10-01->10-01  |    1 |    2000.00 | เช่าอยู่

สรุป: 12 รายการ
รวมยอดเงิน: 310,200.00 บาท

Summary (นับเฉพาะสถานะ Active)
- Total Cars (records) : 22
- Active Cars          : 22
- Deleted Cars         : 0
- Currently Rented     : 5
- Available Now        : 17

Cars by Brand (Active only)
- Honda : 5
- Mazda : 9
- Mitsu : 4
- Nissan : 4
```

ภาพที่ 2-1 *Rental Report*

2.2.1 header_title ส่วนหัวรายงาน

เป็นข้อความ UTF-8 ความยาวไม่เกิน ~100 ไบต์ ใช้ระบุชื่อรายงาน เช่น "Car Rent System - Rental Report" เพื่อบอกผู้ใช้งานว่าไฟล์นี้เป็นรายงานอะไรและของระบบใด

2.2.2 generated_at วันที่และเวลาที่สร้างรายงาน

เป็นข้อความ UTF-8 ความยาวไม่เกิน ~30 ไบต์ แสดงวันที่และเวลาที่สร้างรายงานในรูปแบบ YYYY-MM-DD HH:MM:SS (+ZZZZ) เช่น "2025-10-01 15:43:13 (+0700)" ช่วยอ้างอิงว่าไฟล์นี้ถูกสร้างเมื่อใดและตามเขตเวลาใด

2.4.3 rentals_caption คำอธิบายส่วนรายการเช่า

เป็นข้อความสั้นๆ เพื่อขึ้นหัวส่วนข้อมูลเช่า เช่น "รายการรถที่มีคนเช่า (ทั้งที่คืนแล้วและยังไม่คืน)" ทำให้ผู้อ่านทราบว่าส่วนถัดไปคือบันทึกรายการเช่าทั้งหมด

2.4.4 rentals_table_header หัวตารางรายการเช่า

เป็นข้อความหนึ่งบรรทัดกำหนดหัวคอลัมน์และความกว้างแบบ fixed-width เช่น "Renter | Plate | Brand | Model | Rate | Rent Time | Days | Amount | Status" ใช้เป็นแม่แบบให้แต่ละเรคคอร์ดเช่าจัดเรียงคอลัมน์ตรงกัน อ่านง่าย

2.4.5 rentals_rows แถวข้อมูลรายการเช่า

เป็นข้อมูลหลายบรรทัด (N แถว ตามจำนวนสัญญาที่มีการเช่าจริง) จัดรูปแบบคอลัมน์คงที่ โดยมีความหมายดังนี้:

1. Renter ชื่อลูกค้า (ดึงจาก customers.bin)
2. Plate / Brand / Model ทะเบียน ยี่ห้อ รุ่น (ดึงจาก cars.bin)
3. Rate อัตราค่าเช่าต่อวัน (บาท/วัน)
4. Rent Time ช่วงวันที่เช่าในรูป MM-DD->MM-DD (หากยังไม่คืน ปลายช่วงคือวันที่ปัจจุบัน)
5. Days จำนวนวันเช่า (ขั้นต่ำ 1 วัน คำนวณจากต่างวันที่)
6. Amount ยอดเงินของรายการนั้น (สัญญาปิดแล้วใช้ total_cents, หากยังเช่าอยู่คิด Days × Rate)
7. Status สถานะ "คืนแล้ว" หรือ "เช่าอยู่"
8. แถวเหล่านี้มักมีเส้นคั่น ----- ก่อน/หลังเพื่อแบ่งส่วนอย่างชัดเจน

2.4.6 rentals_summary สรุปรายการเช่า

เป็นข้อความ 1–2 บรรทัด เช่น

"สรุป: 12 รายการ" และ "รวมยอดเงิน: 310,200.00 บาท"

ใช้สรุปจำนวนเรคอร์ดเช่าที่พิมพ์และยอดรวมทั้งหมดเพื่อประเมินภาพรวมรายได้จากการเช่า

2.4.7 fleet_summary สรุปภาพรวมฝั่งรถ (นับเฉพาะ Active)

บล็อกข้อความหลายบรรทัด แสดงตัวชี้วัดคลังรถ:

Total Cars (records) จำนวนเรคคอร์ดรถทั้งหมด (Active + Deleted)

Active Cars จำนวนรถที่ยังใช้งาน (flag=1)

Deleted Cars จำนวนเรคคอร์ดที่ถูกลบแบบ soft delete

Currently Rented จำนวนรถที่สถานะกำลังเช่า (status=1)

Available Now จำนวนรถที่ว่าง (status=0)

ส่วนนี้ช่วยตรวจสุขภาพคลังรถ ณ เวลาสร้างรายงาน

2.4.8 cars_by_brand สถิติจำนวนรถตามยี่ห้อ (เฉพาะ Active)

เป็นรายการบรรทัด - <Brand> : <Count> เช่น

Honda : 5

Mazda : 9 ... เพื่อเห็นการกระจายของรถตามยี่ห้อ ซึ่งมีประโยชน์ต่อการวางแผนจัดซื้อ/ปล่อยเช่า

บทที่ 3

การใช้งานระบบเช่ารถยนต์

โปรแกรมระบบเช่ารถ CarRent-BinIO ถูกพัฒนาขึ้นเพื่อช่วยธุรกิจเช่ารถขนาดเล็กถึงกลางในการจัดเก็บและบริหารข้อมูล ลูกค้า รถ และสัญญาเช่า อย่างเป็นระบบ ใช้งานผ่านเมนูบรรทัดคำสั่ง(CLI) ที่เรียบง่าย แต่ครอบคลุมการท างานหลักครบวงจร ได้แก่ เพิ่ม/แก้ไข/ลบ ค้นดูข้อมูล และออกรายงานสรุปสถานะปัจจุบัน

ระบบออกแบบให้ท างานรวดเร็วและเชื่อถือได้ด้วยไฟล์ไบนารีแบบระเบียนความยาวคงที่ (fixed-length records) พร้อม ดัชนีแบบ open addressing, free-list, และ soft delete เพื่อให้การค้นหาและการจัดสรรพื้นที่เก็บข้อมูลมีประสิทธิภาพ อีกทั้งยังมีการตรวจสอบความถูกต้องของข้อมูล (เช่น บัตรประชาชน 13 หลัก เบอร์โทร ปีรถ วันเช่า/วันคืน) และกลไก คงความสอดคล้องระหว่างสถานะรถกับสัญญา (เปิดสัญญา = รถต้องว่าง, คืนรถ = ปิดสัญญาและตั้งรถกลับเป็นว่าง

สำหรับผู้ใช้งานโปรแกรม

3.1 การใช้งานโปรแกรมระบบการเช่ารถยนต์

3.1.1 เริ่มต้นการทำงานของระบบการเช่ารถยนต์ จะแสดงหน้าเมนูการใช้งานต่างๆ ประกอบไปด้วย Add, Update, Delete, View, Report, Exit และเลือกหมายเลขในการเข้าใช้การทำงาน

```
===== CarRent-BinIO =====
1) Add  2) Update  3) Delete  4) View  5) Report  0) Exit
เลือก:
```

ภาพที่ 3-1 *เมนูหลักการใช้งานโปรแกรม*

3.1.2 กรอกหมายเลข 1 เพื่อเรียกใช้ฟังก์ชัน Add เพิ่มข้อมูลที่ประกอบไปด้วย Customer, Car, Contract, Back

```
===== CarRent-BinIO =====
1) Add  2) Update  3) Delete  4) View  5) Report  0) Exit
เลือก:  1
```

ภาพที่ 3-2 การเลือกใช้งานฟังก์ชัน Add

3.1.3 เมื่อเลือกเมนูฟังก์ชัน Add ขึ้นมาแล้วจากนั้นก็สามารถระบุเมนูที่ต้องการเลือกได้

```
===== CarRent-BinIO =====
1) Add  2) Update  3) Delete  4) View  5) Report  0) Exit
เลือก:  1

[Add] 1) Customer 2) Car 3) Contract  0) Back
เลือก: 1
```

ภาพที่ 3-3 เมนูของ Add

3.1.4 กรอกหมายเลข 2 เพื่อเรียกใช้ฟังก์ชัน Update เพิ่มข้อมูลที่ประกอบไปด้วย Customer, Car, Return Car, Back

```
===== CarRent-BinIO =====
1) Add  2) Update  3) Delete  4) View  5) Report  0) Exit
เลือก: 2
```

ภาพที่ 3-4 การเลือกใช้งานฟังก์ชัน Update

3.1.5 เมื่อเลือกเมนูฟังก์ชัน Update ขึ้นมาแล้วจากนั้นก็สามารถระบุเมนูที่ต้องการเลือกได้

```
===== CarRent-BinIO =====
1) Add  2) Update  3) Delete  4) View  5) Report  0) Exit
เลือก: 2

[Update] 1) Customer 2) Car 3) Return Car  0) Back
เลือก:
```

ภาพที่ 3-5 เมนูของ Update

3.1.6 กรอกหมายเลข 3 เพื่อเรียกใช้ฟังก์ชัน Delete เพิ่มข้อมูลที่ประกอบไปด้วย Customer, Car, Contract, Back

```
===== CarRent-BinIO =====
1) Add  2) Update  3) Delete  4) View  5) Report  0) Exit
เลือก: 3
```

ภาพที่ 3-6 การเลือกใช้งานฟังก์ชัน Delete

3.1.7 เมื่อเลือกเมนูฟังก์ชัน Delete ขึ้นมาแล้วจากนั้นก็สามารถระบุเมนูที่ต้องการเลือกได้

```
===== CarRent-BinIO =====
1) Add  2) Update  3) Delete  4) View  5) Report  0) Exit
เลือก: 3

[Delete] 1) Customer 2) Car 3) Contract  0) Back
เลือก:
```

ภาพที่ 3-7 เมนูของ Delete

3.1.8 กรอกหมายเลข 4 เพื่อเรียกใช้ฟังก์ชัน View เพิ่มข้อมูลที่ประกอบไปด้วย เดี่ยว, ทั้งหมด, กรอง, สถิติ, Back

```
===== CarRent-BinIO =====
1) Add  2) Update  3) Delete  4) View  5) Report  0) Exit
เลือก: 4
```

ภาพที่ 3-8 การเลือกใช้งานฟังก์ชัน View

3.1.9 เมื่อเลือกเมนูฟังก์ชัน View ขึ้นมาแล้วจากนั้นก็สามารถระบุเมนูที่ต้องการเลือกได้

```
===== CarRent-BinIO =====
1) Add  2) Update  3) Delete  4) View  5) Report  0) Exit
เลือก: 4

[View] 1) เดี่ยว 2) ทั้งหมด 3) กรอง 4) สถิติ  0) Back
เลือก:
```

ภาพที่ 3-9 เมนูของ View

3.1.10 กรอกหมายเลข 5 เพื่อเรียกใช้ฟังก์ชัน Report จะบันทึกข้อมูลลงไฟล์ report.txt

```
===== CarRent-BinIO =====
1) Add  2) Update  3) Delete  4) View  5) Report  0) Exit
เลือก: 5
* เขียนรายงานที่ data\report.txt
```

ภาพที่ 3-10 การเลือกใช้งานฟังก์ชัน Report

3.1.11 กรอกหมายเลข 0 เพื่อเรียกใช้ฟังก์ชัน Exit จะบันทึกข้อมูลลงไฟล์ report.txt และออกจากโปรแกรม

```
===== CarRent-BinIO =====
1) Add  2) Update  3) Delete  4) View  5) Report  0) Exit
เลือก: 0
* เขียนรายงานที่ data\report.txt
บันทึกและออก...
```

ภาพที่ 3-11 การเลือกใช้งานฟังก์ชัน Exit

3.2 การใช้งานโปรแกรมเพิ่มข้อมูล

3.2.1 กรอกหมายเลข 1 เพื่อเข้าฟังก์ชัน Add เพื่อเพิ่มข้อมูลทั้งหมดที่มีในโปรแกรม

```
===== CarRent-BinIO =====
1) Add  2) Update  3) Delete  4) View  5) Report  0) Exit
เลือก: 1

[Add] 1) Customer 2) Car 3) Contract  0) Back
เลือก:
```

ภาพที่ 3-12 การเลือกใช้งานฟังก์ชัน Add

3.2.2 กรอกหมายเลข 1 เพื่อเข้าเมนู Customer เพื่อเพิ่มข้อมูลลูกค้าและจากนั้นใส่ข้อมูลในหัวข้อทั้งหมดดังภาพที่ 3-13

```
[Add] 1) Customer 2) Car 3) Contract  0) Back
เลือก:  1
ชื่อ-นามสกุล: John Miller
เลขบัตร 13 หลัก: 1234567891011
โทร (9-10 หลัก): 0863334444
วันเกิด YYYY-MM-DD: 1987-12-12
เพศ (unk/male/female) [unk]: male
+ เพิ่มลูกค้า id=1
```

ภาพที่ 3-13 การเพิ่มข้อมูลลูกค้า

3.2.3 กรอกหมายเลข 2 เพื่อเข้าเมนู Car เพื่อเพิ่มข้อมูลรถยนต์และจากนั้นใส่ข้อมูลในหัวข้อทั้งหมดดังภาพที่ 3-14

```
[Add] 1) Customer 2) Car 3) Contract  0) Back
เลือก: 2
ทะเบียน (<=16):TH-5566
ยี่ห้อ (<=12): BMW
รุ่น (<=16): BMW-i5-M60
ปีผลิต: 2024
ค่าเช่าต่อวัน (บาท): 1000
เลขไมล์ (กม.): 100
สถานะ [available]: available
+ เพิ่มรถ id=1
```

ภาพที่ 3-14 การเพิ่มข้อมูลรถยนต์

3.2.4 กรอกหมายเลข 3 เพื่อเข้าเมนู Contract เพื่อเพิ่มข้อมูลผู้เช่ารถยนต์และจากนั้นใส่ข้อมูลในหัวข้อทั้งหมดดังภาพที่ 3-15

```
[Add] 1) Customer 2) Car 3) Contract  0) Back
เลือก: 3
cus_id: 1
car_id: 9
+ เปิดสัญญา rent_id=11
```

ภาพที่ 3-15 การเพิ่มข้อมูลผู้เช่ารถยนต์

3.3 การใช้งานโปรแกรมแก้ไขข้อมูลใหม่

3.3.1 กรอกหมายเลข 2 เพื่อเข้าฟังก์ชัน Update เพื่อแก้ไขข้อมูลใหม่ทั้งหมดที่มีในโปรแกรม

```
===== CarRent-BinIO =====
1) Add  2) Update  3) Delete  4) View  5) Report  0) Exit
เลือก: 2

[Update] 1) Customer 2) Car 3) Return Car  0) Back
เลือก: 1
```

ภาพที่ 3-16 การเลือกใช้งานฟังก์ชัน Update

3.3.2 กรอกหมายเลข 1 เพื่อเข้าเมนู Customer เพื่อแก้ไขข้อมูลลูกค้าและจากนั้นใส่ข้อมูลในหัวข้อทั้งหมดดังภาพที่ 3-17

```
[Update] 1) Customer 2) Car 3) Return Car  0) Back
เลือก: 1
cus_id: 1
ชื่อ [Sudarat Boonmee]: Harris Pemberton
บัตร13 [1103700000001]: 3216549873265
โทร [0846913810]: 0923216547
เกิด YYYY-MM-DD [1987-04-05]: 1985-12-12
เพศ (unk/male/female) [คงเดิม]: male
* อัปเดตลูกค้าแล้ว
```

ภาพที่ 3-17 การแก้ไขข้อมูลลูกค้า

3.3.3 กรอกหมายเลข 2 เพื่อเข้าเมนู Car เพื่อแก้ไขข้อมูลรถยนต์และจากนั้นใส่ข้อมูลในหัวข้อทั้งหมดดังภาพที่ 3-18

```
[Update] 1) Customer 2) Car 3) Return Car  0) Back
เลือก: 2
car_id: 1
ทะเบียน [TH-0001]:TH-3216
ยี่ห้อ [Mazda]: BMW
รุ่น [2]: BMW-M4
ปีผลิต [2022]: 2025
ค่าเช่าบาท [1500.00]: 1000
เลขไมล์ [32460]: 100
สถานะ (available/rented/maintenance/retired) [available]: available
* อัปเดตรถแล้ว
```

ภาพที่ 3-18 การแก้ไขข้อมูลรถยนต์

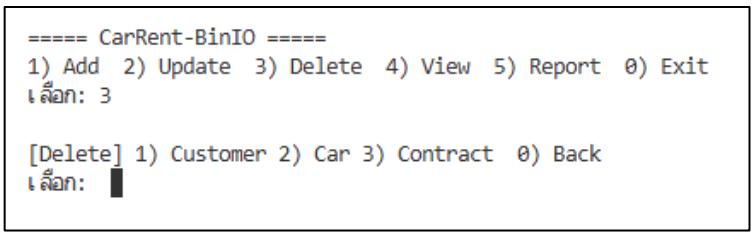
3.3.4 กรอกหมายเลข 3 เพื่อเข้าเมนู Return Car เพื่อทำการคืนรถยนต์และจากนั้นใส่ข้อมูลในหัวข้อทั้งหมดดังภาพที่ 3-19

```
[Update] 1) Customer 2) Car 3) Return Car  0) Back
เลือก: 3
rent_id: 1
วันคืน YYYY-MM-DD หรือพิมพ์ today: today
! ปิดสัญญาแล้ว
```

ภาพที่ 3-19 การคืนรถยนต์

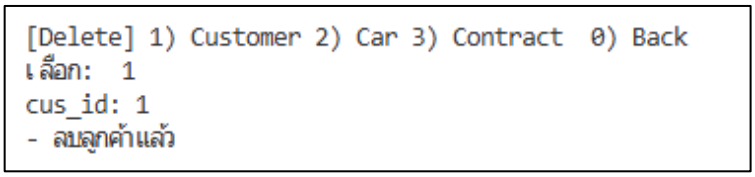
3.4 การใช้งานโปรแกรมลบข้อมูลผู้เช่ารถยนต์

3.4.1 กรอกหมายเลข 3 เพื่อเข้าฟังก์ชัน Delete เพื่อลบข้อมูลทั้งหมดที่มีในโปรแกรม



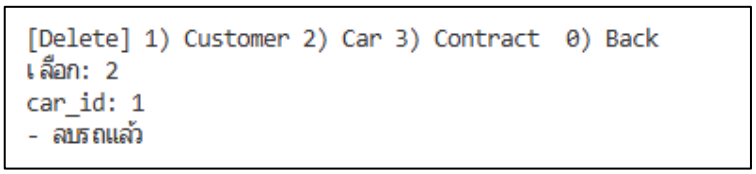
ภาพที่ 3-20 การเลือกใช้งานฟังก์ชัน Delete

3.4.2 กรอกหมายเลข 1 เพื่อเข้าเมนู Customer เพื่อลบข้อมูลลูกค้าและจากนั้นใส่ข้อมูลในหัวข้อทั้งหมดดังภาพที่ 3-21



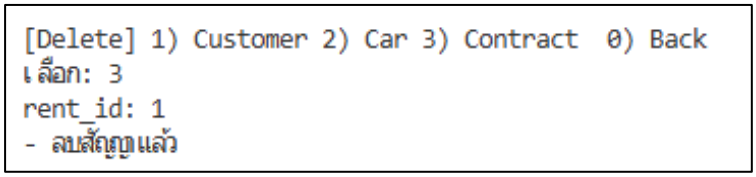
ภาพที่ 3-21การลบข้อมูลลูกค้า

3.4.3 กรอกหมายเลข 2 เพื่อเข้าเมนู Car เพื่อลบข้อมูลรถยนต์และจากนั้นใส่ข้อมูลในหัวข้อทั้งหมดดังภาพที่ 3-22



ภาพที่ 3-22 การลบข้อมูลรถยนต์

3.4.4 กรอกหมายเลข 3 เพื่อเข้าเมนู Contract เพื่อลบข้อมูลผู้เช่ารถยนต์และจากนั้นใส่ข้อมูลในหัวข้อทั้งหมดดังภาพที่ 3-23



ภาพที่ 3-23 การลบข้อมูลผู้เช่ารถยนต์

3.5 การใช้งานโปรแกรมแสดงข้อมูล

3.5.1 กรอกหมายเลข 4 เพื่อเข้าฟังก์ชัน View เพื่อดูข้อมูลทั้งหมดที่มีในโปรแกรม

```
===== CarRent-BinIO =====
1) Add  2) Update  3) Delete  4) View  5) Report  0) Exit
เลือก: 4

[View] 1) เดี่ยว 2) ทั้งหมด 3) กรอง 4) สถิติ  0) Back
เลือก:
```

ภาพที่ 3-24 การเลือกใช้งานฟังก์ชัน View

3.5.2 กรอกหมายเลข 1 เพื่อเข้าเมนู เดี่ยว เพื่อเข้าเมนูเลือกชนิดข้อมูล customer, car, contract

```
[View] 1) เดี่ยว 2) ทั้งหมด 3) กรอง 4) สถิติ  0) Back
เลือก: 1
ชนิด (customer/car/contract, 0=Back):
```

ภาพที่ 3-25 การเลือกใช้งานเมนูเดี่ยว

3.5.3 กรอก customer เพื่อเข้าเมนู customer เพื่อดูข้อมูลลูกค้าแบบเดี่ยวและจากนั้นใส่ข้อมูลในหัวข้อทั้งหมดดังภาพที่ 3-26

```
[View] 1) เดี่ยว 2) ทั้งหมด 3) กรอง 4) สถิติ  0) Back
เลือก: 1
ชนิด (customer/car/contract, 0=Back): customer
id: 1
[Customer] id=1 name=Harris Pemberton id_card=3216549873265 phone=0923216547 birth=1985-12-12 gender=1
```

ภาพที่ 3-26 การเลือกใช้งานดู customer

3.5.4 กรอก car เพื่อเข้าเมนู car เพื่อดูข้อมูลรถยนต์แบบเดี่ยวและจากนั้นใส่ข้อมูลในหัวข้อทั้งหมดดังภาพที่ 3-27

```
[View] 1) เดี่ยว 2) ทั้งหมด 3) กรอง 4) สถิติ  0) Back
เลือก: 1
ชนิด (customer/car/contract, 0=Back):  car
id: 1
[Car] id=1 plate=TH-3216 brand=BMW model=BMW-M4 year=2025 rate=1000.00 status=available
```

ภาพที่ 3-27 การเลือกใช้งานดู car

3.5.5 กรอก contract เพื่อเข้าเมนู contract เพื่อดูข้อมูลผู้เช่ารถยนต์แบบเดี่ยวและจากนั้นใส่ข้อมูลในหัวข้อทั้งหมดดังภาพที่ 3-28

```
[View] 1) เดี่ยว 2) ทั้งหมด 3) กรอง 4) สถิติ  0) Back
เลือก: 1
ชนิด (customer/car/contract, 0=Back): contract
id: 1
[Contract] id=1 cus_id=4 car_id=3 rent=2025-09-03 return=2025-09-04 total=1200.00 returned=1
```

ภาพที่ 3-28 การเลือกใช้งานดู contract

3.5.6 กรอกหมายเลข 2 เพื่อเข้าเมนู ทั้งหมด เพื่อเข้าเมนูเลือกชนิดข้อมูล customer, car, contract

```
[View] 1) เดี่ยว 2) ทั้งหมด 3) กรอง 4) สถิติ  0) Back
เลือก: 2
ชนิด (customer/car/contract, 0=Back):
```

ภาพที่ 3-29 การเลือกใช้งานเมนูทั้งหมด

3.5.7 กรอก customer เพื่อเข้าเมนู customer เพื่อดูข้อมูลลูกค้าทั้งหมดและจากนั้นใส่ข้อมูลในหัวข้อทั้งหมดดังภาพที่ 3-30

```
[View] 1) เดี่ยว 2) ทั้งหมด 3) กรอง 4) สถิติ  0) Back
เลือก: 2
ชนิด (customer/car/contract, 0=Back): customer
  ID | Name                 | Phone      | Birth      | Gender
-------------------------------------------------------------
   2 | Sudarat Inthra       | 0821668732 | 1998-07-02 | unk
   3 | Sudarat Prasert      | 0841227216 | 1996-10-01 | female
   4 | Napat Inthra         | 0866306997 | 1987-08-19 | male
   5 | Somchai Srisuk       | 0866722344 | 1990-05-05 | unk
   6 | Kittisak Chaiyakul   | 0822448136 | 1992-02-12 | male
   7 | Siriporn Chanthara   | 0815831819 | 2003-08-18 | unk
   8 | Warin Chaiyakul      | 0884093639 | 1989-11-20 | male
   9 | Siriporn Prasert     | 0819335534 | 1981-11-08 | male
  10 | Sudarat Prasert      | 0823556182 | 1992-05-15 | female
  11 | Sudarat Boonmee      | 0846913810 | 1987-04-05 | female
  12 | Sudarat Inthra       | 0821668732 | 1998-07-02 | unk
  13 | Sudarat Prasert      | 0841227216 | 1996-10-01 | female
  14 | Napat Inthra         | 0866306997 | 1987-08-19 | male
  15 | Somchai Srisuk       | 0866722344 | 1990-05-05 | unk
  16 | Kittisak Chaiyakul   | 0822448136 | 1992-02-12 | male
  17 | Siriporn Chanthara   | 0815831819 | 2003-08-18 | unk
  18 | Warin Chaiyakul      | 0884093639 | 1989-11-20 | male
  19 | Siriporn Prasert     | 0819335534 | 1981-11-08 | male
  20 | Sudarat Prasert      | 0823556182 | 1992-05-15 | female
```

ภาพที่ 3-30 การเลือกใช้งานดู customer

3.5.8 กรอก car เพื่อเข้าเมนู car เพื่อดูข้อมูลรถยนต์ทั้งหมดและจากนั้นใส่ข้อมูลในหัวข้อทั้งหมดดังภาพที่ 3-31

```
[View] 1) เดี่ยว 2) ทั้งหมด 3) กรอง 4) สถิติ  0) Back
เลือก:  2
ชนิด (customer/car/contract, 0=Back): car
  ID | Plate      | Brand      | Model      | Year |       Rate | Status
-----------------------------------------------------------------------------
   2 | TH-0002    | Mazda      | CX-30      | 2018 |    2000.00 | available
   3 | TH-0003    | Honda      | Jazz       | 2020 |    1200.00 | available
   4 | TH-0004    | Nissan     | Note       | 2025 |    1200.00 | rented
   5 | TH-0005    | Mazda      | 2          | 2020 |     900.00 | available
   6 | TH-0006    | Mazda      | 3          | 2021 |     900.00 | available
   7 | TH-0007    | Mitsu      | Xpander    | 2022 |    1200.00 | available
   8 | TH-0008    | Nissan     | Note       | 2024 |    1200.00 | rented
   9 | TH-0009    | Honda      | City       | 2025 |    2000.00 | rented
  10 | TH-0010    | Mitsu      | Mirage     | 2026 |    1800.00 | available
  11 | TH-0011    | Mazda      | 2          | 2022 |    1500.00 | available
  12 | TH-0012    | Mazda      | CX-30      | 2018 |    2000.00 | available
  13 | TH-0013    | Honda      | Jazz       | 2020 |    1200.00 | available
  14 | TH-0014    | Nissan     | Note       | 2025 |    1200.00 | rented
  15 | TH-0015    | Mazda      | 2          | 2020 |     900.00 | available
  16 | TH-0016    | Mazda      | 3          | 2021 |     900.00 | available
  17 | TH-0017    | Mitsu      | Xpander    | 2022 |    1200.00 | available
  18 | TH-0018    | Nissan     | Note       | 2024 |    1200.00 | rented
  19 | TH-0019    | Honda      | City       | 2025 |    2000.00 | available
  20 | TH-0020    | Mitsu      | Mirage     | 2026 |    1800.00 | available
```

ภาพที่ 3-31 การเลือกใช้งานดู car

3.5.9 กรอก contract เพื่อเข้าเมนู contract เพื่อดูข้อมูลผู้เช่ารถยนต์ทั้งหมดและจากนั้นใส่ข้อมูลในหัวข้อทั้งหมดดังภาพที่ 3-32

```
[View] 1) เดี่ยว 2) ทั้งหมด 3) กรอง 4) สถิติ  0) Back
เลือก: 2
ชนิด (customer/car/contract, 0=Back): contract
Rent_ID | Cus_ID | Name                | Car_ID | Rent_Time              |      Total
-------------------------------------------------------------------------------------
      2 |      2 | Sudarat Inthra      |      4 | 2025-07-22->-          |       0.00
      3 |      7 | Siriporn Chanthara  |      2 | 2025-09-13->2025-09-18 |   10000.00
      4 |      8 | Warin Chaiyakul     |      8 | 2025-08-18->-          |       0.00
      5 |      1 | Harris Pemberton    |     10 | 2025-06-22->2025-06-27 |    9000.00
      6 |     14 | Napat Inthra        |     13 | 2025-09-03->2025-09-04 |    1200.00
      7 |     12 | Sudarat Inthra      |     14 | 2025-07-22->-          |       0.00
      8 |     17 | Siriporn Chanthara  |     12 | 2025-09-13->2025-09-18 |   10000.00
      9 |     18 | Warin Chaiyakul     |     18 | 2025-08-18->-          |       0.00
     10 |     11 | Sudarat Boonmee     |     20 | 2025-06-22->2025-06-27 |    9000.00
     11 |      1 | Harris Pemberton    |      9 | 2025-10-02->-          |       0.00
```

ภาพที่ 3-32 การเลือกใช้งานดู contract

3.5.6 กรอกหมายเลข 3 เพื่อเข้าเมนู กรอง เพื่อเข้าเมนูเลือกชนิดข้อมูล customer, car, contract

```
[View] 1) เดี่ยว 2) ทั้งหมด 3) กรอง 4) สถิติ  0) Back
เลือก: 3
ชนิด (customer/car/contract, 0=Back):
```

ภาพที่ 3-33 การเลือกใช้งานเมนูกรอง

3.5.7 กรอก customer เพื่อเข้าเมนู customer เพื่อค้นหาข้อมูลลูกค้าทั้งหมดที่ระบุและจากนั้นใส่ข้อมูลในหัวข้อดังภาพที่ 3-34

```
[View] 1) เดี่ยว 2) ทั้งหมด 3) กรอง 4) สถิติ  0) Back
เลือก: 3
ชนิด (customer/car/contract, 0=Back): customer
ค้นหาชื่อ: S
   2 | Sudarat Inthra
   3 | Sudarat Prasert
   5 | Somchai Srisuk
   6 | Kittisak Chaiyakul
   7 | Siriporn Chanthara
   9 | Siriporn Prasert
  10 | Sudarat Prasert
  11 | Sudarat Boonmee
  12 | Sudarat Inthra
  13 | Sudarat Prasert
  15 | Somchai Srisuk
  16 | Kittisak Chaiyakul
  17 | Siriporn Chanthara
  19 | Siriporn Prasert
  20 | Sudarat Prasert
```

ภาพที่ 3-34 การเลือกใช้งานดู customer

3.5.8 กรอก car เพื่อเข้าเมนู car เพื่อค้นหาข้อมูลรถยนต์ทั้งหมดที่ระบุและจากนั้นใส่ข้อมูลสถานะดูในหัวข้อดังภาพที่ 3-35

```
[View] 1) เดี่ยว 2) ทั้งหมด 3) กรอง 4) สถิติ  0) Back
เลือก: 3
ชนิด (customer/car/contract, 0=Back): car
สถานะ (available/rented/maintenance/retired หรือเว้นว่าง):
   2 | TH-0002    | Mazda      | CX-30      | 2018 | 2000.00    | available
   3 | TH-0003    | Honda      | Jazz       | 2020 | 1200.00    | available
   4 | TH-0004    | Nissan     | Note       | 2025 | 1200.00    | rented
   5 | TH-0005    | Mazda      | 2          | 2020 | 900.00     | available
   6 | TH-0006    | Mazda      | 3          | 2021 | 900.00     | available
   7 | TH-0007    | Mitsu      | Xpander    | 2022 | 1200.00    | available
   8 | TH-0008    | Nissan     | Note       | 2024 | 1200.00    | rented
   9 | TH-0009    | Honda      | City       | 2025 | 2000.00    | rented
  10 | TH-0010    | Mitsu      | Mirage     | 2026 | 1800.00    | available
  11 | TH-0011    | Mazda      | 2          | 2022 | 1500.00    | available
  12 | TH-0012    | Mazda      | CX-30      | 2018 | 2000.00    | available
  13 | TH-0013    | Honda      | Jazz       | 2020 | 1200.00    | available
  14 | TH-0014    | Nissan     | Note       | 2025 | 1200.00    | rented
  15 | TH-0015    | Mazda      | 2          | 2020 | 900.00     | available
  16 | TH-0016    | Mazda      | 3          | 2021 | 900.00     | available
  17 | TH-0017    | Mitsu      | Xpander    | 2022 | 1200.00    | available
  18 | TH-0018    | Nissan     | Note       | 2024 | 1200.00    | rented
  19 | TH-0019    | Honda      | City       | 2025 | 2000.00    | available
  20 | TH-0020    | Mitsu      | Mirage     | 2026 | 1800.00    | available
```

ภาพที่ 3-35 การเลือกใช้งานดู car

3.5.8 กรอก contract เพื่อเข้าเมนู contract เพื่อค้นหาข้อมูลผู้เช่ารถยนต์ทั้งหมดที่ระบุ และจากนั้นใส่ข้อมูลวันเวลาดูในหัวข้อดังภาพที่ 3-36

```
[View] 1) เดี่ยว 2) ทั้งหมด 3) กรอง 4) สถิติ  0) Back
เลือก: 3
ชนิด (customer/car/contract, 0=Back): contract
ช่วง FROM,TO (YYYY-MM-DD,YYYY-MM-DD): 2025-07-12,2025-10-12
   2 | 2025-07-22
   3 | 2025-09-13
   4 | 2025-08-18
   6 | 2025-09-03
   7 | 2025-07-22
   8 | 2025-09-13
   9 | 2025-08-18
  11 | 2025-10-02
```

ภาพที่ 3-36 การเลือกใช้งานดู contract

3.5.9 กรอกหมายเลข 4 เพื่อเข้าเมนู สถิติ เพื่อดูสถานะของการเช่ารถยนต์

```
[View] 1) เดี่ยว 2) ทั้งหมด 3) กรอง 4) สถิติ  0) Back
เลือก: 4
Cars by status:
  available = 14
  rented = 5
  maintenance = 0
  retired = 0
Open contracts = 5
```

ภาพที่ 3-37 การเลือกใช้งานเมนูสถิติ

บทที่ 4

อธิบายการทำงานของ Code

4.1 ไลบารีพื้นฐานในระบบเช่ารถ

4.1.1 from __future__ import annotations คำสั่งนี้เปิดโหมด “เลื่อนการตีความ type hints” ทำให้ค่าภายใน annotation ถูกเก็บเป็นสตริงไปก่อน แล้วค่อยประเมินตอนรัน ช่วยให้เขียน type ที่อ้างถึงคลาสซึ่งประกาศทีหลัง (forward reference) ได้สะดวก ลดปัญหา import วนลูป และทำให้โค้ดกำหนดชนิดข้อมูลได้ครบถ้วนโดยไม่ต้องย้ายลำดับนิยามคลาส/ฟังก์ชัน

```
from __future__ import annotations
```

ภาพที่ 4-1 module annotations

4.1.2 os ให้เครื่องมือคุยกับระบบไฟล์และระบบปฏิบัติการ เช่น ตรวจ/สร้างโฟลเดอร์ (os.path.isdir, os.makedirs), รวมพาธ (os.path.join), และที่สำคัญคือ os.fsync() เพื่อบังคับเขียนบัฟเฟอร์ลงดิสก์จริงหลังอัปเดตไฟล์ไบนารี (header/index/record) ลดความเสี่ยงข้อมูลสูญหายเมื่อไฟดับหรือโปรแกรมปิดกะทันหัน

```
import os
```

ภาพที่ 4-2 module os

4.1.3 sys ใช้จัดการเรื่องระดับตัวแปลรันไทม์ของ Python เช่น การออกจากโปรแกรมด้วยสถานะ (sys.exit(code)), อ่านอาร์กิวเมนต์ดิบจากบรรทัดคำสั่ง, และพิมพ์ข้อความไป stderr ได้เมื่อเกิดข้อผิดพลาด ตอนจบโปรแกรมไฟล์นี้จะเรียก sys.exit(main()) เพื่อส่งรหัสสถานะสรุปการทำงาน

```
import sys
```

ภาพที่ 4-3 module sys

4.1.4 struct คือหัวใจของการทำ “ไฟล์ไบนารีแบบระเบียนคงที่” (fixed-length records) เพราะใช้แปลงข้อมูล Python (int, bytes, ฯลฯ) ไปเป็นบล็อกไบต์ตามรูปแบบที่กำหนด และระบุ endianness ได้ชัดเจน (โปรเจ็กต์นี้ใช้ < = little-endian เสมอ) ทั้งส่วน Header 128B, Index 16B/ช่อง และเรคคอร์ดของ Customers/Cars/Contracts ล้วน pack/unpack ด้วย struct เพื่อให้ทุกเรคคอร์ดมีขนาดเท่ากัน อ่าน/เขียนแบบสุ่มตำแหน่งได้แม่นยำ

```
import struct
```

ภาพที่ 4-4 module struct

4.1.5 argparse ทำให้รับอาร์กิวเมนต์จากบรรทัดคำสั่งแบบเป็นระบบ เช่น --data-dir เพื่อกำหนดโฟลเดอร์เก็บ .bin/.txt พร้อมข้อความช่วยเหลืออัตโนมัติ (-h). ช่วยให้สคริปต์ตัวเดียวปรับที่เก็บข้อมูลได้ยืดหยุ่นเวลานำไปใช้งานจริงหรือทดสอบ

```
import argparse
```

ภาพที่ 4-5 module argparse

4.1.6 @dataclass ใช้นิยามคลาสข้อมูล (เช่น Header, IndexSlot) ให้สั้นและชัด โดยสร้าง __init__, __repr__ ให้อัตโนมัติ โค้ดจึงอ่านง่ายและลดข้อผิดพลาดเวลาแพ็ก/แกะข้อมูลจาก struct เพราะฟิลด์ต่างๆ ของ header/index ถูกประกาศเป็นชื่อชัดเจนในคลาสเดียวกัน

```
from dataclasses import dataclass
```

ภาพที่ 4-6 module dataclass

4.1.7 datetime, date กลุ่มนี้จัดการวันเวลา เช่นการสร้าง timestamp ปัจจุบัน (datetime.now().timestamp()), แปลง/จัดรูปแบบวัน (YYYY-MM-DD ⟷ YYYYMMDD) และคำนวณจำนวนวันเช่ากับคืนรถ (อย่างน้อย 1 วัน). รายงาน report.txt ก็แทรกวัน-เวลาที่สร้างด้วย datetime.now().astimezone() เพื่อบอกโซนเวลา

```
from datetime import datetime, date
```

ภาพที่ 4-7 datetime

4.1.8 typing ใช้ใส่ type hints ให้ฟังก์ชัน/ตัวแปร เช่น Optional[bytes] (อาจไม่มีค่า), Iterable[Tuple[int, bytes]] (ตัววนเรคคอร์ด), หรือ Dict[int, Any] (cache รายชื่อ). แม้ไม่กระทบผลลัพธ์ตอนรันจริง แต่ช่วยรีวิว/ดูแลโค้ดและลดบั๊กระยะยาว

```
from typing import Optional, Iterable, Tuple, Dict, Any
```

ภาพที่ 4-8 module typing

4.2 ฟังก์ชั่นเมนูระบบเช่ารถ

4.2.1 def run คือวงลูปหลักของโปรแกรมที่ควบคุม “หน้าเมนู” ทั้งหมด: เริ่มจากพิมพ์เมนูหลัก 1) Add 2) Update 3) Delete 4) View 5) Report 0) Exit แล้วรอผู้ใช้กรอกตัวเลือกที่บรรทัดเลือก: (ถ้ากด Enter เปล่า ๆ จะตีความเป็น 0) จากนั้นสวิตช์ไปยังเมนูย่อยด้วยลูปย่อยแยกตามหมวด — Add/Update/Delete/View—ซึ่งแต่ละอันมีตัวเลือกงานย่อย (เช่น Add: Customer/Car/Contract, Update: Customer/Car/Return Car) และปุ่ม 0/Back เพื่อย้อนกลับสู่เมนูหลักได้ตลอด; ตัวเลือก 5 จะเรียก generate_report() เพื่อเขียนรายงานสรุปลง report.txt ทันที ส่วนตัวเลือก 0 จะสร้างรายงานครั้งสุดท้าย, พิมพ์ “บันทึกและออก...”, ปิดไฟล์ทั้งหมด แล้วจบโปรแกรม โดยรอบการทำงานถูกห่อด้วย try/except เพื่อจับข้อผิดพลาด แสดงข้อความเตือน และวนกลับเมนูโดยไม่ให้โปรแกรมล้ม.

```
def run(self):
    while True:
        print("\n===== CarRent-BinIO =====")
        print("1) Add  2) Update  3) Delete  4) View  5) Report  0) Exit")
        c = (input('เลือก: ')).strip()
        try:
            if c == '1':
                # --- Add submenu ---
                while True:
                    print("\n[Add] 1) Customer 2) Car 3) Contract  0) Back")
                    ch = input('เลือก: ').strip().lower()
                    if ch in ('0', 'b', 'back'): break
                    {'1': self.add_customer,
                    '2': self.add_car,
                    '3': self.add_contract}.get(ch, lambda: print('ตัวเลือกไม่ถูกต้อง'))()
            elif c == '2':
                # --- Update submenu ---
                while True:
                    print("\n[Update] 1) Customer 2) Car 3) Return Car  0) Back")
                    ch = input('เลือก: ').strip().lower()
                    if ch in ('0', 'b', 'back'): break
                    {'1': self.update_customer,
                    '2': self.update_car,
                    '3': self.return_car}.get(ch, lambda: print('ตัวเลือกไม่ถูกต้อง'))()
            elif c == '3':
                # --- Delete submenu ---
                while True:
                    print("\n[Delete] 1) Customer 2) Car 3) Contract  0) Back")
                    ch = input('เลือก: ').strip().lower()
                    if ch in ('0', 'b', 'back'): break
                    {'1': self.delete_customer,
                    '2': self.delete_car,
                    '3': self.delete_contract}.get(ch, lambda: print('ตัวเลือกไม่ถูกต้อง'))()
            elif c == '4':
                while True:
                    print("\n[View] 1) เดี่ยว 2) ทั้งหมด 3) กรอง 4) สถิติ  0) Back")
                    ch = input('เลือก: ').strip().lower()
                    if ch in ('0', 'b', 'back'): break
                    {'1': self.view_single,
                    '2': self.view_all,
                    '3': self.view_filter,
                    '4': self.view_stats}.get(ch, lambda: print('ตัวเลือกไม่ถูกต้อง'))()
            elif c == '5':
                out = os.path.join(os.path.dirname(self.customers.path), 'report.txt')
                self.generate_report(out)

            elif c == '0':
                out = os.path.join(os.path.dirname(self.customers.path), 'report.txt')
                self.generate_report(out)
                print('บันทึกและออก...')
                self.close()
                break
            else:
                print('ตัวเลือกไม่ถูกต้อง')

        except Exception as e:
            print('! error:', e)
```

ภาพที่ 4-9 code menu

4.3 ฟังค์ชันเพิ่มข้อมูล

4.3.1 ฟังก์ชัน: add_customer

ฟังก์ชันนี้เพิ่มลูกค้าใหม่ลงไฟล์ customers.bin ในรูปแบบไบนารีแบบระเบียนคงที่ โดยรับข้อมูลชื่อ เลขบัตร 13 หลัก เบอร์โทร วันเกิด และเพศ จากนั้นตรวจรูปแบบ (เลขบัตร/โทร) และแปลงวันเกิดจาก YYYY-MM-DD เป็นเลข YYYYMMDD. ระบบขอหมายเลข cus_id ใหม่จาก header (next_id) แล้วจัดรูปข้อมูลด้วย struct.pack() (ตัด/เติม \x00 ให้ช่องข้อความด้วย fit) ก่อนเขียนเรคคอร์ดลงตำแหน่งที่จัดสรร (รีไซเคิลจาก free-list หากมี) พร้อมอัปเดตดัชนีและตัวนับใน header.

```
def add_customer(self):
    name = input('ชื่อ-นามสกุล: ').strip()
    idc  = input('เลขบัตร 13 หลัก: ').strip()
    phone= input('โทร (9-10 หลัก): ').strip()
    dob  = input('วันเกิด YYYY-MM-DD: ').strip()
    gopt = (input('เพศ (unk/male/female) [unk]: ').strip() or 'unk').lower()
    gender = {'unk':0,'male':1,'female':2}.get(gopt,0)
    if not name or not is_idcard(idc) or not is_phone(phone):
        print('! ข้อมูลไม่ถูกต้อง'); return
    cid = self.customers.next_id()
    rec = self.customers.pack(1, cid, idc, name, phone, ymd_to_int(dob), gender)
    self.customers.add_record(cid, rec); print(f'+ เพิ่มลูกค้า id={cid}')
```

ภาพที่ 4-10 add customer

4.3.2 ฟังก์ชัน: add_car

หน้าที่คือบันทึกรถคันใหม่ลง cars.bin โดยรับทะเบียน ยี่ห้อ รุ่น ปีผลิต ค่าเช่าต่อวัน (บาท) เลขไมล์ และสถานะ ตรวจความถูกต้อง (ทะเบียน/ปี/ตัวเลขไม่ติดลบ) แล้วแปลงค่าเช่าจากบาทเป็น “สตางค์” เพื่อเก็บเป็นจำนวนเต็ม (เลี่ยงปัญหาทศนิยม). จากนั้นออก car_id ใหม่ จัดแพ็กเรคคอร์ดด้วย struct พร้อม updated_at เป็นเวลาปัจจุบัน และเพิ่ม entry ใน index ให้ค้นได้เร็ว

```
def add_car(self):
    plate = input('ทะเบียน (<=16): ').strip()
    brand = input('ยี่ห้อ (<=12): ').strip()
    model = input('รุ่น (<=16): ').strip()
    try:
        year  = int(input('ปีผลิต: ').strip())
        rate  = float(input('ค่าเช่าต่อวัน (บาท): ').strip())
        odo   = int(input('เลขไมล์ (กม.): ').strip())
    except Exception:
        print('! ตัวเลขไม่ถูกต้อง'); return
    stat  = CAR_STATUS_REV.get((input('สถานะ [available]: ').strip() or 'available'), 0)
    if not is_plate(plate) or not is_year(year) or rate < 0 or odo < 0:
        print('! ข้อมูลไม่ถูกต้อง'); return
    car_id = self.cars.next_id(); rec = self.cars.pack(1, car_id, plate, brand, model, year, int(round(rate*100)), odo, stat, now_ts())
    self.cars.add_record(car_id, rec); print(f'+ เพิ่มรถ id={car_id}')
```

ภาพที่ 4-11 add car

4.3.3 ฟังก์ชัน: add_contract

ฟังก์ชันนี้เปิดสัญญาเช่ารถ โดยตรวจว่ามีลูกค้าตาม cus_id และรถตาม car_id จริงและรถต้องมีสถานะ available. วันเช่าถูกตั้งเป็น “วันนี้” อัตโนมัติ (แปลงเป็น YYYYMMDD). ระบบออก rent_id ใหม่และเขียนเรคคอร์ดสัญญา (returned=0, total_cents=0). ท้ายสุดอัปเดตรถคันนั้นเป็นสถานะ rented เพื่อรักษาความสอดคล้องระหว่างรถกับสัญญา

```
def add_contract(self):
    try:
        cus_id = int(input('cus_id: '))
        car_id = int(input('car_id: '))
        # rent = ymd_to_int(input('วันที่เช่า YYYY-MM-DD: ').strip())
        today = datetime.now()
        rent = today.year*10000 + today.month*100 + today.day
    except Exception:
        print('! อินพุตไม่ถูกต้อง'); return
    car_raw = self.cars.read_record(car_id)
    if not car_raw: print('! ไม่พบรถ'); return
    car = self.cars.unpack(car_raw)
    if car['status'] != 0: print('! รถไม่ว่าง'); return
    if not self.customers.read_record(cus_id): print('! ไม่พบลูกค้า'); return
    rid = self.contracts.next_id()
    self.contracts.add_record(rid, self.contracts.pack(1, rid, cus_id, car_id, rent, 0, 0, 0))
    self.cars.update_record(car_id, self.cars.pack(1, car['car_id'], car['license'], car['brand'], car['model'],
                                                   car['year'], car['rate_cents'], car['odometer_km'], 1, now_ts()))
    print(f'+ เปิดสัญญา rent_id={rid}')
```

ภาพที่ 4-12 add car

4.4 ฟังค์ชันอัปเดตข้อมูล

4.4.1 ฟังก์ชัน: update_customer

ทำหน้าที่แก้ไขข้อมูลลูกค้าที่มีอยู่ โดยโหลดเรคคอร์ดเดิมมาเป็นค่าเริ่มต้น ให้ผู้ใช้กด Enter เพื่อคงค่าเดิมเป็นรายฟิลด์ ตรวจรูปแบบ (เลขบัตร/โทร) และแปลงวันเกิดถ้ามีการแก้ไข แล้ว pack ทับเรคคอร์ดเดิมที่ตำแหน่งเดิม (in-place) พร้อมอัปเดตเวลาใน header เพื่อบันทึกว่ามีการเปลี่ยนข้อมูลเกิดขึ้น

```
def update_customer(self):
    try: cid = int(input('cus_id: '))
    except Exception: print('! อินพุตไม่ถูกต้อง'); return
    raw = self.customers.read_record(cid)
    if not raw: print('! ไม่พบลูกค้า'); return
    r = self.customers.unpack(raw)
    name = input(f"ชื่อ [{r['name']}]: ").strip() or r['name']
    idc  = input(f"บัตร13 [{r['id_card']}]: ").strip() or r['id_card']
    phone= input(f"โทร [{r['phone']}]: ").strip() or r['phone']
    dob  = input(f"เกิด YYYY-MM-DD [{int_to_ymd(r['birth_ymd'])}]: ").strip()
    g    = input('เพศ (unk/male/female) [คงเดิม]: ').strip().lower()
    gender = {'unk':0,'male':1,'female':2}.get(g, r['gender'])
    if not name or not is_idcard(idc) or not is_phone(phone): print('! ข้อมูลไม่ถูกต้อง'); return
    self.customers.update_record(cid, self.customers.pack(1, cid, idc, name, phone, (r['birth_ymd'] if not dob else ymd_to_int(dob)), gender))
    print('* อัปเดตลูกค้าแล้ว')
```

ภาพที่ 4-13 update customer

4.4.2 ฟังก์ชัน: update_car

ใช้แก้ไขรายละเอียดรถ เช่น ทะเบียน ปีผลิต ค่าเช่า เลขไมล์ และสถานะ โดยตรวจความถูกต้องของค่าที่ป้อน อีกด้านหนึ่งมี “การ์ดธุรกิจ” ให้สถานะรถสอดคล้องกับสัญญาเปิดอยู่: ถ้าเปลี่ยนจาก rented เป็น available และคันนั้นมีสัญญาเปิด ระบบจะปิดสัญญาอัตโนมัติในวันปัจจุบันและคำนวณยอดรวมให้; เคสมีสัญญาเปิดแต่จะตั้งสถานะอื่นที่ไม่ใช่ rented หรือตั้ง rented โดยไม่มีสัญญา จะไม่อนุญาต. บันทึกเสร็จตั้ง updated_at ใหม่เสมอ

```
def update_car(self):
    try: car_id = int(input('car_id: '))
    except Exception: print('! อินพุตไม่ถูกต้อง'); return

    raw = self.cars.read_record(car_id)
    if not raw: print('! ไม่พบรถ'); return
    r = self.cars.unpack(raw)

    plate = input(f"ทะเบียน [{r['license']}]: ").strip() or r['license']
    brand = input(f"ยี่ห้อ [{r['brand']}]: ").strip() or r['brand']
    model = input(f"รุ่น [{r['model']}]: ").strip() or r['model']
    try:
        year  = int(input(f"ปีผลิต [{r['year']}]: ") or r['year'])
        rateb = float(input(f"ค่าเช่าบาท [{r['rate_cents']/100:.2f}]: ") or r['rate_cents']/100)
        odo   = int(input(f"เลขไมล์ [{r['odometer_km']}]: ") or r['odometer_km'])
    except Exception:
        print('! ตัวเลขไม่ถูกต้อง'); return

    stat  = CAR_STATUS_REV.get(
        input(f"สถานะ ({'/'.join(CAR_STATUS.values())}) [{CAR_STATUS[r['status']]}]: ").strip() or CAR_STATUS[r['status']],
        r['status']
    )

    old_status = r['status']
    new_status = stat

    # หา open contract ล่าสุดของรถคันนี้
    open_latest = None
    for _, rc in self.contracts.iter_active():
        cc = self.contracts.unpack(rc)
        if cc['car_id'] == car_id and cc['returned'] == 0:
            if (open_latest is None) or (cc['rent_ymd'] > open_latest['rent_ymd']):
                open_latest = cc

    if open_latest:
        if new_status == 0 and old_status == 1:
            # auto-close สัญญาด้วย "วันนี้" และใช้เรตรถเดิม
            t = datetime.now()
            ret = t.year*10000 + t.month*100 + t.day
            def to_dt(n:int): return date(n//10000, (n//100)%100, n%100)
            days = (to_dt(ret) - to_dt(open_latest['rent_ymd'])).days or 1
            total = days * r['rate_cents']
            self.contracts.update_record(
                open_latest['rent_id'],
                self.contracts.pack(
                    1, open_latest['rent_id'], open_latest['cus_id'], open_latest['car_id'],
                    open_latest['rent_ymd'], ret, total, 1
                )
            )
            print(f"* ปิดสัญญาอัตโนมัติ rent_id={open_latest['rent_id']} ยอด {total/100:.2f} บาท ({days} วัน)")
        elif new_status != 1:
            print('! รถคันนี้ยังมีสัญญาเช่าเปิดอยู่ ต้องคืนรถ (return_car) ก่อน หรือคงสถานะเป็น rented')
            return
    else:
        # ไม่มีสัญญาเปิดอยู่ แต่อยากตั้งเป็น rented -> ไม่อนุญาต
        if new_status == 1:
            print('! จะตั้งสถานะเป็น rented ต้องเปิดสัญญาก่อน (add_contract)')
            return

    # ---------- อัปเดตรถ ----------
    if not is_plate(plate) or not is_year(year) or rateb < 0 or odo < 0:
        print('! ข้อมูลไม่ถูกต้อง'); return

    self.cars.update_record(
        car_id,
        self.cars.pack(1, car_id, plate, brand, model, year, int(round(rateb*100)), odo, new_status, now_ts())
    )
    print('* อัปเดตรถแล้ว')
```

ภาพที่ 4-14 update car

4.4.3 ฟังก์ชัน: return_car

ฟังก์ชันคืนรถ/ปิดสัญญา โดยรับ rent_id และวันคืน (รองรับคำว่า today/t/now หรือเว้นว่าง = ใช้วันปัจจุบัน) จากนั้นตรวจว่าสัญญายังไม่ปิดและวันคืนต้องไม่ก่อนวันเช่า คำนวณจำนวนวันเช่า (อย่างน้อย 1) × rate_cents ของรถเพื่อได้ total_cents. อัปเดตเรคคอร์ดสัญญาให้ returned=1, กำหนด return_ymd และยอดรวม แล้วตั้งสถานะรถกลับเป็น available ในไฟล์รถ

```python
def return_car(self):
    try:
        rid = int(input('rent_id: '))
        ret_in = input('วันคืน YYYY-MM-DD หรือพิมพ์ today: ').strip().lower()
        if ret_in in ('today', 't', 'now', ''):
            t = datetime.now()
            ret = t.year*10000 + t.month*100 + t.day
        else:
            ret = ymd_to_int(ret_in)
    except Exception:
        print('! อินพุตไม่ถูกต้อง'); return

    raw = self.contracts.read_record(rid)
    if not raw:
        print('! ไม่พบสัญญา'); return
    r = self.contracts.unpack(raw)
    if r['returned'] == 1:
        print('! ปิดสัญญาแล้ว'); return
    if ret < r['rent_ymd']:
        print('! วันที่ผิด'); return

    # อ่านข้อมูลรถจากสัญญา
    car_raw = self.cars.read_record(r['car_id'])
    if not car_raw:
        print('! ไม่พบรถในสัญญา'); return
    car = self.cars.unpack(car_raw)

    # คำนวณจำนวนวัน (ขั้นต่ำ 1 วัน) และยอด
    def to_dt(n:int) -> date: return date(n//10000, (n//100)%100, n%100)
    days = (to_dt(ret) - to_dt(r['rent_ymd'])).days or 1
    total = days * car['rate_cents']

    # อัปเดตสัญญา -> ปิดสัญญา
    self.contracts.update_record(
        rid,
        self.contracts.pack(1, r['rent_id'], r['cus_id'], r['car_id'], r['rent_ymd'], ret, total, 1)
    )

    # อัปเดตรถ: ถ้าปัจจุบันเป็น rented(1) ให้กลับเป็น available(0) มิฉะนั้นคงสถานะเดิม
    new_status = 0 if car['status'] == 1 else car['status']
    self.cars.update_record(
        car['car_id'],
        self.cars.pack(
            1, car['car_id'], car['license'], car['brand'], car['model'],
            car['year'], car['rate_cents'], car['odometer_km'], new_status, now_ts()
        )
    )

    print(f"* ปิดสัญญาแล้ว ยอด {total/100:.2f} บาท ({days} วัน)")
```

ภาพที่ 4-15 return car

4.5 ฟังค์ชันลบข้อมูล

4.5.1 ฟังก์ชัน: delete_customer

ลบลูกค้าแบบ **soft delete** โดยไม่ตัดข้อมูลทิ้งถาวร: ระบบอ่านตำแหน่งเรคคอร์ด เปลี่ยน flag เป็น 0 แล้วเขียน “ตัวชี้ next_free” ลงช่วง padding เพื่อพ่วงเข้าหัว free-list, จากนั้นตอก tombstone ในช่อง index เดิมและปรับตัวนับใน header. วิธีนี้ช่วยรีไซเคิลช่องว่างเมื่อมีการเพิ่มข้อมูลครั้งต่อไป

```python
def delete_customer(self):
    try: cid = int(input('cus_id: '))
    except Exception: print('! อินพุตไม่ถูกต้อง'); return
    try: self.customers.delete_record(cid); print('- ลบลูกค้าแล้ว')
    except Exception as e: print('!',e)
```

ภาพที่ 4-16 delete customer

4.5.2 ฟังก์ชัน: delete_car

ลบรถแบบ soft delete เช่นเดียวกับลูกค้า แต่มีข้อกำหนดว่าถ้ารถกำลังเช่าอยู่ (status == rented) จะ ห้ามลบ เพื่อไม่ให้ข้อมูลสัญญาขาดความสอดคล้อง. เมื่อผ่านเงื่อนไขแล้วระบบจะตั้ง flag=0, พ่วง free-list, ทำ tombstone และอัปเดตตัวนับใน header

```
def delete_car(self):
    try: car_id = int(input('car_id: '))
    except Exception: print('! อินพุตไม่ถูกต้อง'); return
    raw = self.cars.read_record(car_id)
    if not raw: print('! ไม่พบรถ'); return
    if self.cars.unpack(raw)['status'] == 1:
        print('! รถกำลังเช่า ลบไม่ได้'); return
    try: self.cars.delete_record(car_id); print('- ลบรถแล้ว')
    except Exception as e: print('!',e)
```

ภาพที่ 4-17 delete car

4.5.3 ฟังก์ชัน: delete_contract

ลบสัญญาแบบ soft delete โดยทำขั้นตอนเดียวกับตารางอื่น ๆ (ตั้ง flag=0, พ่วง free-list, tombstone index, อัปเดตตัวนับ) ช่วยคงประวัติไว้ในไฟล์และลดการกระจายของดัชนีเมื่อมีการเพิ่มสัญญาใหม่

```
def delete_contract(self):
    try: rid = int(input('rent_id: '))
    except Exception: print('! อินพุตไม่ถูกต้อง'); return
    try: self.contracts.delete_record(rid); print('- ลบสัญญาแล้ว')
    except Exception as e: print('!',e)
```

ภาพที่ 4-18 delete contract

4.6 ฟังค์ชันดูข้อมูล

4.6.1 ฟังก์ชัน: view_single

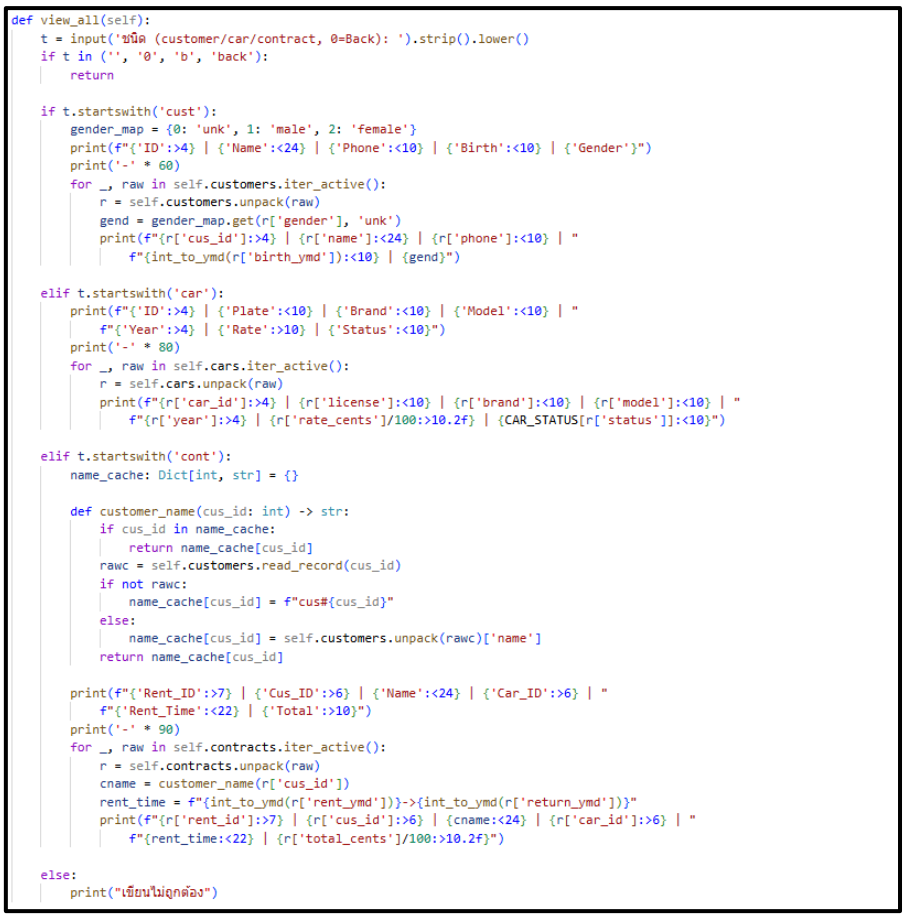
แสดงรายละเอียด “หนึ่งรายการ” ตามชนิดที่เลือก (customer/car/contract) และ id ที่ป้อนเข้าไป โดยอ่านเรคคอร์ดจากไฟล์ แปลงฟิลด์วันที่จาก YYYYMMDD เป็นรูปแบบอ่านง่าย YYYY-MM-DD แล้วพิมพ์สรุป ถ้าไม่พบจะแจ้งข้อความบอกผู้ใช้ทันที

```
def view_single(self):
    t = input('ชนิด (customer/car/contract, 0=Back): ').strip().lower()
    if t in ('', '0', 'b', 'back'):
        return
    try: i = int(input('id: '))
    except Exception: print('! อินพุตไม่ถูกต้อง'); return
    if t.startswith('cust'):
        raw = self.customers.read_record(i);
        if not raw: print('! ไม่พบ'); return
        r = self.customers.unpack(raw)
        print(f"[Customer] id={r['cus_id']} name={r['name']} id_card={r['id_card']} phone={r['phone']} "
              f"birth={int_to_ymd(r['birth_ymd'])} gender={r['gender']}")
    elif t.startswith('car'):
        raw = self.cars.read_record(i)
        if not raw: print('! ไม่พบ'); return
        r = self.cars.unpack(raw)
        print(f"[Car] id={r['car_id']} plate={r['license']} brand={r['brand']} model={r['model']} "
              f"year={r['year']} rate={r['rate_cents']/100:.2f} status={CAR_STATUS[r['status']]}")
    else:
        raw = self.contracts.read_record(i)
        if not raw: print('! ไม่พบ'); return
        r = self.contracts.unpack(raw)
        print(f"[Contract] id={r['rent_id']} cus_id={r['cus_id']} car_id={r['car_id']} rent={int_to_ymd(r['rent_ymd'])} "
              f"return={int_to_ymd(r['return_ymd'])} total={r['total_cents']/100:.2f} returned={r['returned']}")
```

ภาพที่ 4-19 single

4.6.2 ฟังก์ชัน: view_all

view_all รับชนิดที่ผู้ใช้ต้องการ (customer/car/contract) แล้วแสดง “รายการที่ยัง active” เป็นตารางอ่านง่าย พร้อมปุ่ม 0/Back ย้อนกลับได้. แบบ **customer** โชว์ cus_id, name, phone, birth (แปลงวันที่ด้วย int_to_ymd) และแม็ปเพศเป็น unk/male/female; แบบ **car** โชว์ car_id, plate, brand, model, year, rate(บาท/วัน) และ status; แบบ **contract** โชว์ rent_id, cus_id, name(ดึงจากแคช), car_id, rent_time (YYYY-MM-DD->YYYY-MM-DD) และ total (จัดเลขชิดขวา 2 ตำแหน่ง). ทุกตารางจัดหัวคอลัมน์และการจัดวางให้สม่ำเสมอ; ถ้าระบุชนิดไม่ถูกต้องจะแจ้งเตือนทันที.



```
def view_all(self):
    t = input('ชนิด (customer/car/contract, 0=Back): ').strip().lower()
    if t in ('', '0', 'b', 'back'):
        return

    if t.startswith('cust'):
        gender_map = {0: 'unk', 1: 'male', 2: 'female'}
        print(f"{'ID':>4} | {'Name':<24} | {'Phone':<10} | {'Birth':<10} | {'Gender'}")
        print('-' * 60)
        for _, raw in self.customers.iter_active():
            r = self.customers.unpack(raw)
            gend = gender_map.get(r['gender'], 'unk')
            print(f"{r['cus_id']:>4} | {r['name']:<24} | {r['phone']:<10} | "
                f"{int_to_ymd(r['birth_ymd']):<10} | {gend}")

    elif t.startswith('car'):
        print(f"{'ID':>4} | {'Plate':<10} | {'Brand':<10} | {'Model':<10} | "
            f"{'Year':>4} | {'Rate':>10} | {'Status':<10}")
        print('-' * 80)
        for _, raw in self.cars.iter_active():
            r = self.cars.unpack(raw)
            print(f"{r['car_id']:>4} | {r['license']:<10} | {r['brand']:<10} | {r['model']:<10} | "
                f"{r['year']:>4} | {r['rate_cents']/100:>10.2f} | {CAR_STATUS[r['status']]:<10}")

    elif t.startswith('cont'):
        name_cache: Dict[int, str] = {}

        def customer_name(cus_id: int) -> str:
            if cus_id in name_cache:
                return name_cache[cus_id]
            rawc = self.customers.read_record(cus_id)
            if not rawc:
                name_cache[cus_id] = f"cus#{cus_id}"
            else:
                name_cache[cus_id] = self.customers.unpack(rawc)['name']
            return name_cache[cus_id]

        print(f"{'Rent_ID':>7} | {'Cus_ID':>6} | {'Name':<24} | {'Car_ID':>6} | "
            f"{'Rent_Time':<22} | {'Total':>10}")
        print('-' * 90)
        for _, raw in self.contracts.iter_active():
            r = self.contracts.unpack(raw)
            cname = customer_name(r['cus_id'])
            rent_time = f"{int_to_ymd(r['rent_ymd'])}->{int_to_ymd(r['return_ymd'])}"
            print(f"{r['rent_id']:>7} | {r['cus_id']:>6} | {cname:<24} | {r['car_id']:>6} | "
                f"{rent_time:<22} | {r['total_cents']/100:>10.2f}")

    else:
        print("เขียนไม่ถูกต้อง")
```

ภาพที่ 4-20 view all

4.6.3 ฟังก์ชัน: view_filter

แสดงข้อมูลแบบมีเงื่อนไข: ฝั่งลูกค้าจะค้นจาก “ส่วนหนึ่งของชื่อ” แบบไม่สนตัวพิมพ์; ฝั่งรถกรองตามสถานะ (available/rented/maintenance/retired); ฝั่งสัญญากรองช่วงวันเช่าโดยรับ FROM,TO รูปแบบ YYYY-MM-DD,YYYY-MM-DD แล้วแปลงเป็นตัวเลข YYYYMMDD เพื่อตรวจช่วงก่อนพิมพ์เฉพาะเรคคอร์ดที่เข้าเงื่อนไข

```python
def view_filter(self):
    t = input('ชนิด (customer/car/contract, 0=Back): ').strip().lower()
    if t in ('', '0', 'b', 'back'):
        return
    if t.startswith('cust'):
        q = input('ค้นหาชื่อ: ').strip().lower()
        for _, raw in self.customers.iter_active():
            r = self.customers.unpack(raw)
            if q in r['name'].lower():
                print(f"{r['cus_id']:>4} | {r['name']}")

    elif t.startswith('car'):
        raw_in = input('สถานะ (available/rented/maintenance/retired หรือเว้นว่าง): ').strip().lower()
        st_code = None  # None = ไม่กรอง
        if raw_in:
            if raw_in.isdigit():
                v = int(raw_in)
                if v in CAR_STATUS:
                    st_code = v
            else:
                # ชื่อเต็มก่อน
                exact = [code for code, label in CAR_STATUS.items() if label == raw_in]
                if exact:
                    st_code = exact[0]
                else:
                    # prefix match (avai/ren/main/ret)
                    matched = [code for code, label in CAR_STATUS.items() if label.startswith(raw_in)]
                    if len(matched) == 1:
                        st_code = matched[0]
                    elif len(matched) > 1:
                        print("คำคันกำกวม: ", ', '.join(CAR_STATUS[m] for m in matched))
                        return
        for _, raw in self.cars.iter_active():
            r = self.cars.unpack(raw)
            if st_code is None or r['status'] == st_code:
                print(f"{r['car_id']:>4} | {r['license']:<10} | {r['brand']:<10} | {r['model']:<10} | "
                      f"{r['year']} | {r['rate_cents']/100:<10.2f} | {CAR_STATUS[r['status']]:<10}")
    elif t.startswith('cont'):
        try:
            a_str, b_str = input('ช่วง FROM,TO (YYYY-MM-DD,YYYY-MM-DD): ').split(',')
            a, b = ymd_to_int(a_str.strip()), ymd_to_int(b_str.strip())
        except Exception:
            print('รูปแบบวันที่ไม่ถูกต้อง'); return
        for _, raw in self.contracts.iter_active():
            r = self.contracts.unpack(raw)
            if a <= r['rent_ymd'] <= b:
                print(f"{r['rent_id']:>4} | {int_to_ymd(r['rent_ymd'])}")
    else:
        print("เขียนไม่ถูกต้อง")
```

ภาพที่ 4-21 view filter

4.6.4 ฟังก์ชัน: view_stats

สรุปสถิติสั้น ๆ ของระบบ โดยนับจำนวนรถแยกตามสถานะจาก cars.iter_active() และนับจำนวนสัญญาที่ “ยังเปิดอยู่” (returned == 0) เพื่อให้ผู้ใช้มองเห็นภาพรวมการใช้งานรถในระบบปัจจุบันได้รวดเร็ว

```python
def view_stats(self):
    cnt = {k:0 for k in CAR_STATUS}
    for _,raw in self.cars.iter_active():
        cnt[self.cars.unpack(raw)['status']] += 1
    print('Cars by status:')
    for k,v in cnt.items(): print(f"  {CAR_STATUS[k]} = {v}")
    open_ct = sum(1 for _,raw in self.contracts.iter_active() if self.contracts.unpack(raw)['returned']==0)
    print('Open contracts =', open_ct)
```

ภาพที่ 4-22 view stats

4.7 เมนูgenerate_report

4.7.1 ตัวช่วยวันที่/ช่วงเวลา

สรุปฟังก์ชันช่วย: fmd ฟอร์แมต YYYYMMDD ⟶ MM-DD, to_dt แปลงเลขเป็น date, และ days_between คำนวณจำนวนวัน (ถ้าอินพุตว่างให้ 0 แล้วค่อยบังคับขั้นต่ำเป็น 1 ตอนแสดงผล). เตรียมดิกแคชสองก้อนสำหรับชื่อผู้เช่าและข้อมูลรถ เพื่อลด I/O จากไฟล์.

```
def generate_report(self, out_path: str):
    fmd = lambda n: '-' if not n else f"{(n//100)%100:02d}-{n%100:02d}"
    to_dt = lambda n: date(n//10000, (n//100)%100, n%100)
    days_between = lambda a,b: 0 if (not a or not b) else (to_dt(b) - to_dt(a)).days

    name_cache, car_cache = {}, {}
```

ภาพที่ 4-23 code report ส่วนที่ 1

4.7.2 แคชข้อมูลลูกค้า/รถ

ดึงชื่อจาก customers.bin และข้อมูลรถจาก cars.bin ครั้งแรกแล้วเก็บในแคช; ถ้าไม่พบข้อมูลจะใส่ค่าดีฟอลต์ (cus#ID หรือ brand/model “Unknown”) เพื่อให้รายงานยังพิมพ์ต่อได้ไม่ล้ม.

```
def customer_name(cus_id: int) -> str:
    if cus_id not in name_cache:
        raw = self.customers.read_record(cus_id)
        name_cache[cus_id] = f"cus#{cus_id}" if not raw else self.customers.unpack(raw)['name']
    return name_cache[cus_id]

def car_info(car_id: int) -> dict:
    if car_id not in car_cache:
        raw = self.cars.read_record(car_id)
        car_cache[car_id] = {'license': f'car#{car_id}', 'brand': 'Unknown', 'model': 'Unknown', 'rate_cents': 0} \
                            if not raw else self.cars.unpack(raw)
    return car_cache[car_id]
```

ภาพที่ 4-24 code report ส่วนที่ 2

4.7.3 ส่วนหัวรายงาน

ประกอบหัวรายงานพร้อมเวลาปัจจุบัน (รวมโซนเวลา) และตั้งหัวข้อย่อย “รายการรถที่มีคนเช่า” ก่อนเข้าสู่ตาราง.

```
lines = []
ts = datetime.now().astimezone().strftime('%Y-%m-%d %H:%M:%S (%z)')
lines += [
    'Car Rent System — Rental Report',
    f'Generated At : {ts}',
    '',
    'รายการรถที่มีคนเช่า (ทั้งที่คืนแล้วและยังไม่คืน)',
    ''
]
```

ภาพที่ 4-25code report ส่วนที่ 3

4.7.4 รวบรวมและเตรียมตารางรายการเช่า

ดึง สัญญาที่มีการเช่าจริง (rent_ymd>0) จากเรคคอร์ด active, จัดเรียงตามวันเช่า แล้ววางหัวตาราง/เส้นคั่น พร้อมคำนวณ today_ymd ใช้กับรายการที่ยังไม่คืน.

```
rentals = [c for _, raw in self.contracts.iter_active()
        for c in [self.contracts.unpack(raw)] if c['rent_ymd'] > 0]

if not rentals:
    lines.append('ไม่มีรายการเช่า')
else:
    rentals.sort(key=lambda x: x['rent_ymd'])
    lines.append(f"{'Renter':<20} | {'Plate':<10} | {'Brand':<10} | {'Model':<12} | "
                 f"{'Rate':>8} | {'Rent Time':<13} | {'Days':>4} | {'Amount':>10} | {'Status':<8}")
    lines.append('-' * 120)

    total_amount = 0.0
    today = datetime.now(); today_ymd = today.year*10000 + today.month*100 + today.day
```

ภาพที่ 4-26 code report ส่วนที่ 4

4.7.5 เติมแถวตาราง + คิดยอดเงิน/สถานะ

สำหรับแต่ละสัญญา: สร้างช่วงเวลา start->end (ถ้าคืนแล้วใช้ return_ymd ไม่งั้นใช้ “วันนี้”), คิด days ขั้นต่ำ 1 วัน, จำนวนเงิน (ปิดแล้ว = ใช้ total_cents, ยังเช่า = days * rate_cents), ตั้งสถานะ “คืนแล้ว/เช่าอยู่” แล้วพิมพ์แถวพร้อมสะสมยอดรวม และพิมพ์จำนวนรายการและ “รวมยอดเงิน” (มีคอมมาและสองทศนิยม) ปิดท้ายบล็อกตารางเช่า.

```
for r in rentals:
    c = car_info(r['car_id']); cname = customer_name(r['cus_id'])
    start = fmd(r['rent_ymd'])
    if r['returned'] == 1 and r['return_ymd'] > 0:
        end = fmd(r['return_ymd']); rent_time = f"{start}->{end}"
        days = days_between(r['rent_ymd'], r['return_ymd']) or 1
        amount = r['total_cents'] / 100; status = "คืนแล้ว"
    else:
        end = fmd(today_ymd); rent_time = f"{start}->{end}"
        days = days_between(r['rent_ymd'], today_ymd) or 1
        amount = (days * c['rate_cents']) / 100; status = "เช่าอยู่"

    total_amount += amount
    lines.append(f"{cname:<20.20} | {c['license']:<10.10} | {c['brand']:<10.10} | "
                 f"{c['model']:<12.12} | {c['rate_cents']/100:>8.0f} | {rent_time:<13} | "
                 f"{days:>4} | {amount:>10.2f} | {status:<8}")

lines += ['', f'สรุป: {len(rentals)} รายการ', f'รวมยอดเงิน: {total_amount:,.2f} บาท']
```

ภาพที่ 4-27 code report ส่วนที่ 5

4.7.6 สรุปภาพรวมคลังรถ

นับจำนวนทั้งหมด/Active/Deleted และแจกแจงกำลังเช่า/ว่าง พร้อมนับจำนวนรถแยกตามยี่ห้อเฉพาะ Active เพื่อพิมพ์ในสรุป.

```
lines += ['', 'Summary (นับเฉพาะสถานะ Active)']
total = active = rented = avail = 0; by_brand = {}

for _, raw in self.cars.iter_all():
    total += 1
    c = self.cars.unpack(raw); is_active = (raw[0] == 1)
    if is_active:
        active += 1
        by_brand[c['brand']] = by_brand.get(c['brand'], 0) + 1
        if c['status'] == 1: rented += 1
        elif c['status'] == 0: avail += 1

deleted = total - active
lines += [
    f'- Total Cars (records) : {total}',
    f'- Active Cars          : {active}',
    f'- Deleted Cars         : {deleted}',
    f'- Currently Rented     : {rented}',
    f'- Available Now        : {avail}'
]
```

ภาพที่ 4-28 code report ส่วนที่ 6

4.7.7 Cars by Brand & เขียนไฟล์

พิมพ์จำนวนรถตามยี่ห้อ (เรียงชื่อยี่ห้อ) แล้วเขียน lines ทั้งหมดลงไฟล์ผลลัพธ์ พร้อมข้อความยืนยัน.

```
lines += ['', 'Cars by Brand (Active only)']
if by_brand:
    for b in sorted(by_brand): lines.append(f'- {b} : {by_brand[b]}')
else:
    lines.append('(no active cars)')

with open(out_path, 'w', encoding='utf-8') as f:
    f.write('\n'.join(lines) + '\n')
print('* เขียนรายงานที่', out_path)
```

ภาพที่ 4-29 code report ส่วนที่ 7

บทที่ 5
สรุปผลการดำเนินงานและข้อเสนอแนะ

5.1 สรุปผลการดำเนินงาน

ระบบจัดการเช่ารถด้วยการจัดเก็บข้อมูลแบบไฟล์ไบนารี (Binary File) โดยมีการออกแบบให้สามารถรองรับการเพิ่ม (Add) การแก้ไข (Update) การลบ (Delete) และการแสดงผลข้อมูล (View) รวมถึงการสร้างรายงาน (Report) ได้อย่างครบถ้วน ฐานข้อมูลหลักแบ่งออกเป็น 3 ส่วน ได้แก่ ข้อมูลลูกค้า ข้อมูลรถ และข้อมูลสัญญาเช่า ผลการดำเนินงานสามารถทำให้ผู้ใช้จัดการข้อมูลได้สะดวก ถูกต้อง และลดความซ้ำซ้อนของข้อมูล

5.2 ปัญหาและอุปสรรคในการดำเนินงาน

1. การจัดการรูปแบบไฟล์ไบนารีต้องอาศัยความระมัดระวัง เนื่องจากหากมีการอ่านหรือเขียนข้อมูลผิดพลาดอาจส่งผลกระทบต่อทั้งไฟล์

2. การตรวจสอบความถูกต้องของข้อมูลผู้ใช้ (เช่น การกรอกเลขบัตรประชาชน เบอร์โทรศัพท์ และวันที่) ต้องมีการเขียนฟังก์ชันตรวจสอบอย่างละเอียดเพื่อป้องกันข้อมูลผิดรูปแบบ

3. การเชื่อมโยงความสัมพันธ์ระหว่างตาราง (ลูกค้า-รถ-สัญญา) ต้องมีเงื่อนไขพิเศษ เช่น ไม่สามารถลบรถที่กำลังถูกเช่าได้ และการคืนรถต้องสอดคล้องกับสถานะของสัญญา

5.4 ข้อเสนอแนะ

1. ควรเพิ่มฟังก์ชันสำรองข้อมูล (Backup/Restore) เพื่อความปลอดภัยของข้อมูล

2. อาจพัฒนาระบบให้รองรับฐานข้อมูลเชิงสัมพันธ์ (เช่น SQLite หรือ MySQL) เพื่อการขยายระบบในอนาคต

3. เพิ่มอินเตอร์เฟซแบบกราฟิก (GUI) หรือเว็ปแอปพลิเคชั่น เพื่อให้ผู้ใช้ทั่วไปใช้งานได้สะดวกยิ่งขึ้น

4. พัฒนาโมดูลการวิเคราะห์ข้อมูล เช่น สถิติการเช่า ความนิยมของรถรุ่นต่างๆ เพื่อช่วยสนับสนุนการตัดสินใจทางธุรกิจ

5.5 สิ่งที่ผู้จัดทำได้รับในการพัฒนาโครงงาน

จากการพัฒนาโครงงานระบบจัดการเช่ารถ ผู้จัดทำได้รับความรู้และประสบการณ์ด้านการออกแบบระบบและการเขียนโปรแกรมด้วยภาษา Python โดยเฉพาะการจัดการข้อมูลด้วยไฟล์ไบนารีและโครงสร้างข้อมูลแบบ Fixed-Length Record ซึ่งต้องอาศัยความรอบคอบในการวางแผนและการเข้ารหัสข้อมูล นอกจากนี้ยังได้ฝึกการคิดวิเคราะห์และแก้ไขปัญหาเชิงตรรกะ เพื่อทำให้โปรแกรมสามารถทำงานได้ถูกต้องตามที่ออกแบบไว้ อีกทั้งยังได้ฝึกทักษะการทำงานเป็นระบบ การจัดสรรเวลาให้เหมาะสมกับแผนงาน รวมถึงการทำงานร่วมกันในทีมและการแบ่งหน้าที่ความรับผิดชอบอย่างเหมาะสม ทำให้ผู้จัดทำมีความเข้าใจในกระบวนการพัฒนาระบบซอฟต์แวร์มากยิ่งขึ้น และสามารถนำความรู้และประสบการณ์ที่ได้รับไปประยุกต์ใช้ในการทำงานหรือโครงการอื่น ๆ ในอนาคตได้อย่างมั่นใจ