ระบบเช่ารถ

Car Rent System

นาย ปัณณทัต ทองคำชุม รหัสนักศึกษา 6806022610062 Sec.1

นาย ปุญญพัฒน์ดนัย มั่นคง รหัสนักศึกษา 6806022610119 Sec.1

นาย ปภังกร สุวรรณรัตน์ รหัสนักศึกษา 6806022610097 Sec.1

นาย อคิราภ์ สมัครจิตร์ รหัสนักศึกษา 6806022610186 Sec.1

โครงงานนี้เป็นส่วนหนึ่งของการศึกษาหลักสูตรวิศวกรรมศาสตรบัณฑิต

สาขาวิศวกรรมสารสนเทศและเครือข่าย ภาควิชาเทคโนโลยีสารสนเทศ

คณะเทคโนโลยีและการจัดการอุตสาหกรรม มหาวิทยาลัยเทคโนโลยีพระจอมเกล้าพระนครเหนือ

ปีการศึกษา 2568

ลิขสิทธิ์ของมหาวิทยาลัยเทคโนโลยีพระจอมเกล้าพระนครเหนือ

คำนำ

การจัดทำโครงงาน “ระบบเช่ารถ” นี้เป็นส่วนหนึ่งของวิชา Computer Programming ของหลักสูตรวิศวกรรมศาสตรบัณฑิต สาขาวิศวกรรมสารสนเทศและเครือข่าย ภาควิชาเทคโนโลยีสารสนเทศ คณะเทคโนโลยีและการจัดการอุตสาหกรรม มหาวิทยาลัยเทคโนโลยี พระจอมเกล้าพระนครเหนือ เพื่อให้นักศึกษาได้นำความรู้ที่เรียนมาทั้งหมดมาประยุกต์ใช้ในการ พัฒนาโปรแกรมที่สามารถทำงานได้จริง โดยเน้นการออกแบบและเขียนโปรแกรมด้วยภาษา Python ซึ่งเป็นภาษาที่เรียนมาในวิชา Computer Programing โดยโครงงานนี้จะช่วยการคิดวิเคราะห์และ การแก้ปัญหาทางเทคนิค เพื่อเตรียมความพร้อมในการประกอบอาชีพด้านวิศวกรรมสารสนเทศและ เครือข่ายในอนาคต

คณะผู้จัดทำหวังว่า รายงานฉบับนี้จะเป็นประโยชน์กับผู้อ่าน หรือนักเรียน นักศึกษา ที่กำลังหาข้อมูลเรื่องนี้อยู่ หากมีข้อแนะนำหรือข้อผิดพลาดประการใด ผู้จัดทำขอน้อมรับไว้และขออภัยมา ณ ที่นี้ด้วย

สารบัญ

	หน้า
คำนำ	ก
สารบัญ	ข
สารบัญ(ต่อ)	ค
สารบัญรูปภาพ	ง
สารบัญรูปภาพ(ต่อ)	จ
สารบัญรูปภาพ(ต่อ)	ฉ
สารบัญรูปภาพ(ต่อ)	ช
สารบัญตาราง	ซ
บทที่ 1 บทนำ	1
1.1 วัตถุประสงค์ของโครงงาน	1
1.2 ขอบเขตของโครงงาน	1
1.3 ประโยชน์ที่ได้รับ	2
1.4 เครื่องมือที่คาดว่าจะต้องใช้	2
บทที่ 2 ระบบเช่ารถ	3
2.1 แฟ้มข้อมูลรถ car.dat	3
2.2 แฟ้มข้อมูลลูกค้า customer.dat	5
2.3 แฟ้มข้อมูลเช่ารถ rentals.dat	6
2.4 ไฟล์ report.txt	8
บทที่ 3 การใช้งานระบบเช่ารถ	10
3.1 การใช้งานโปรแกรมระบบเช่ารถ	10
3.2 การใช้งานโปรแกรมเพิ่มข้อมูล	14
3.3 การใช้งานโปรแกรมแก้ไขข้อมูล	17
3.4 การใช้งานโปรแกรมลบข้อมูล	20
3.5 การใช้งานโปรแกรมดูข้อมูล	23
3.6 การใช้งานโปรแกรมสร้างรายงาน	32
3.7 การปิดใช้งานโปรแกรม	33

สารบัญ(ต่อ)

	หน้า
บทที่ 4 อธิบายการทำงานของ code	34
4.1 ฟังก์ชันไบนารีพื้นฐานในระบบเช่ารถ	34
4.2 ฟังก์ชัน add_customer()	36
4.3 ฟังก์ชัน add_car()	37
4.4 ฟังก์ชัน add_rental()	38
4.5 ฟังก์ชัน update_entity()	39
4.6 ฟังก์ชัน delete_entity()	42
4.7 ฟังก์ชัน view_one()	44
4.8 ฟังก์ชัน view_all()	45
4.9 ฟังก์ชัน view_stats()	47
4.10 ฟังก์ชัน generate_report()	48
4.11 ฟังกัน main()	50
บทที่ 5 สรุปผลการดำเนินงานและข้อเสนอแนะ	52
5.1 สรุปผลการดำเนินงาน	52
5.2 ปัญหาและอุปสรรคในการดำเนินงาน	52
5.3 ข้อเสนอแนะ	52
5.4 สิ่งที่ผู้จัดทำได้รับในการพัฒนาโครงงาน	53

สารบัญรูปภาพ

หน้า

รูปภาพที่ 2-1 ไฟล์ report 8

รูปภาพที่ 3-1 การเลือกใช้งานฟังก์ชั่น Add 10

รูปภาพที่ 3-2 เมนูของ Add 11

รูปภาพที่ 3-3 การเลือกใช้งานฟังก์ชั่น Update 11

รูปภาพที่ 3-4 เมนูของ Update 11

รูปภาพที่ 3-5 การเลือกใช้งานฟังก์ชั่น Delete 12

รูปภาพที่ 3-6 เมนูของ Delete 12

รูปภาพที่ 3-7 การเลือกใช้งานฟังก์ชั่น View 12

รูปภาพที่ 3-8 เมนูของ View 13

รูปภาพที่ 3-9 การเลือกใช้งานฟังก์ชั่น Generate Report 13

รูปภาพที่ 3-10 การเลือกใช้งานฟังก์ชั่น Exit 13

รูปภาพที่ 3-11 การเลือกใช้งานฟังก์ชั่น Add 14

รูปภาพที่ 3-12 การเลือกใช้งานเมนู Customer ของฟังก์ชัน Add 14

รูปภาพที่ 3-13 การเพิ่มข้อมูลลูกค้า 14

รูปภาพที่ 3-14 การเลือกใช้งานฟังก์ชั่น Add 15

รูปภาพที่ 3-15 การเลือกใช้งานเมนู Car ของฟังก์ชัน Add 15

รูปภาพที่ 3-16 การเพิ่มข้อมูลรถ 15

รูปภาพที่ 3-17 การเลือกใช้งานฟังก์ชั่น Add 16

รูปภาพที่ 3-18 การเลือกใช้งานเมนู Rental ของฟังก์ชัน Add 16

รูปภาพที่ 3-19 การเพิ่มข้อมูลการเช่า 16

รูปภาพที่ 3-20 การเลือกใช้งานฟังก์ชั่น Update 17

รูปภาพที่ 3-21 การเลือกใช้งานเมนู Customer ของฟังก์ชัน Update 17

รูปภาพที่ 3-22 การแก้ไขข้อมูลลูกค้า 17

รูปภาพที่ 3-23 การเลือกใช้งานฟังก์ชั่น Update 18

รูปภาพที่ 3-24 การเลือกใช้งานเมนู Car ของฟังก์ชัน Update 18

สารบัญรูปภาพ(ต่อ)

หน้า

รูปภาพที่ 3-25 การแก้ไขข้อมูลรถ 18

รูปภาพที่ 3-26 การเลือกใช้งานฟังก์ชั่น Update 19

รูปภาพที่ 3-27 การเลือกใช้งานเมนู Rental ของฟังก์ชัน Update 19

รูปภาพที่ 3-28 การแก้ไขข้อมูลการเช่า 19

รูปภาพที่ 3-29 การเลือกใช้งานฟังก์ชั่น Delete 20

รูปภาพที่ 3-30 การเลือกใช้งานเมนู Customer ของฟังก์ชัน Delete 20

รูปภาพที่ 3-31 การลบข้อมูลลูกค้า 20

รูปภาพที่ 3-32 การเลือกใช้งานฟังก์ชั่น Delete 21

รูปภาพที่ 3-33 การเลือกใช้งานเมนู Car ของฟังก์ชัน Delete 21

รูปภาพที่ 3-34 การลบข้อมูลรถ 21

รูปภาพที่ 3-35 การเลือกใช้งานฟังก์ชั่น Delete 22

รูปภาพที่ 3-36 การเลือกใช้งานเมนู Rental ของฟังก์ชัน Delete 22

รูปภาพที่ 3-37 การลบข้อมูลการเช่า 22

รูปภาพที่ 3-38 การเลือกใช้งานฟังก์ชั่น View 23

รูปภาพที่ 3-38 การเลือกใช้งานเมนู View one 23

รูปภาพที่ 3-39 การเลือกใช้งานเมนู Customer 23

รูปภาพที่ 3-40 การดูข้อมูลลูกค้า 24

รูปภาพที่ 3-41 การเลือกใช้งานฟังก์ชั่น View 24

รูปภาพที่ 3-42 การเลือกใช้งานเมนู View one 24

รูปภาพที่ 3-43 การเลือกใช้งานเมนู Customer 25

รูปภาพที่ 3-44 การดูข้อมูลรถ 25

รูปภาพที่ 3-45 การเลือกใช้งานฟังก์ชั่น View 25

รูปภาพที่ 3-46 การเลือกใช้งานเมนู View one 25

รูปภาพที่ 3-47 การเลือกใช้งานเมนู Rental 26

รูปภาพที่ 3-48 การดูข้อมูลการเช่า 26

รูปภาพที่ 3-49 การเลือกใช้งานฟังก์ชั่น View 26

สารบัญรูปภาพ(ต่อ)

หน้า

รูปภาพที่ 3-50 การเลือกใช้งานเมนู View all 27

รูปภาพที่ 3-51 การเลือกใช้งานเมนู Customer 27

รูปภาพที่ 3-52 การดูข้อมูลลูกค้า 27

รูปภาพที่ 3-53 การเลือกใช้งานฟังก์ชั่น View 28

รูปภาพที่ 3-54 การเลือกใช้งานเมนู View all 28

รูปภาพที่ 3-55 การเลือกใช้งานเมนู Car 28

รูปภาพที่ 3-56 การดูข้อมูลรถ 29

รูปภาพที่ 3-57 การเลือกใช้งานฟังก์ชั่น View 29

รูปภาพที่ 3-58 การเลือกใช้งานเมนู View all 29

รูปภาพที่ 3-59 การเลือกใช้งานเมนู Rental 30

รูปภาพที่ 3-60 การดูข้อมูลการเช่า 30

รูปภาพที่ 3-61 การเลือกใช้งานเมนู Summary stats 30

รูปภาพที่ 3-62 การดูข้อมูลสถิติสรุป 30

รูปภาพที่ 3-61 การเลือกใช้งานเมนู Back 31

รูปภาพที่ 3-62 การเลือกใช้งานฟังก์ชันสร้างรายงาน 32

รูปภาพที่ 3-63 ไฟล์ report.txt 32

รูปภาพที่ 3-64 การเลือกใช้งานฟังก์ชันปิดโปรแกรม 33

รูปภาพที่ 3-65 Exit Program 33

รูปภาพที่ 4-1 Code Module Struct 34

รูปภาพที่ 4-2 Code Module datetime , date 34

รูปภาพที่ 4-3 Code import os , sys , io 35

รูปภาพที่ 4-4 Code Customer 35

รูปภาพที่ 4-5 Code Car 35

รูปภาพที่ 4-6 Code Rent 35

รูปภาพที่ 4-7 add_customer 36

รูปภาพที่ 4-8 add_car 37

สารบัญรูปภาพ(ต่อ)

หน้า

รูปภาพที่ 4-9 add_rent 38

รูปภาพที่ 4-10 update_entity 41

รูปภาพที่ 4-11 delete_entity 43

รูปภาพที่ 4-12 view_one 44

รูปภาพที่ 4-13 view_all 46

รูปภาพที่ 4-14 view_stats 47

รูปภาพที่ 4-15 generate_report 49

รูปภาพที่ 4-16 main menu 51

สารบัญตาราง

หน้า

ตารางที่ 2.1 แฟ้มข้อมูลรถ 3

ตารางที่ 2.2 แฟ้มข้อมูลลูกค้า 5

ตารางที่ 2.3 แฟ้มข้อมูลเช่ารถ 6

บทที่ 1

บทนำ

1.1 วัตถุประสงค์ของโครงงาน

1.1.1 เพื่อพัฒนาระบบเช่ารถได้อย่างมีประสิทธิภาพ

1.1.2 เพื่อฝึกฝนทักษะการเขียนโปรแกรมด้วย Python

1.1.3 เพื่อเรียนรู้วิธีการจัดการข้อมูลและไฟล์

1.1.4 เพื่อเรียนรู้การทำางานร่วมกันเป็นทีม

1.2 ขอบเขตของโครงงาน

1.2.1 ระบบเช่ารถ มีฟังก์ชั่นพื้นฐานทั้งหมด 15 ฟังก์ชั่น เช่น

1. เพิ่มข้อมูลรถ
2. เพิ่มข้อมูลลูกค้า
3. เพิ่มข้อมูลการเช่า
4. แก้ไขข้อมูลรถ
5. แก้ไขข้อมูลลูกค้า
6. แก้ไขข้อมูลการเช่า
7. ลบข้อมูลรถ
8. ลบข้อมูลลูกค้า
9. ลบข้อมูลการเช่า
10. ดูรายการเดียว
11. ดูทั้งหมด
12. ดูแบบกรอง
13. ดูสถิติโดยสรุป
14. Generate Report (.txt)
15. ออกจากโปรแกรม

1.2.2 ระบบเช่ารถประกอบด้วย 4 ไฟล์ ได้แก่

1. แฟ้มข้อมูลรถ car.dat
2. แฟ้มข้อมูลลูกค้า customer.dat
3. แฟ้มข้อมูลเช่ารถ rentals.dat
4. ไฟล์ report.txt

1.2.3 ระบบเช่ารถมีการจัดเก็บข้อมูลไว้ใน Text File ชื่อ report ซึ่งมี รหัสการเช่า ชื่อผู้เช่า เบอร์โทรศัพท์ผู้เช่า หมายเลขทะเบียนรถ ยี่ห้อรถ รุ่นรถ ราคาเช่าต่อวัน วันยืมรถ วันคืนรถ รวมวันที่ยืมรถ และ ยอดรวมเงินค่าเช่ารถ

1.2.4 ระบบเช่ารถ จะมีเมนูเพื่อให้ผู้ใช้สามารถเลือก ดำเนินการได้

1.3 ประโยชน์ที่ได้รับ

1.3.1 พัฒนาระบบเช่ารถได้อย่างมีประสิทธิภาพ

1.3.2 พัฒนาทักษะการเขียนโปรแกรม

1.3.3 เรียนรู้การจัดการข้อมูลและไฟล์

1.3.4 เรียนรู้การทำงานร่วมกันเป็นทีม

1.4 เครื่องมือที่คาดว่าจะต้องใช้

1.4.1 โปรแกรม Visual Studio Code

1.4.2 Microsoft Office

บทที่ 2

ระบบเช่ารถ

2.1 แฟ้มข้อมูลรถ car.dat

แฟ้มข้อมูลรถประกอบด้วย 8 ฟิลด์หลัก ซึ่งแต่ละฟิลด์มีรายละเอียดและความสำคัญดังนี้

ฟิลด์	ชนิด	ขนาด(bytes)	ตัวอย่าง	คำอธิบาย
car_id	I	5	2001	อ้างอิงไปที่ car.dat
license_plate	12s	50	ขค 7678	เลขทะเบียนรถ
brand	12s	4	Honda	ชื่อยี่ห้อของรถ
model	16s	50	Civic	ชื่อรุ่นของรถ
year	I	4	2025	ปีที่ผลิตของรถ
rate (thb/day)	F	3	1,500	ราคาเช่าต่อวัน
status	I	4	1	1 = Active, 0 = Inactive
is_rented	I	4	0	1=ถูกเช่า, 0=ว่าง

ตารางที่ 2.1 แฟ้มข้อมูลรถ

2.1.1 car_id รหัสรถ

car_id เป็นรหัสรถที่ใช้ในการระบุรถแต่ละคันอย่างชัดเจนและไม่ซ้ำกัน ฟิลด์นี้ถูกสร้างขึ้นโดยระบบในรูปแบบของตัวเลข (integer) เช่น 1001, 1002, 1003 เป็นต้น การมี รหัสรถที่เป็นเอกลักษณ์นี้เป็นสิ่งจำเป็นเพื่อหลีกเลี่ยงความสับสนระหว่างรถหลายคัน และ ช่วยให้สามารถค้นหาและเรียกดูข้อมูลของรถได้อย่างแม่นยำและรวดเร็ว โดยเฉพาะในกรณีที่มี รถจำนวนมาก

2.1.2 license_plate เลขทะเบียนรถ

license_plate เป็นเลขทะเบียนรถใช้ในการระบุรถแต่ละคันอย่างชัดเจนและไม่ซ้ำกัน ฟิลด์นี้ถูกสร้างขึ้นโดยระบบในรูปแบบของตัวอักษร (String) เช่น ขค 7678 , กค 8569 เป็นต้น การมีเลขป้ายทะเบียนนี้เป็นสิ่งจำเป็นเพื่อหลีกเลี่ยงความสับสนระหว่างรถหลายคัน และ ช่วยให้สามารถค้นหาและเรียกดูข้อมูลของรถได้อย่างแม่นยำและรวดเร็ว โดยเฉพาะในกรณีที่มี รถจำนวนมาก

2.1.3 brand ยี่ห้อของรถ

brand เป็นยี่ห้อรถใช้ในการระบุรถแต่ละคันอย่างชัดเจนและไม่ซ้ำกัน ฟิลด์นี้ถูกสร้างขึ้นโดยระบบในรูปแบบของตัวอักษร (String) เช่น Honda , Toyota , Nissan เป็นต้น ใช้บ่งบอกถึงแหล่งที่มาและสร้างความแตกต่างจากคู่แข่ง มักมีเอกลักษณ์ด้านคุณภาพ เทคโนโลยี การออกแบบและภาพลักษณ์ ดังนั้น Brand ของรถไม่ใช่เพียงชื่อผู้ผลิต แต่ยังสะท้อนถึง คุณค่า ความน่าเชื่อถือและความชอบของผู้บริโภค อีกด้วย

2.1.4 model ชื่อรุ่นของรถ

model เป็นรุ่นรถใช้ในการระบุรถแต่ละคันอย่างชัดเจนและไม่ซ้ำกัน ฟิลด์นี้ถูกสร้างขึ้นโดยระบบในรูปแบบของข้อความ (String) เช่น Civic , Vios , Amela เป็นต้น คือรุ่นรถยนต์ที่ผู้ผลิตออกแบบมาเพื่อตอบสนองความต้องการเฉพาะของผู้บริโภคแต่ละกลุ่ม การเข้าใจความแตกต่างระหว่างสองคำนี้จึงมีความสำคัญ เพราะช่วยให้ผู้บริโภคเลือกซื้อรถยนต์ที่ตรงกับความต้องการและความคาดหวัง อีกทั้งยังเป็นปัจจัยที่ช่วยให้นักวิชาการหรือผู้ทำงานด้านการตลาดสามารถวิเคราะห์พฤติกรรมผู้บริโภคและกลยุทธ์การแข่งขันของบริษัทผู้ผลิตรถยนต์ได้ดียิ่งขึ้น

2.1.5 year ปีที่ผลิตรถ

year เป็นปีที่ผลิตรถใช้ในการระบุรถแต่ละคันว่าผลิตในปีไหน ฟิลด์นี้ถูกสร้างขึ้นโดยระบบในรูปแบบของตัวเลข (integer) เช่น 2021 , 2023 , 2025 เป็นต้น ปีของรถยนต์ไม่ใช่เพียงตัวเลขปีที่ระบุในสมุดทะเบียนเท่านั้น แต่ยังสะท้อนถึงอายุการใช้งาน ความก้าวหน้าของเทคโนโลยี มูลค่าในตลาด และความนิยมของรถยนต์ในช่วงเวลานั้น ๆ การทำความเข้าใจปี จึงมีความสำคัญต่อทั้งผู้ซื้อ ผู้ขาย และผู้วิจัยด้านอุตสาหกรรมยานยนต์

2.1.6 rate (thb/day) ราคาเช่ารถต่อวัน

rate (thb/day) เป็นค่าบริการที่บริษัทให้เช่ารถยนต์กำหนดขึ้นสำหรับการใช้รถยนต์หนึ่งวัน ฟิลด์นี้ถูกสร้างขึ้นโดยระบบในรูปแบบของตัวเลขทศนิยม (float) เช่น 1,100.00 , 1,500.00 เป็นต้น โดยมีหน่วยเป็นเงินบาทต่อวัน ซึ่งเป็นข้อมูลสำคัญสำหรับผู้บริโภคและผู้ประกอบการ เนื่องจากสะท้อนทั้ง ต้นทุนการให้บริการ และ ความคุ้มค่าที่ลูกค้าได้รับ โดยราคาจะขึ้นอยู่กับ ประเภทรถ แบรนด์ รุ่น ปีผลิต ระยะเวลาเช่า และบริการเสริม ซึ่งมีความสำคัญต่อการตัดสินใจของผู้บริโภค รวมถึงการวิเคราะห์เชิงธุรกิจในอุตสาหกรรมการเช่ารถ

2.1.7 status สถานะ

status คือ สถานะรถคันนั้นๆ ซึ่งฟิลด์นี้จะแสดงข้อมูลสถานะของรถแต่ละคัน ฟิลด์นี้เป็นประเภทข้อมูลตัวเลขจำนวนเต็ม (integer) เช่น 1 = Active (ใช้งานอยู่/ยังมีในระบบ) หรือ 0 = Inactive (ไม่ใช้งานแล้ว) การมีสถานะของรถในระบบมีความสำคัญอย่างยิ่ง เพราะ ใช้ในการดูสถานะของรถแต่ละคัน และทำการแก้ไขข้อมูลต่างๆ รถแต่ละคันจะมีสถานะ ตามที่ระบุในการใช้งานอยู่ หรือการถูกลบ และ ระบบจะใช้สถานะดังกล่าวในการแสดงผลข้อมูลที่ เกี่ยวข้องกับรถคันนั้น

2.1.8 is_rented แสดงสถานะว่ารถคันนี้ถูกยืมอยู่หรือไม่

เป็นฟิลด์จำนวนเต็ม (integer 4 bytes) ใช้บอกสถานะว่ารถคันนั้นหลังเหตุการณ์ยังคง ถูกยืมอยู่หรือไม่ เช่น 1 = กำลังถูกยืม 0 = ว่าง ฟิลด์นี้มีประโยชน์ในการตรวจสอบสภาพ ล่าสุดของรถ

2.2 แฟ้มข้อมูลลูกค้า customer.dat

แฟ้มข้อมูลรถประกอบด้วย 4 ฟิลด์หลัก ซึ่งแต่ละฟิลด์มีรายละเอียดและความสำคัญดังนี้

ฟิลด์	ชนิด	ขนาด	ตัวอย่าง
customer_id	I	4	1001
id_card	15s	15	76894235
name	255s	255	อคิราภ์ สมัครจิตร
tel	I	10	0961924162

ตารางที่ 2.2 แฟ้มข้อมูลลูกค้า

2.2.1 customer_id รหัสลูกค้า

customer_id เป็นรหัสที่ใช้ระบุลูกค้าแต่ละคนในระบบ ฟิลด์นี้เป็นตัวเลขจำนวนเต็ม (integer) มีความเป็นเอกลักษณ์ไม่ซ้ำกัน เช่น 1001 , 1002 การมีรหัสลูกค้าช่วยให้ระบบสามารถจัดการข้อมูลลูกค้าจำนวนมากได้อย่างถูกต้องและรวดเร็ว

2.2.2 id_card เลขประจำตัวใบขับขี่ของลูกค้า

Id_card เป็นเลขประจำตัวใบขับขี่ของลูกค้า ฟิลด์นี้เป็นข้อความ (String) มีความเป็นเอกลักษณ์ไม่ซ้ำกัน เช่น 45678965 , 78964524 การมีเลขประจำตัวใบขับขี่ของลูกค้าเพื่อดูสิทธิการอนุญาติในการขับขี่รถของลูกค้าที่ออกโดยรัฐ เพื่อป้องกันปัญหาด้านกฎหมายทั้งตัวลูกค้า และผู้ให้เช่า

2.2.3 name ชื่อลูกค้า

name เป็นฟิลด์ข้อความ (string) ความยาวสูงสุด 255 ตัวอักษร ใช้เก็บชื่อ–นามสกุลของลูกค้า เช่น "สมชาย ใจดี", "John Smith" เป็นต้น ฟิลด์นี้เป็นข้อมูลสำคัญเพื่อแสดงผลและยืนยันตัวตนของลูกค้า

2.2.4 tel เบอร์โทรศัพท์ลูกค้า

tel เป็นเบอร์ลูกค้า ฟิลด์นี้เป็นตัวเลขจำนวนเต็ม (integer) มีความเป็นเอกลักษณ์ไม่ซ้ำกัน เช่น 0961924162 , 0853177476 การมีเบอร์ลูกค้าช่วยให้สามารถติดต่อลูกค้าได้ และสามารถติดตามลูกค้าได้ตลอด

2.3 แฟ้มข้อมูลเช่ารถ rentals.dat

แฟ้มข้อมูลรถประกอบด้วย 8 ฟิลด์หลัก ซึ่งแต่ละฟิลด์มีรายละเอียดและความสำคัญดังนี้

ฟิลด์	ชนิด	ขนาด(bytes)	ตัวอย่าง	คำอธิบาย
rental_id	I	4	3001	รหัสการเช่า
car_id	I	4	2001	อ้างอิงไปที่ cars.dat
customer_id	F	4	1001	อ้างอิงไปที่ customer.dat
start_ts	I	8	29/09/2025	วันที่เริ่มเช่ารถ
end_ts	I	8	10/10/2025	วันที่คืนรถ
total_days	I	8	11	จำนวนวันที่เช่า
total_amount	F	6	11,000	ราคารวม
status	I	4	1	1=Open, 0=Closed, -1=Deleted

ตารางที่ 2.3 แฟ้มข้อมูลเช่ารถ

2.3.1 rental_id รหัสการเช่า

rental_id เป็นรหัสการเช่าที่ใช้ในการระบุการเช่าอย่างชัดเจนและไม่ซ้ำกัน ฟิลด์นี้ถูกสร้างขึ้นโดยระบบในรูปแบบของตัวเลข (integer) เช่น 3001, 3002, 3003 เป็นต้น การมีรหัสการเช่าที่เป็นเอกลักษณ์นี้เป็นสิ่งจำเป็นเพื่อหลีกเลี่ยงความสับสนระหว่างการเช่าหลายคัน และช่วยให้สามารถค้นหาและเรียกดูข้อมูลของการเช่าได้อย่างแม่นยำและรวดเร็ว โดยเฉพาะในกรณีที่มี การเช่าจำนวนมาก

2.3.2 car_id รหัสรถ

car_id เป็นรหัสรถที่ใช้ในการระบุรถแต่ละคันอย่างชัดเจนและไม่ซ้ำกัน ฟิลด์นี้ถูกสร้างขึ้นโดยระบบในรูปแบบของตัวเลข (integer) เช่น 1001, 1002, 1003 เป็นต้น การมี รหัสรถที่เป็นเอกลักษณ์นี้เป็นสิ่งจำเป็นเพื่อหลีกเลี่ยงความสับสนระหว่างรถหลายคัน และ ช่วยให้สามารถค้นหาและเรียกดูข้อมูลของรถได้อย่างแม่นยำและรวดเร็ว โดยเฉพาะในกรณีที่มี รถจำนวนมาก

2.3.3 customer_id รหัสลูกค้า

customer_id เป็นรหัสที่ใช้ระบุลูกค้าแต่ละคนในระบบ ฟิลด์นี้เป็นตัวเลขจำนวนเต็ม (integer) มีความเป็นเอกลักษณ์ไม่ซ้ำกัน เช่น 1001 , 1002 การมีรหัสลูกค้าช่วยให้ระบบสามารถจัดการข้อมูลลูกค้าจำนวนมากได้อย่างถูกต้องและรวดเร็ว

2.3.4 start_ts วันที่เริ่มเช่ารถ

เป็นฟิลด์ตัวเลข (integer) ใช้เก็บวันที่เช่ารถในรูปแบบ YYYY-MM DD เช่น 2025-09-30 ข้อมูลนี้ใช้ตรวจสอบว่ารถถูกเช่าไปตั้งแต่เมื่อใด

2.3.5 end_ts วันที่คืนรถ

เป็นฟิลด์ตัวเลข (integer) ใช้เก็บวันที่คืนรถในรูปแบบ YYYY-MM DD เช่น 2025-10-05 หากยังไม่ได้คืนค่าของฟิลด์นี้อาจเป็นค่าว่าง

2.3.6 total_days จำนวนวันที่เช่า

เป็นฟิลด์ตัวเลข (integer) ใช้เก็บจำนวนวันที่เช่า เช่น 5 , 11, 20 เป็นต้น เพื่อเป็นสรุปว่าลูกค้าเช่ารถไปจำนวนกี่วัน และจะนำไปคิดเพื่อคำนวณค่าเช่าทั้งหมด

2.3.7 total_amount ราคารวม

เป็นฟิลด์ตัวเลขทศนิยม (float) ใช้เก็บจำนวนเงินค่าเช่า เช่น 5,600 , 11,000 เป็นต้น เพื่อเป็นสรุปว่าลูกค้าจะต้องจ่ายค่าเช่ารถทั้งหมดเท่าไหร่

2.3.8 status สถานะ

status คือ สถานะการเช่านั้นๆ ซึ่งฟิลด์นี้จะแสดงข้อมูลสถานะของการเช่า ฟิลด์นี้เป็นประเภทข้อมูลตัวเลขจำนวนเต็ม (integer) เช่น 1 = open , 0 = closed และ -1 = deleted การมีสถานะของการเช่าในระบบมีความสำคัญอย่างยิ่ง เพราะ ใช้ในการดูสถานะของการเช่าของรถ และทำการแก้ไขข้อมูลต่างๆ

2.4 ไฟล์ report.txt

ไฟล์ report.txt ในระบบเช่ารถของคุณประกอบด้วย 9 ฟิลด์หลัก ซึ่งแต่ละฟิลด์มีรายละเอียดและความสำคัญดังนี้

```
Car Rent System - Summary Report
Generated At: 2025-10-02 14:10:03
App Version: 1.0
Endianness: Little-Endian
Encoding: UTF-8 (fixed-length)

Rental_ID | Customer name | Tel.       | License_plate | Brand  | Model | Rate  | Date Rent  | Return date | Rental Day | Total Price
-------------------------------------------------------------------------------------------------------------------------------------------
3001      | Alice         | 0987315278 | ขค 5679       | Toyota | Vios  | 1,100 | 10/01/2025 | 10/10/2025  | 10         | 11,000.00
3002      | Mix           | 0942847589 | กข 9999       | Honda  | Civic | 1,500 | 09/28/2025 | 10/02/2025  | 5          | 7,500.00
3003      | Auu           | 0831285679 | กง 2134       | Nissan | Amela | 1,300 | 10/02/2025 | 10/05/2025  | 4          | 5,200.00
3004      | Tea           | 0963278917 | บก 1234       | Mazda  | 3     | 1,100 | 10/02/2025 | 10/06/2025  | 5          | 5,500.00

--- Summary ---

Customers : 4
Cars      : 4 (Active 4, Inactive 0)
Rentals   : 4 (Open 4, Closed 0, Deleted 0)

Rate Statistics (Active cars only)
- Min Rate : 1,100
- Max Rate : 1,500
- Avg Rate : 1,250

Cars by Brand
- Honda : 1
- Mazda : 1
- Nissan : 1
- Toyota : 1
```

รูปภาพที่ 2-1 ไฟล์ report

2.4.1 header_text ส่วนหัวรายงาน

เป็นฟิลด์ข้อความ (string 100 bytes) ใช้เก็บข้อความส่วนหัวของรายงาน เช่น "Car Rent System – Summary Report" ฟิลด์นี้แสดงชื่อหรือวัตถุประสงค์ของรายงานเพื่อให้ผู้ใช้ เข้าใจว่าเป็นรายงานประเภทใด

2.4.2 generated_at วันที่และเวลาที่สร้างรายงาน (YYYY-MM-DD HH:MM)

เป็นฟิลด์ข้อความ (string 25 bytes) ใช้เก็บวันและเวลาที่รายงานถูกสร้างขึ้นในรูปแบบ YYYY-MM-DD HH:MM เช่น "2025-10-01 09:30" ฟิลด์นี้ช่วยในการติดตามและอ้างอิงว่าไฟล์รายงานถูกสร้างเมื่อใด

2.4.3 app_version เวอร์ชันโปรแกรม เช่น "1.0"

เป็นฟิลด์ข้อความ (string 10 bytes) ใช้เก็บหมายเลขเวอร์ชันของโปรแกรมที่สร้างรายงาน เช่น "1.0", "2.1.5" ฟิลด์นี้มีประโยชน์ในการตรวจสอบว่าไฟล์รายงานถูกสร้างด้วยเวอร์ชันของระบบใด

2.4.4 encoding การเข้ารหัสไฟล์

เป็นฟิลด์ข้อความ (string 20 bytes) ใช้ระบุรูปแบบการเข้ารหัสไฟล์ เช่น "UTF-8", "ISO 8859-1" เพื่อให้การอ่านไฟล์รายงานถูกต้องตรงกับภาษาที่ใช้งาน

2.4.5 car_table_header หัวตาราง

car_table_header เป็นฟิลด์ข้อความ (string 80 bytes) ใช้เก็บหัวตารางสำหรับ แสดงข้อมูลรถ เช่น "Rental_ID | Customer_Name | Tel. | License_plate | Brand | Model | Rate I Date Rent I Retrun Date I Rental Day I Total Price" เพื่อ กำหนดโครงสร้างของตารางในส่วนรายงาน

2.4.6 car_records ข้อมูลตาราง

เป็นฟิลด์ข้อความ (string 120 * N bytes, โดย N = จำนวน) ใช้เก็บข้อมูลรถ แต่ละเล่มในรูปแบบความยาวคงที่ (fixed-length record) เช่น รายการรหัสการเช่า, ชื่อลูกค้า, เบอร์โทรลูกค้า, ป้ายทะเบียน, ยี่ห้อ , รุ่นรถ , ราคาเช่าต่อวัน , วันที่เริ่มเช่า , วันที่คืนรถ , จำนวนวันท่า , จำนวนเงินค่าเช่าทั้งหมด

2.4.7 summary_section สรุปข้อมูล

เป็นฟิลด์ข้อความ (string 150 bytes) ใช้เก็บข้อมูลสรุปของระบบ เช่น Customer = จำนวนลูกค้าทั้งหมด , Car = จำนวนรถทั้งหมดในระบบ Rentals = จำนวนการเช่าทั้งหมดในระบบ

2.4.8 statistics_section สถิติราคา

เป็นฟิลด์ข้อความ (string 50 bytes) ใช้เก็บข้อมูลสถิติราคา เช่น Min Rate = ราคาต่ำสุดของราคาเช่า Max Rate = ราคาสูงสุดของราคาเช่า Avg Rate = เป็นราคาเฉลี่ยของราคาเช่า

2.4.9 car_by_brand จำนวนยี่ห้อของรถ

เป็นฟิลด์ข้อความ (string 50 bytes) ใช้เก็บข้อมูลของรุ่นรถที่มีอยู่ เช่น Honda 1 = รถในยี่ห้อของ honda มี 1 คัน , Nissan 1 = รถในยี่ห้อของ Nissan มี 1 คัน เป็นต้น เพื่อสรุปจำนวนรถในยี่ห้อนั้นๆ ว่ามีกี่คัน

บทที่ 3

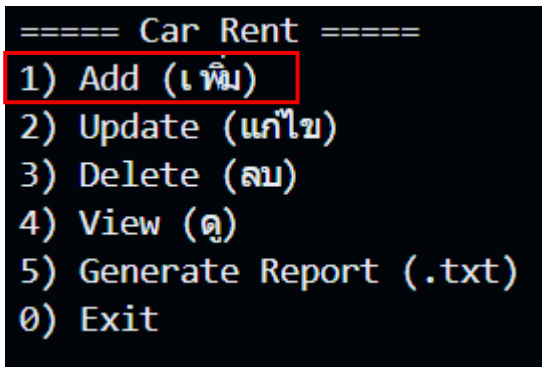
การใช้งานระบบเช่ารถ

โปรแกรมการเช่ารถยนต์เป็นบริการที่ช่วยตอบสนองความต้องการของผู้ใช้ในหลายด้าน ทั้งเพื่อการเดินทางส่วนตัว ธุรกิจ และการท่องเที่ยว โดยโปรแกรมเช่ารถช่วยลดภาระค่าใช้จ่ายจากการครอบครองรถยนต์ เช่น ค่าเสื่อมราคา ค่าซ่อมบำรุง และค่าภาษี อีกทั้งยังเพิ่มความยืดหยุ่นให้ผู้ใช้สามารถเลือกรถตามวัตถุประสงค์ เช่น รถเล็กสำหรับใช้งานในเมือง รถ SUV สำหรับเดินทางไกล หรือแม้แต่รถหรูสำหรับโอกาสพิเศษ ทำให้ผู้ใช้สามารถเข้าถึงการใช้รถได้โดยไม่จำเป็นต้องลงทุนซื้อรถยนต์

สำหรับผู้ใช้งานโปรแกรม

3.1 การใช้งานโปรแกรมระบบการเช่ารถ

3.1.1 กรอกหมายเลข 1 ภายในกรอบสีแดงเพื่อเรียกฟังก์ชัน Add เพิ่มข้อมูล ที่ประกอบไปด้วย Customer , Car , Rental



รูปภาพที่ 3-1 การเลือกใช้งานฟังก์ชั่น Add

3.1.2 เมื่อเมนูฟังก์ชั่น Add ขึ้นมาแล้วจากนั้นก็สามารถระบุเมนูที่ต้องการเลือกได้

```
---- Select Entity ----
1) Customer
2) Car
3) Rental
```

รูปภาพที่ 3-2 เมนูของ Add

3.1.3 กรอกหมายเลข 2 ภายในกรอบสีแดงเพื่อเรียกฟังก์ชัน Update เพื่อแก้ไขข้อมูลที่มีอยู่ ที่ประกอบไปด้วย Customer , Car , Rental

```
===== Car Rent =====
1) Add (เพิ่ม)
2) Update (แก้ไข)
3) Delete (ลบ)
4) View (ดู)
5) Generate Report (.txt)
0) Exit
```

รูปภาพที่ 3-3 การเลือกใช้งานฟังก์ชั่น Update

3.1.4 เมื่อเมนูฟังก์ชั่น Update ขึ้นมาแล้วจากนั้นก็สามารถระบุเมนูที่ต้องการเลือกได้

```
---- Select Entity ----
1) Customer
2) Car
3) Rental
```

รูปภาพที่ 3-4 เมนูของ Update

3.1.5 กรอกหมายเลข 3 ภายในกรอบสีแดงเพื่อเรียกฟังก์ชัน Delete เพื่อลบข้อมูลที่มีอยู่ ที่ประกอบไปด้วย Customer , Car , Rental

```
===== Car Rent =====
1) Add (เพิ่ม)
2) Update (แก้ไข)
3) Delete (ลบ)
4) View (ดู)
5) Generate Report (.txt)
0) Exit
```

รูปภาพที่ 3-5 การเลือกใช้งานฟังก์ชั่น Delete

3.1.6 เมื่อเมนูฟังก์ชั่น Delete ขึ้นมาแล้วจากนั้นก็สามารถระบุเมนูที่ต้องการเลือกได้

```
---- Select Entity ----
1) Customer
2) Car
3) Rental
```

รูปภาพที่ 3-6 เมนูของ Delete

3.1.7 กรอกหมายเลข 4 ภายในกรอบสีแดงเพื่อเรียกฟังก์ชัน View เพื่อดูข้อมูลที่มีอยู่ ที่ประกอบไปด้วย View one , View all , Filter , Summary stats , Back

```
===== Car Rent =====
1) Add (เพิ่ม)
2) Update (แก้ไข)
3) Delete (ลบ)
4) View (ดู)
5) Generate Report (.txt)
0) Exit
```

รูปภาพที่ 3-7 การเลือกใช้งานฟังก์ชั่น View

3.1.8 เมื่อเมนูฟังก์ชั่น View ขึ้นมาแล้วจากนั้นก็สามารถระบุเมนูที่ต้องการเลือกได้

```
---- View Submenu ----
1) View one
2) View all
3) Filter
4) Summary stats
0) Back
```

รูปภาพที่ 3-8 เมนูของ View

3.1.9 กรอกหมายเลข 5 ภายในกรอบสีแดงเพื่อเรียกฟังก์ชัน Generate report เพื่อเพิ่มไฟล์ report.txt ที่สามารถดูข้อมูลของการเช่าได้

```
===== Car Rent =====
1) Add (เพิ่ม)
2) Update (แก้ไข)
3) Delete (ลบ)
4) View (ดู)
5) Generate Report (.txt)
0) Exit
```

รูปภาพที่ 3-9 การเลือกใช้งานฟังก์ชั่น Generate Report

3.1.10 กรอกหมายเลข 0 ภายในกรอบสีแดงเพื่อเรียกฟังก์ชัน Exit เพื่อออกจากโปรแกรม

```
===== Car Rent =====
1) Add (เพิ่ม)
2) Update (แก้ไข)
3) Delete (ลบ)
4) View (ดู)
5) Generate Report (.txt)
0) Exit
```

รูปภาพที่ 3-10 การเลือกใช้งานฟังก์ชั่น Exit

3.2 การใช้งานโปรแกรมเพิ่มข้อมูล

3.2.1 กรอกหมายเลข 1 เพื่อเลือกฟังก์ชันเพิ่มข้อมูล

```
===== Car Rent =====
1) Add (เพิ่ม)
2) Update (แก้ไข)
3) Delete (ลบ)
4) View (ดู)
5) Generate Report (.txt)
0) Exit

Choose: 1
```

รูปภาพที่ 3-11 การเลือกใช้งานฟังก์ชั่น Add

3.2.2 เมื่อกรอกหมายเลข 1 แล้วจะเมนูมาให้เลือกว่าจะเพิ่มข้อมูลอะไร Customer , Car , Rental กด 1 เพื่อเพิ่มข้อมูลของลูกค้า และจะมีช่องเพื่อกรอกข้อมูล Customer ID , ID Card , Name , Tel.

```
---- Select Entity ----
1) Customer
2) Car
3) Rental

Choose: 1
```

รูปภาพที่ 3-12 การเลือกใช้งานเมนู Customer ของฟังก์ชัน Add

```
Customer ID: 1005
ID Card : 76894525
Name : Pong Munkong
Tel : 0961924162
  ✓ บันทึกลูกค้าแล้ว
```

รูปภาพที่ 3-13 การเพิ่มข้อมูลลูกค้า

3.2.3 กรอกหมายเลข 1 เพื่อเลือกฟังก์ชันเพิ่มข้อมูล

```
===== Car Rent =====
1) Add (เพิ่ม)
2) Update (แก้ไข)
3) Delete (ลบ)
4) View (ดู)
5) Generate Report (.txt)
0) Exit

Choose: 1
```

รูปภาพที่ 3-14 การเลือกใช้งานฟังก์ชั่น Add

3.2.4 เมื่อกรอกหมายเลข 1 แล้วจะเมนูมาให้เลือกว่าจะเพิ่มข้อมูลอะไร Customer , Car , Rental กด 2 เพื่อเพิ่มข้อมูลของรถ และจะมีช่องเพื่อกรอกข้อมูล Car ID , License plate , Brand , Model , Year , Rate (THB/Day)

```
---- Select Entity ----
1) Customer
2) Car
3) Rental

Choose: 2
```

รูปภาพที่ 3-15 การเลือกใช้งานเมนู Car ของฟังก์ชัน Add

```
Car ID: 2005
License plate : ขค 7678
Brand : Honda
Model : City
Year: 2020
Rate (THB/Day): 1100
  ✓ บันทึกรถแล้ว
```

รูปภาพที่ 3-16 การเพิ่มข้อมูลรถ

3.2.5 กรอกหมายเลข 1 เพื่อเลือกฟังก์ชันเพิ่มข้อมูล

```
===== Car Rent =====
1) Add (เพิ่ม)
2) Update (แก้ไข)
3) Delete (ลบ)
4) View (ดู)
5) Generate Report (.txt)
0) Exit

Choose: 1
```

รูปภาพที่ 3-17 การเลือกใช้งานฟังก์ชั่น Add

3.2.6 เมื่อกรอกหมายเลข 1 แล้วจะเมนูมาให้เลือกว่าจะเพิ่มข้อมูลอะไร Customer , Car , Rental กด 3 เพื่อเพิ่มข้อมูลการเช่า และจะมีช่องเพื่อกรอกข้อมูล Rental ID , Car ID , Customer ID , Start date (YYYY-MM-DD) , End date (YYYY-MM-DD)

```
---- Select Entity ----
1) Customer
2) Car
3) Rental

Choose: 3
```

รูปภาพที่ 3-18 การเลือกใช้งานเมนู Rental ของฟังก์ชัน Add

```
Rental ID: 3005
Car ID: 2005
Customer ID: 1005
Start date (YYYY-MM-DD): 2025-09-29
End date (YYYY-MM-DD): 2025-10-10
  ✓ บันทึกเช่าแล้ว (Amount 13200.00)
```

รูปภาพที่ 3-19 การเพิ่มข้อมูลการเช่า

3.3 การใช้งานโปรแกรมแก้ไขข้อมูล

3.3.1 กรอกหมายเลข 2 เพื่อเลือกฟังก์ชันแก้ไขข้อมูล

```
===== Car Rent =====
1) Add (เพิ่ม)
2) Update (แก้ไข)
3) Delete (ลบ)
4) View (ดู)
5) Generate Report (.txt)
0) Exit

Choose: 2
```

รูปภาพที่ 3-20 การเลือกใช้งานฟังก์ชั่น Update

3.3.2 เมื่อกรอกหมายเลข 2 แล้วจะเมนูมาให้เลือกว่าจะแก้ไขข้อมูลอะไร Customer , Car , Rental กด 1 เพื่อแก้ไขข้อมูลลูกค้า และจะมีช่องเพื่อกรอกข้อมูล Customer ID , Name , Tel หากไม่ต้องการแก้ไขให้กด Enter ได้เลย

```
---- Select Entity ----
1) Customer
2) Car
3) Rental

Choose: 1
```

รูปภาพที่ 3-21 การเลือกใช้งานเมนู Customer ของฟังก์ชัน Update

```
Customer ID ที่จะแก้: 1005
Name [Pong Munkong]:
Tel  [0961924162]:
  ✓ อัปเดตแล้ว
```

รูปภาพที่ 3-22 การแก้ไขข้อมูลลูกค้า

3.3.3 กรอกหมายเลข 2 เพื่อเลือกฟังก์ชันแก้ไขข้อมูล

```
===== Car Rent =====
1) Add (เพิ่ม)
2) Update (แก้ไข)
3) Delete (ลบ)
4) View (ดู)
5) Generate Report (.txt)
0) Exit

Choose: 2
```

รูปภาพที่ 3-23 การเลือกใช้งานฟังก์ชั่น Update

3.3.4 เมื่อกรอกหมายเลข 2 แล้วจะเมนูมาให้เลือกว่าจะแก้ไขข้อมูลอะไร Customer , Car , Rental กด 2 เพื่อแก้ไขข้อมูลรถ และจะมีช่องเพื่อกรอกข้อมูล Car ID , Brand , Model , Year , Rate , Status หากไม่ต้องการแก้ไขให้กด Enter ได้เลย

```
---- Select Entity ----
1) Customer
2) Car
3) Rental

Choose: 2
```

รูปภาพที่ 3-24 การเลือกใช้งานเมนู Car ของฟังก์ชัน Update

```
Car ID ที่จะแก้: 2005
Brand [Honda]:
Model [City]:
Year  [2020]:
Rate  [1100]:
Status(1=Active,0=Inactive) [1]:
  ✓ อัปเดตแล้ว
```

รูปภาพที่ 3-25 การแก้ไขข้อมูลรถ

3.3.5 กรอกหมายเลข 2 เพื่อเลือกฟังก์ชันแก้ไขข้อมูล

```
===== Car Rent =====
1) Add (เพิ่ม)
2) Update (แก้ไข)
3) Delete (ลบ)
4) View (ดู)
5) Generate Report (.txt)
0) Exit

Choose: 2
```

รูปภาพที่ 3-26 การเลือกใช้งานฟังก์ชั่น Update

3.3.6 เมื่อกรอกหมายเลข 2 แล้วจะเมนูมาให้เลือกว่าจะแก้ไขข้อมูลอะไร Customer , Car , Rental กด 3 เพื่อแก้ไขข้อมูลการเช่า และจะมีช่องเพื่อกรอกข้อมูล Rental ID , Status , Start , End หากไม่ต้องการแก้ไขให้กด Enter ได้เลย

```
---- Select Entity ----
1) Customer
2) Car
3) Rental

Choose: 3
```

รูปภาพที่ 3-27 การเลือกใช้งานเมนู Rental ของฟังก์ชัน Update

```
Rental ID ที่จะแก้: 3005
Status(1=Open,0=Closed,-1=Deleted) [1]:
Start [2025-09-29 YYYY-MM-DD or blank]:
End   [2025-10-10 YYYY-MM-DD or blank]:
  ✓ อัปเดตแล้ว
```

รูปภาพที่ 3-28 การแก้ไขข้อมูลการเช่า

3.4 การใช้งานโปรแกรมลบข้อมูล

3.4.1 กรอกหมายเลข 3 เพื่อเลือกฟังก์ชันลบข้อมูล

```
===== Car Rent =====
1) Add (เพิ่ม)
2) Update (แก้ไข)
3) Delete (ลบ)
4) View (ดู)
5) Generate Report (.txt)
0) Exit

Choose: 3
```

รูปภาพที่ 3-29 การเลือกใช้งานฟังก์ชั่น Delete

3.4.2 เมื่อกรอกหมายเลข 3 แล้วจะเมนูมาให้เลือกว่าจะลบข้อมูลอะไร Customer , Car , Rental กด 1 เพื่อลบข้อมูลลูกค้า และจะมีช่องเพื่อกรอกข้อมูล Customer ID การที่จะลบได้ลูกค้าจะต้องไม่มีสัญญาเช่าติดค้างอยู่

```
---- Select Entity ----
1) Customer
2) Car
3) Rental

Choose: 1
```

รูปภาพที่ 3-30 การเลือกใช้งานเมนู Customer ของฟังก์ชัน Delete

```
Customer ID ที่จะลบ (จะตั้ง Inactive): 1005
  ✓ ทำเครื่องหมายแล้ว (ลูกค้าไม่มี status; ข้ามการลบ)
```

รูปภาพที่ 3-31 การลบข้อมูลลูกค้า

3.4.3 กรอกหมายเลข 3 เพื่อเลือกฟังก์ชันลบข้อมูล

```
===== Car Rent =====
1) Add (เพิ่ม)
2) Update (แก้ไข)
3) Delete (ลบ)
4) View (ดู)
5) Generate Report (.txt)
0) Exit

Choose: 3
```

รูปภาพที่ 3-32 การเลือกใช้งานฟังก์ชั่น Delete

3.4.4 เมื่อกรอกหมายเลข 3 แล้วจะเมนูมาให้เลือกว่าจะลบข้อมูลอะไร Customer , Car , Rental กด 2 เพื่อลบข้อมูลรถ และจะมีช่องเพื่อกรอกข้อมูล Car ID การที่จะลบได้รถจะต้องไม่มีสัญญาเช่าติดค้างอยู่

```
---- Select Entity ----
1) Customer
2) Car
3) Rental

Choose: 2
```

รูปภาพที่ 3-33 การเลือกใช้งานเมนู Car ของฟังก์ชัน Delete

```
Car ID ที่จะลบ (จะตั้ง Inactive): 2005
  ✓ ตั้งรถเป็น Inactive แล้ว
```

รูปภาพที่ 3-34 การลบข้อมูลรถ

3.4.5 กรอกหมายเลข 3 เพื่อเลือกฟังก์ชันลบข้อมูล

```
===== Car Rent =====
1) Add (เพิ่ม)
2) Update (แก้ไข)
3) Delete (ลบ)
4) View (ดู)
5) Generate Report (.txt)
0) Exit

Choose: 3
```

รูปภาพที่ 3-35 การเลือกใช้งานฟังก์ชั่น Delete

3.4.6 เมื่อกรอกหมายเลข 3 แล้วจะเมนูมาให้เลือกว่าจะลบข้อมูลอะไร Customer , Car , Rental กด 3 เพื่อลบข้อมูลการเช่า และจะมีช่องเพื่อกรอกข้อมูล Rental ID

```
---- Select Entity ----
1) Customer
2) Car
3) Rental

Choose: 3
```

รูปภาพที่ 3-36 การเลือกใช้งานเมนู Rental ของฟังก์ชัน Delete

```
Rental ID ที่จะลบ: 3005
  ✓ ทำเครื่องหมายลบแล้ว (-1)
```

รูปภาพที่ 3-37 การลบข้อมูลการเช่า

3.5 การใช้งานโปรแกรมดูข้อมูล

3.5.1 กรอกหมายเลข 4 เพื่อเลือกฟังก์ชันดูข้อมูล

```
===== Car Rent =====
1) Add (เพิ่ม)
2) Update (แก้ไข)
3) Delete (ลบ)
4) View (ดู)
5) Generate Report (.txt)
0) Exit

Choose: 4
```

รูปภาพที่ 3-38 การเลือกใช้งานฟังก์ชั่น View

3.5.2 เมื่อกรอกหมายเลข 4 แล้วจะเมนูมาให้เลือกว่าจะดูข้อมูลแบบไหน View one , View all , Summary stats , Back กด 1 เพื่อดูข้อมูลแบบเดียว จะมีเมนูขึ้นมาว่าจะดูข้อมูลของอะไร Customer , Car , Rental กด 1 เพื่อดูข้อมูลของลูกค้า และจะมีให้กรอกข้อมูล Customer ID

```
---- View Submenu ----
1) View one
2) View all
3) Summary stats
0) Back

Choose:
```

รูปภาพที่ 3-38 การเลือกใช้งานเมนู View one

```
---- Select Entity ----
1) Customer
2) Car
3) Rental

Choose: 1
```

รูปภาพที่ 3-39 การเลือกใช้งานเมนู Customer

```
Customer ID: 1005
{'customer_id': 1005, 'id_card': '76894525', 'name': 'Pong Munkong', 'tel': '0961924162'}
```

รูปภาพที่ 3-40 การดูข้อมูลลูกค้า

3.5.3 กรอกหมายเลข 4 เพื่อเลือกฟังก์ชันดูข้อมูล

```
===== Car Rent =====
1) Add (เพิ่ม)
2) Update (แก้ไข)
3) Delete (ลบ)
4) View (ดู)
5) Generate Report (.txt)
0) Exit

Choose: 4
```

รูปภาพที่ 3-41 การเลือกใช้งานฟังก์ชั่น View

3.5.4 เมื่อกรอกหมายเลข 4 แล้วจะเมนูมาให้เลือกว่าจะดูข้อมูลแบบไหน View one , View all , Summary stats , Back กด 1 เพื่อดูข้อมูลแบบเดียว จะมีเมนูขึ้นมาว่าจะดูข้อมูลของอะไร Customer , Car , Rental กด 2 เพื่อดูข้อมูลของรถ และจะมีให้กรอกข้อมูล Car ID

```
---- View Submenu ----
1) View one
2) View all
3) Summary stats
0) Back

Choose:
```

รูปภาพที่ 3-42 การเลือกใช้งานเมนู View one

```
---- Select Entity ----
1) Customer
2) Car
3) Rental

Choose: 2
```

รูปภาพที่ 3-43 การเลือกใช้งานเมนู Customer

```
Car ID: 2005
{'car_id': 2005, 'plate': 'ขค 7678', 'brand': 'Honda', 'model': 'City', 'year': 2020, 'rate': 1100, 'status': 0, 'is_rented': 1}
```

รูปภาพที่ 3-44 การดูข้อมูลรถ

3.5.5 กรอกหมายเลข 4 เพื่อเลือกฟังก์ชันดูข้อมูล

```
===== Car Rent =====
1) Add (เพิ่ม)
2) Update (แก้ไข)
3) Delete (ลบ)
4) View (ดู)
5) Generate Report (.txt)
0) Exit

Choose: 4
```

รูปภาพที่ 3-45 การเลือกใช้งานฟังก์ชั่น View

3.5.6 เมื่อกรอกหมายเลข 4 แล้วจะเมนูมาให้เลือกว่าจะดูข้อมูลแบบไหน View one , View all , Summary stats , Back กด 1 เพื่อดูข้อมูลแบบเดียว จะมีเมนูขึ้นมาว่าจะดูข้อมูลของอะไร Customer , Car , Rental กด 3 เพื่อดูข้อมูลการเช่า และจะมีให้กรอกข้อมูล Rental ID

```
---- View Submenu ----
1) View one
2) View all
3) Summary stats
0) Back

Choose:
```

รูปภาพที่ 3-46 การเลือกใช้งานเมนู View one

```
---- Select Entity ----
1) Customer
2) Car
3) Rental

Choose: 3
```

รูปภาพที่ 3-47 การเลือกใช้งานเมนู Rental

```
Rental ID: 3005
{'rental_id': 3005, 'car_id': 2005, 'customer_id': 1005, 'start_ymd': 20250929, 'end_ymd': 20251010, 'total_days': 12
, 'status': -1, 'total_amount': 13200.0, 'start': datetime.date(2025, 9, 29), 'end': datetime.date(2025, 10, 10)}
```

รูปภาพที่ 3-48 การดูข้อมูลการเช่า

3.5.7 กรอกหมายเลข 4 เพื่อเลือกฟังก์ชันดูข้อมูล

```
===== Car Rent =====
1) Add (เพิ่ม)
2) Update (แก้ไข)
3) Delete (ลบ)
4) View (ดู)
5) Generate Report (.txt)
0) Exit

Choose: 4
```

รูปภาพที่ 3-49 การเลือกใช้งานฟังก์ชั่น View

3.5.8 เมื่อกรอกหมายเลข 4 แล้วจะเมนูมาให้เลือกว่าจะดูข้อมูลแบบไหน View one , View all , Summary stats , Back กด 2 เพื่อดูข้อมูลทั้งหมด จะมีเมนูขึ้นมาว่าจะดูข้อมูลของอะไร Customer , Car , Rental กด 1 เพื่อดูข้อมูลลูกค้า

```
---- View Submenu ----
1) View one
2) View all
3) Summary stats
0) Back

Choose:
```

รูปภาพที่ 3-50 การเลือกใช้งานเมนู View all

```
---- Select Entity ----
1) Customer
2) Car
3) Rental

Choose: 1
```

รูปภาพที่ 3-51 การเลือกใช้งานเมนู Customer

```
ID   | ID Card  | Name        | Tel
-----------------------------------------
1001 | 98372634 | Alice       | 0987315278
1002 | 46789281 | Mix         | 0942847589
1003 | 48928415 | Auu         | 0831285679
1004 | 48731823 | Tea         | 0963278917
1005 | 76894525 | Pong Munkong | 0961924162
```

รูปภาพที่ 3-52 การดูข้อมูลลูกค้า

3.5.9 กรอกหมายเลข 4 เพื่อเลือกฟังก์ชันดูข้อมูล

```
===== Car Rent =====
1) Add (เพิ่ม)
2) Update (แก้ไข)
3) Delete (ลบ)
4) View (ดู)
5) Generate Report (.txt)
0) Exit

Choose: 4
```

รูปภาพที่ 3-53 การเลือกใช้งานฟังก์ชั่น View

3.5.10 เมื่อกรอกหมายเลข 4 แล้วจะเมนูมาให้เลือกว่าจะดูข้อมูลแบบไหน View one , View all , Summary stats , Back กด 2 เพื่อดูข้อมูลทั้งหมด จะมีเมนูขึ้นมาว่าจะดูข้อมูลของอะไร Customer , Car , Rental กด 2 เพื่อดูข้อมูลรถ

```
---- View Submenu ----
1) View one
2) View all
3) Summary stats
0) Back

Choose:
```

รูปภาพที่ 3-54 การเลือกใช้งานเมนู View all

```
---- Select Entity ----
1) Customer
2) Car
3) Rental

Choose: 2
```

รูปภาพที่ 3-55 การเลือกใช้งานเมนู Car

```
ID   | Plate    | Brand  | Model | Year | Rate | Status   | Rented
-----------------------------------------------------------------
2001 | ขค 5679  | Toyota | Vios  | 2021 | 1100 | Active   | Yes
2002 | กข 9999  | Honda  | Civic | 2025 | 1500 | Active   | Yes
2003 | กง 2134  | Nissan | Amela | 2025 | 1300 | Active   | Yes
2004 | บก 1234  | Mazda  | 3     | 2025 | 1100 | Active   | Yes
2005 | ขค 7678  | Honda  | City  | 2020 | 1100 | Inactive | Yes
```

รูปภาพที่ 3-56 การดูข้อมูลรถ

3.5.11 กรอกหมายเลข 4 เพื่อเลือกฟังก์ชันดูข้อมูล

```
===== Car Rent =====
1) Add (เพิ่ม)
2) Update (แก้ไข)
3) Delete (ลบ)
4) View (ดู)
5) Generate Report (.txt)
0) Exit

Choose: 4
```

รูปภาพที่ 3-57 การเลือกใช้งานฟังก์ชั่น View

3.5.12 เมื่อกรอกหมายเลข 4 แล้วจะเมนูมาให้เลือกว่าจะดูข้อมูลแบบไหน View one , View all , Summary stats , Back กด 2 เพื่อดูข้อมูลทั้งหมด จะมีเมนูขึ้นมาว่าจะดูข้อมูลของอะไร Customer , Car , Rental กด 3 เพื่อดูข้อมูลการเช่า

```
---- View Submenu ----
1) View one
2) View all
3) Summary stats
0) Back

Choose:
```

รูปภาพที่ 3-58 การเลือกใช้งานเมนู View all

```
---- Select Entity ----
1) Customer
2) Car
3) Rental

Choose: 3
```

รูปภาพที่ 3-59 การเลือกใช้งานเมนู Rental

```
RID  | Car  | Cust | Start      | End        | Days | Status   | Amount
-----------------------------------------------------------------------
3001 | 2001 | 1001 | 2025-10-01 | 2025-10-10 | 10   | Open     | 11,000.00
3002 | 2002 | 1002 | 2025-09-28 | 2025-10-02 | 5    | Open     | 7,500.00
3003 | 2003 | 1003 | 2025-10-02 | 2025-10-05 | 4    | Open     | 5,200.00
3004 | 2004 | 1004 | 2025-10-02 | 2025-10-06 | 5    | Open     | 5,500.00
3005 | 2005 | 1005 | 2025-09-29 | 2025-10-10 | 12   | Deleted  | 13,200.00
```

รูปภาพที่ 3-60 การดูข้อมูลการเช่า

3.5.13 เมื่อกรอกหมายเลข 4 แล้วจะเมนูมาให้เลือกว่าจะดูข้อมูลแบบไหน View one , View all , Summary stats , Back กด 3 เพื่อดูข้อมูลสถิติสรุป โดยจะบอกจำนวนลูกค้า , จำนวนรถ , จำนวนข้อมูลการเช่า

```
---- View Submenu ----
1) View one
2) View all
3) Summary stats
0) Back

Choose:
```

รูปภาพที่ 3-61 การเลือกใช้งานเมนู Summary stats

```
--- Summary Stats ---
Customers : 4
Cars      : 4 (Active 4, Inactive 0)
Rentals   : 4 (Open 4, Closed 0, Deleted 0)
```

รูปภาพที่ 3-62 การดูข้อมูลสถิติสรุป

3.5.14 เมื่อกรอกหมายเลข 4 แล้วจะเมนูมาให้เลือกว่าจะดูข้อมูลแบบไหน View one , View all , Summary stats , Back กด 0 เพื่อย้อนกลับไปยังเมนูก่อนหน้า

```
---- View Submenu ----
1) View one
2) View all
3) Summary stats
0) Back

Choose: 0
```

รูปภาพที่ 3-61 การเลือกใช้งานเมนู Back

3.6 การใช้งานโปรแกรมสร้างรายงาน

3.6.1 กรอกหมายเลข 5 เพื่อเลือกฟังก์ชันสร้างรายงาน

```
===== Car Rent =====
1) Add (เพิ่ม)
2) Update (แก้ไข)
3) Delete (ลบ)
4) View (ดู)
5) Generate Report (.txt)
0) Exit

Choose: 5
```

รูปภาพที่ 3-62 การเลือกใช้งานฟังก์ชันสร้างรายงาน

3.6.2 เมื่อกรอกเสร็จระบบจะทำงานการรายงาน ชื่อไฟล์ว่า report.txt เพื่อดูข้อมูลต่างๆ

```
Car Rent System - Summary Report
Generated At: 2025-10-02 19:58:57
App Version: 1.0
Endianness: Little-Endian
Encoding: UTF-8 (fixed-length)

Rental_ID | Customer name | Tel.       | License_plate | Brand  | Model | Rate  | Date Rent  | Return date | Rental Day | Total Price
-------------------------------------------------------------------------------------------------------------------------------------------
3001      | Alice         | 0987315278 | ขค 5679       | Toyota | Vios  | 1,100 | 10/01/2025 | 10/10/2025  | 10         | 11,000.00
3002      | Mix           | 0942847589 | กข 9999       | Honda  | Civic | 1,500 | 09/28/2025 | 10/02/2025  | 5          | 7,500.00
3003      | Auu           | 0831285679 | กง 2134       | Nissan | Amela | 1,300 | 10/02/2025 | 10/05/2025  | 4          | 5,200.00
3004      | Tea           | 0963278917 | นก 1234       | Mazda  | 3     | 1,100 | 10/02/2025 | 10/06/2025  | 5          | 5,500.00

--- Summary ---

Customers : 4
Cars      : 4 (Active 4, Inactive 0)
Rentals   : 4 (Open 4, Closed 0, Deleted 0)

Rate Statistics (Active cars only)
- Min Rate : 1,100
- Max Rate : 1,500
- Avg Rate : 1,250

Cars by Brand
- Honda : 1
- Mazda : 1
- Nissan : 1
- Toyota : 1
```

รูปภาพที่ 3-63 ไฟล์ report.txt

3.7 การปิดใช้งานโปรแกรม

3.7.1 กรอกหมายเลข 0 เพื่อเลือกฟังก์ชันปิดโปรแกรม

```
===== Car Rent =====
1) Add (เพิ่ม)
2) Update (แก้ไข)
3) Delete (ลบ)
4) View (ดู)
5) Generate Report (.txt)
0) Exit

Choose: 0
```

รูปภาพที่ 3-64 การเลือกใช้งานฟังก์ชันปิดโปรแกรม

3.7.2 เมื่อกด Enter โปรแกรมจะทำงานปิด และทำการสร้างไฟล์ report.txt เพื่ออัพเดตข้อมูลให้เป็นปัจจุบัน

```
✓ สร้างรายงานแล้ว -> report.txt
Bye!
```

รูปภาพที่ 3-65 Exit Program

บทที่ 4

อธิบายการทำงานของ code

4.1 ฟังก์ชั่นไบนารีพื้นฐานในระบบเซ่ารถ

4.1.1 Module Struct เป็นโมดูลใน Python ที่ใช้สำหรับการจัดการข้อมูลแบบไบนารี เช่น การแปลงข้อมูลจากรูปแบบ Python (เช่น integer, float) ไปเป็นไบต์ หรือการแปลงข้อมูลจากไบต์กลับมาเป็นรูปแบบ Python อีกครั้ง โมดูลนี้สำคัญเมื่อเราต้องการทำงานกับไฟล์หรือข้อมูลที่อยู่ในรูปแบบไบนารี เช่นไฟล์

```
import struct
```

รูปภาพที่ 4-1 Code Module Struct

4.1.2 import datetime และ date คือคำสั่งในภาษา Python ที่ใช้สำหรับนำเข้า (import) คลาสจากโมดูล datetime มาใช้งาน datetime ⟶ ใช้สำหรับจัดการข้อมูลที่เกี่ยวข้องกับ วันและเวลา เช่น วันที่ปัจจุบัน เวลาในขณะนั้น หรือการคำนวณเพิ่ม/ลบวันและเวลา เพื่อใช้ในการบันทึกเวลา การจัดตารางงาน หรือการตรวจสอบเวลาที่ผ่านมาแล้ว date ⟶ ใช้สำหรับจัดการข้อมูลที่เกี่ยวข้องกับ วันที่ (ปี/เดือน/วัน) โดยไม่รวมเวลา เช่น ใช้แสดงวันเกิด กำหนดวันนัดหมาย หรือตรวจสอบว่าวันที่ใดตรงกับวันอะไร (เช่น วันจันทร์-อาทิตย์) ดังนั้นการ import datetime, date จะช่วยให้เราสามารถเขียนโปรแกรมที่เกี่ยวข้องกับวันและเวลาได้สะดวกขึ้น ทั้งการแสดงผล การเปรียบเทียบ และการคำนวณเกี่ยวกับเวลา

```
import datetime, date
```

รูปภาพที่ 4-2 Code Module datetime , date

4.1.3 import os, sys, io เป็นการนำเข้า (import) โมดูลที่เกี่ยวข้องกับการทำงานของระบบปฏิบัติการ การจัดการโปรแกรม และการทำงานกับไฟล์/ข้อมูล โดยมีรายละเอียดดังนี้

os ⟶ ใช้สำหรับทำงานที่เกี่ยวข้องกับ ระบบปฏิบัติการ (Operating System) เช่น การสร้าง/ลบโฟลเดอร์, ตรวจสอบ path ของไฟล์, จัดการ environment variables หรือเรียกคำสั่งของระบบจาก Python ได้โดยตรง

sys ⟶ ใช้สำหรับจัดการสิ่งที่เกี่ยวข้องกับ Python Interpreter เช่น การอ่านค่าพารามิเตอร์ที่ส่งมาตอนรันโปรแกรม (sys.argv), การหยุดการทำงานของโปรแกรม (sys.exit()), และการปรับแต่ง path ของโมดูลที่จะนำเข้า

io ⟶ ใช้สำหรับทำงานกับ input/output stream เช่น การอ่านไฟล์, เขียนไฟล์, การทำงานกับข้อมูลในรูปแบบ text หรือ binary ให้มีความยืดหยุ่นและสะดวกมากขึ้น

ดังนั้นการ import os, sys, io จะช่วยให้โปรแกรมสามารถติดต่อกับระบบปฏิบัติการ จัดการพารามิเตอร์ที่ใช้รันโปรแกรม และทำงานกับข้อมูลหรือไฟล์ได้อย่างสะดวกและมีประสิทธิภาพ

```
import os, sys, io
```

รูปภาพที่ 4-3 Code import os , sys , io

4.1.4 โครงสร้างข้อมูลของ Customer , Car , Rent คือ format string กำหนด layout ของ record แต่ละเล่มดังรูปที่ 4-4 , 4-5 , 4-6

```
CUSTOMER_FMT = "<i15s60s12s"
```

รูปภาพที่ 4-4 Code Customer

```
CAR_FMT = "<i12s12s16siiii"
```

รูปภาพที่ 4-5 Code Car

```
RENT_FMT = "<iiiIIiif"
```

รูปภาพที่ 4-6 Code Rent

4.2 ฟังก์ชัน add_customer()

4.2.1 ฟังก์ชัน add_customer() ฟังก์ชันนี้ใช้สำหรับ เพิ่มข้อมูลลูกค้าใหม่ ลงในระบบ โดยมีขั้นตอนการทำงานดังนี้ ใช้ฟังก์ชัน read_all() เพื่ออ่านข้อมูลลูกค้าทั้งหมดจากไฟล์เก็บข้อมูล กำหนดให้ใช้ customer_id เป็น กุญแจ (key) สำหรับตรวจสอบว่ามีรหัสลูกค้าซ้ำหรือไม่ รับค่ารหัสลูกค้า (Customer ID) จากผู้ใช้ โดยให้ค่าเริ่มต้นเป็น 1 หากรหัสลูกค้าที่กรอกมีอยู่แล้วในระบบ จะแสดงข้อความเตือนว่า “ซ้ำ” และยกเลิกการเพิ่มข้อมูล รับข้อมูลลูกค้าใหม่ ได้แก่ รหัสลูกค้า, เลขบัตรประชาชน, ชื่อ-นามสกุล, และเบอร์โทรศัพท์ แล้วเก็บในรูปแบบ Dictionary ใช้ append_record() เพื่อบันทึกข้อมูลลูกค้าใหม่ (d) เข้าไปในไฟล์ เมื่อบันทึกเสร็จ จะแสดงข้อความยืนยันว่าได้บันทึกลูกค้าแล้ว

```
def add_customer():
    rows = read_all(CUST_PATH, CUSTOMER_FMT, CUSTOMER_FIELDS, CUSTOMER_SIZE)
    idx = index_by_key(rows, "customer_id")
    cid = ask_int("Customer ID: ", 1)
    if cid in idx:
        print("  ! ซ้ำ")
        return
    d = {
        "customer_id": cid,
        "id_card": ask_str("ID Card : ", 15),
        "name": ask_str("Name : ", 60),
        "tel": ask_str("Tel : ", 12),
    }
    append_record(CUST_PATH, CUSTOMER_FMT, pack_customer(d))
    print("  ✓ บันทึกลูกค้าแล้ว")
```

รูปภาพที่ 4-7 add_customer

4.3 ฟังก์ชัน add_car()

4.3.1 ฟังก์ชัน add_car() ฟังก์ชันนี้ใช้สำหรับ เพิ่มข้อมูลรถใหม่ เข้าสู่ระบบ โดยทำงานตามลำดับดังนี้ อ่านข้อมูลรถทั้งหมดจากไฟล์เก็บข้อมูลด้วย read_all() สร้างดัชนีโดยใช้ car_id เป็นกุญแจหลัก เพื่อเช็คว่ามีรหัสรถนี้อยู่แล้วหรือไม่ รับค่า Car ID ใหม่จากผู้ใช้ หาก Car ID ซ้ำกับข้อมูลเดิม ระบบจะแสดงข้อความว่า “ซ้ำ” และยกเลิกการเพิ่มข้อมูล หาก Car ID ไม่ซ้ำ จะให้ผู้ใช้กรอกข้อมูลรถใหม่ ได้แก่ ทะเบียนรถ (plate) , ยี่ห้อรถ (brand) , รุ่นรถ (model) , ปีที่ผลิต (year) โดยกำหนดให้กรอกได้เฉพาะปี 1900–2100 , อัตราค่าเช่า (rate) ต่อวัน (THB/Day) , สถานะ (status) เริ่มต้นเป็น CAR_ACTIVE , is_rented กำหนดเป็น 0 เพื่อบอกว่ายังไม่ได้ถูกเช่า บันทึกข้อมูลรถใหม่ที่ได้ลงไฟล์เก็บข้อมูลด้วย append_record() แสดงข้อความยืนยันว่าได้บันทึกรถใหม่เรียบร้อยแล้ว

```
def add_car():
    rows = read_all(CAR_PATH, CAR_FMT, CAR_FIELDS, CAR_SIZE)
    idx = index_by_key(rows, "car_id")
    cid = ask_int("Car ID: ", 1)
    if cid in idx:
        print("  ! ซ้ำ")
        return
    d = {
        "car_id": cid,
        "plate": ask_str("License plate : ", 12),
        "brand": ask_str("Brand : ", 12),
        "model": ask_str("Model : ", 16),
        "year": ask_int("Year: ", 1900, 2100),
        "rate": ask_int("Rate (THB/Day): ", 0),
        "status": CAR_ACTIVE,
        "is_rented": 0,
    }
    append_record(CAR_PATH, CAR_FMT, pack_car(d))
    print("  ✓ บันทึกรถแล้ว")
```

รูปภาพที่ 4-8 add_car

4.4 ฟังก์ชัน add_rental()

4.4.1 ฟังก์ชัน add_rental() ฟังก์ชันนี้ใช้สำหรับ บันทึกข้อมูลการเช่ารถใหม่ โดยทำงานตามลำดับขั้นตอนดังนี้ อ่านข้อมูล รถทั้งหมด, ลูกค้าทั้งหมด และการเช่าที่มีอยู่แล้ว สร้างดัชนี (index) สำหรับค้นหารถจาก car_id และลูกค้าจาก customer_id เก็บค่า rental_id ที่มีอยู่แล้วทั้งหมดเพื่อป้องกันไม่ให้ซ้ำ รับค่า Rental ID จากผู้ใช้ และตรวจสอบว่ามีรหัสนี้อยู่แล้วหรือไม่ ถ้าซ้ำจะยกเลิกการทำงาน รับค่า Car ID และ Customer ID ตรวจสอบว่ามีอยู่จริงในระบบหรือไม่ ตรวจสอบสถานะของรถว่าต้องเป็น CAR_ACTIVE เท่านั้นจึงจะสามารถนำมาให้เช่าได้ รับวันเริ่มต้นและวันสิ้นสุดการเช่า พร้อมตรวจสอบว่าต้องไม่เป็นวันที่ย้อนหลัง (วันสิ้นสุดต้องไม่ก่อนวันเริ่มต้น) คำนวณจำนวนวันที่เช่า (days) และจำนวนเงินรวม (total) จากอัตราค่าเช่าต่อวัน ตรวจสอบว่ารถคันนี้ถูกเช่าในช่วงเวลาเดียวกันไปแล้วหรือไม่ (ป้องกันการจองซ้ำทับช่วงเวลา) หากผ่านเงื่อนไขทั้งหมดจะสร้างข้อมูลการเช่าใหม่ (rental record) และบันทึกลงไฟล์ อัปเดตสถานะของรถ (is_rented = 1) เพื่อบอกว่ารถถูกเช่าแล้ว แสดงข้อความยืนยันว่าการบันทึกสำเร็จ พร้อมจำนวนเงินที่ต้องชำระ

```python
def add_rental():
    cars = read_all(CAR_PATH, CAR_FMT, CAR_FIELDS, CAR_SIZE)
    customers = read_all(CUST_PATH, CUSTOMER_FMT, CUSTOMER_FIELDS, CUSTOMER_SIZE)
    rents = read_all(RENT_PATH, RENT_FMT, RENT_FIELDS, RENT_SIZE)
    car_idx, cust_idx = index_by_key(cars, "car_id"), index_by_key(
        customers, "customer_id"
    )
    rent_ids = set(r["rental_id"] for r in rents)

    rid = ask_int("Rental ID: ", 1)
    if rid in rent_ids:
        print("  ! ซ้ำ")
        return
    car_id = ask_int("Car ID: ", 1)
    cust_id = ask_int("Customer ID: ", 1)
    if car_id not in car_idx:
        print("  ! car_id ไม่พบ")
        return
    if cust_id not in cust_idx:
        print("  ! customer_id ไม่พบ")
        return
    if car_idx[car_id][1]["status"] != CAR_ACTIVE:
        print("  ! รถไม่อยู่สถานะ Active")
        return

    start, end = ask_ymd("Start date"), ask_ymd("End date")
    sd, ed = ymd(start), ymd(end)
    if ed < sd:
        print("  ! end < start")
        return

    days = (ed - sd).days + 1
    rate = car_idx[car_id][1]["rate"]
    total = float(days * rate)

    # overlap check (non-deleted)
    existing = read_all(RENT_PATH, RENT_FMT, RENT_FIELDS, RENT_SIZE)
    for r in existing:
        if r["car_id"] != car_id or r["status"] == RENT_DELETED:
            continue
        if r["start_ymd"] <= end and r["end_ymd"] >= start:
            print("  ! ช่วงเช่าทับกับรายการเดิม (car_id เดียวกัน)")
            return

    d = {
        "rental_id": rid,
        "car_id": car_id,
        "customer_id": cust_id,
        "start_ymd": start,
        "end_ymd": end,
        "total_days": days,
        "status": RENT_OPEN,
        "total_amount": total,
    }
    append_record(RENT_PATH, RENT_FMT, pack_rent(d))

    # mark car as rented
    i_car, car_row = car_idx[car_id]
    car_row["is_rented"] = 1
    write_record_by_index(CAR_PATH, CAR_FMT, pack_car(car_row), CAR_SIZE, i_car)
    print(f"  ✓ บันทึกเช่าแล้ว (Amount {total:.2f})")
```

รูปภาพที่ 4-9 add_rent

4.5 ฟังก์ชัน update_entity()

4.5.1 ฟังก์ชัน update_entity() ทำหน้าที่ แก้ไข (Update) ข้อมูลที่มีอยู่ในระบบ แยกตามชนิดเอนทิตีที่ผู้ใช้เลือกได้ 3 แบบคือ ลูกค้า (customer), รถ (car) และ การเช่า (rental) พร้อมตรวจสอบความถูกต้องขั้นต่ำ และคำนวณค่าที่เกี่ยวข้องใหม่ก่อนบันทึกทับระเบียนเดิม อ่านข้อมูลของเอนทิตีนั้น ๆ จากไฟล์เก็บข้อมูล สร้างดัชนีค้นหา (index) ด้วยคีย์หลัก (เช่น customer_id, car_id, rental_id) รับ ID ที่ต้องการแก้ไข จากผู้ใช้ และตรวจสอบว่ามีอยู่จริงหรือไม่ ถ้าไม่พบ: แจ้ง “ไม่พบ” และยุติการทำงานส่วนนั้น ดึงระเบียนปัจจุบัน (current record) ออกมาเพื่อใช้อ้างอิง/กรอกค่าเริ่มต้น กรณีแก้ไข ลูกค้า (entity == "customer") ข้อมูลที่แก้ไขได้: ชื่อ (name), เบอร์โทรศัพท์ (tel) กระบวนการ แสดงค่าปัจจุบันเป็นค่าเริ่มต้นในพรอมพ์ หากผู้ใช้กด Enter เปล่า จะ คงค่าเดิม อัปเดตค่าใหม่กลับเข้าแถวข้อมูลลูกค้า บันทึกระเบียนกลับลงไฟล์ ณ ตำแหน่งเดิม ผลลัพธ์ที่แสดง: ข้อความยืนยันว่า “อัปเดตแล้ว” กรณีแก้ไข รถ (entity == "car") ข้อมูลที่แก้ไขได้: เลขทะเบียน (plate) – เวอร์ชันนี้เพิ่มให้แก้ไขได้ ยี่ห้อ (brand), รุ่น (model) ปี (year) ค่าเช่าต่อวัน (rate) สถานะ (status) : 1 = Active, 0 = Inactive กระบวนการ รับค่าจากผู้ใช้โดยแสดงค่าปัจจุบันเป็นค่าเริ่มต้น แปลงชนิดข้อมูลที่จำเป็น เช่น year, rate, status ให้เป็นจำนวนเต็ม เขียนค่าที่แก้ไขกลับเข้าระเบียน (รวมถึง plate ที่เพิ่งแก้) บันทึกระเบียนกลับลงไฟล์ ณ ตำแหน่งเดิม ข้อสังเกต: ไม่มีการคำนวณใด ๆ เพิ่มเติมในส่วนของรถ (นอกจากการแปลงชนิดข้อมูล) ผลลัพธ์ที่แสดง: ข้อความยืนยันว่า “อัปเดตแล้ว” กรณีแก้ไข การเช่า (entity == "rental") ข้อมูลที่แก้ไขได้: สถานะ (status) : 1 = Open, 0 = Closed, -1 = Deleted วันเริ่ม (start_ymd) และวันสิ้นสุด (end_ymd) — ผู้ใช้ใส่เป็นรูปแบบ YYYY-MM-DD หรือเว้นว่างเพื่อคงค่าเดิม กระบวนการ รับค่า สถานะใหม่ และวันที่เริ่ม/สิ้นสุด (ถ้ากรอก) หากผู้ใช้กรอกวันใหม่ จะทำการ แปลง วันที่รูปแบบ YYYY-MM-DD ให้เป็นค่าจำนวนเต็ม YYYYMMDD ตามโครงสร้างเดิมของระบบ ตรวจสอบเงื่อนไขสำคัญ: end >= start ถ้าไม่ผ่าน เงื่อนไข: แจ้ง “end < start” แล้ว ยุติการอัปเดต คำนวณค่าใหม่อัตโนมัติ total_days = จำนวนวันระหว่าง start–end (รวมปลายทาง +1 วัน) ค้นหาอัตราค่าเช่ารถ (rate) ของรถคันนั้นจากข้อมูลรถ แล้วคำนวณ total_amount = total_days * rate บันทึกระเบียนการเช่าที่แก้ไขแล้วกลับลงไฟล์ ณ ตำแหน่งเดิม อัปเดตสถานะรถที่เกี่ยวข้อง ถ้าสถานะการเช่าเป็น RENT_OPEN ให้ตั้ง is_rented = 1 มิฉะนั้น is_rented = 0 เขียนระเบียนรถกลับลงไฟล์เพื่อสะท้อนสถานะปัจจุบัน ผลลัพธ์ที่แสดง: ข้อความยืนยันว่า “อัปเดตแล้ว” (สำหรับ rental) หลังคำนวณและบันทึกสำเร็จ เงื่อนไขคุมคุณภาพที่สำคัญ ตรวจพบ ID ไม่อยู่ในระบบ ⟶ แจ้งเตือนและหยุดการทำงานเฉพาะส่วน

นั้น การอัปเดตวันเช่าต้อง ไม่สวนทาง (วันสิ้นสุดก่อนวันเริ่มต้นไม่ได้) เมื่อเปลี่ยนสถานะการเช่า จะซิงก์สถานะรถ เสมอเพื่อความสอดคล้องของข้อมูลจริง update_entity() เป็นฟังก์ชันแก้ไขข้อมูลที่ทำงานครบวงจร: ค้นหา ⟶ รับค่าใหม่ ⟶ ตรวจเงื่อนไข ⟶ คำนวณ (เฉพาะ rental) ⟶ บันทึกทับ ⟶ แจ้งผล ฝั่ง ลูกค้า: ปรับชื่อ/โทรศัพท์ ฝั่ง รถ: ปรับทะเบียน/ยี่ห้อ/รุ่น/ปี/ราคา/สถานะ ฝั่งเช่า: ปรับสถานะ/ช่วงวัน พร้อมคำนวณ จำนวนวันและยอดเงินใหม่ และอัปเดต สถานะรถกำลังถูกเช่าหรือไม่ ให้ตรงกับความเป็นจริงของระบบ

```python
def update_entity(entity: str):
    if entity == "customer":
        rows = read_all(CUST_PATH, CUSTOMER_FMT, CUSTOMER_FIELDS, CUSTOMER_SIZE)
        idx = index_by_key(rows, "customer_id")
        if not rows:
            print("  (ว่าง)"); return
        k = ask_int("Customer ID ที่จะแก้: ", 1)
        if k not in idx:
            print("  ! ไม่พบ"); return
        i, cur = idx[k]
        cur["name"] = input(f"Name [{cur['name']}]: ").strip() or cur["name"]
        cur["tel"]  = input(f"Tel  [{cur['tel']}]: ").strip() or cur["tel"]
        write_record_by_index(CUST_PATH, CUSTOMER_FMT, pack_customer(cur), CUSTOMER_SIZE, i)
        print("  ✓ อัปเดตแล้ว")

    elif entity == "car":
        rows = read_all(CAR_PATH, CAR_FMT, CAR_FIELDS, CAR_SIZE)
        idx = index_by_key(rows, "car_id")
        if not rows:
            print("  (ว่าง)"); return
        k = ask_int("Car ID ที่จะแก้: ", 1)
        if k not in idx:
            print("  ! ไม่พบ"); return
        i, cur = idx[k]

        # 👉 เพิ่มการแก้ไขป้ายทะเบียน (plate)
        plate = input(f"Plate [{cur['plate']}]: ").strip() or cur["plate"]

        brand = input(f"Brand [{cur['brand']}]: ").strip() or cur["brand"]
        model = input(f"Model [{cur['model']}]: ").strip() or cur["model"]
        year  = input(f"Year  [{cur['year']}]: ").strip()
        rate  = input(f"Rate  [{cur['rate']}]: ").strip()
        status= input(f"Status(1=Active,0=Inactive) [{cur['status']}]: ").strip()

        if year:    cur["year"]   = int(year)
        if rate:    cur["rate"]   = int(rate)
        if status != "": cur["status"] = int(status)

        cur["plate"] = plate             # <- บันทึกค่า plate ที่แก้
        cur["brand"], cur["model"] = brand, model

        write_record_by_index(CAR_PATH, CAR_FMT, pack_car(cur), CAR_SIZE, i)
        print("  ✓ อัปเดตแล้ว")

    elif entity == "rental":
        rows = read_all(RENT_PATH, RENT_FMT, RENT_FIELDS, RENT_SIZE)
        idx = index_by_key(rows, "rental_id")
        if not rows:
            print("  (ว่าง)"); return
        k = ask_int("Rental ID ที่จะแก้: ", 1)
        if k not in idx:
            print("  ! ไม่พบ"); return
        i, cur = idx[k]
        status = input(f"Status(1=Open,0=Closed,-1=Deleted) [{cur['status']}]: ").strip()
        if status!="": cur["status"] = int(status)
        s_in = input(f"Start [{ymd(cur['start_ymd'])} YYYY-MM-DD or blank]: ").strip()
        e_in = input(f"End   [{ymd(cur['end_ymd'])} YYYY-MM-DD or blank]: ").strip()
        if s_in:
            y,m,d = map(int, s_in.split("-")); cur["start_ymd"] = y*10000+m*100+d
        if e_in:
            y,m,d = map(int, e_in.split("-")); cur["end_ymd"] = y*10000+m*100+d
        sd, ed = ymd(cur["start_ymd"]), ymd(cur["end_ymd"])
        if ed < sd:
            print("  ! end < start"); return
        cur["total_days"] = (ed - sd).days + 1
        cars = read_all(CAR_PATH, CAR_FMT, CAR_FIELDS, CAR_SIZE)
        car_idx = index_by_key(cars, "car_id")
        rate = car_idx.get(cur["car_id"], (None, {"rate": 0}))[1]["rate"]
        cur["total_amount"] = float(cur["total_days"] * rate)
        write_record_by_index(RENT_PATH, RENT_FMT, pack_rent(cur), RENT_SIZE, i)
        print("  ✓ อัปเดตแล้ว")

        if cur["car_id"] in car_idx:
            i_car, car_row = car_idx[cur["car_id"]]
            car_row["is_rented"] = 1 if cur["status"] == RENT_OPEN else 0
            write_record_by_index(CAR_PATH, CAR_FMT, pack_car(car_row), CAR_SIZE, i_car)
```

รูปภาพที่ 4-10 update_entity

4.6 ฟังก์ชัน delete_entity()

4.6.1 ฟังก์ชัน delete_entity() ฟังก์ชันนี้ใช้สำหรับ “ลบ” ข้อมูลในระบบ โดยออกแบบแบบ ปลอดภัยต่อความเชื่อมโยงของข้อมูล (referential integrity) คือหลีกเลี่ยงการลบข้อมูลที่ยังเชื่อมกับรายการเช่า และใช้แนวคิด soft-delete / inactivate ตามความเหมาะสมของชนิดข้อมูล ตรวจสอบความเชื่อมโยงกับข้อมูลการเช่า (rental) ระบบอ่านรายการเช่าทั้งหมด แล้วเช็กว่ามีรายการเช่าที่อ้างอิง customer_id หรือ car_id เป้าหมายอยู่หรือไม่ (และสถานะรายการเช่านั้นยังไม่ถูกลบ) ถ้ามีความเชื่อมโยงอยู่ จะ ไม่อนุญาตให้ลบจริง และเปลี่ยนไปใช้การ “ปิดการใช้งาน (Inactive)” แทน (สำหรับรถ) หรือแจ้งข้ามการลบ (สำหรับลูกค้า) การลบ/เปลี่ยนสถานะ ตามชนิดข้อมูล กรณี entity == "customer" ระบบค้นหา Customer ID ที่ผู้ใช้ระบุ หากไม่พบบันทึก จะแจ้งว่า “ไม่พบ” หากพบ: ในโมเดลข้อมูลลูกค้า ไม่มีฟิลด์ status จึง ไม่ทำการลบจริง และจะแจ้งว่า “ทำเครื่องหมายลบแล้ว (ลูกค้าไม่มี status; ข้ามการลบ)” เพื่อรักษาความถูกต้องของประวัติธุรกรรม (ลูกค้ายังอยู่ในฐานข้อมูล) กรณี entity == "car" ระบบค้นหา Car ID ที่ผู้ใช้ระบุ หากไม่พบบันทึก จะแจ้งว่า “ไม่พบ” หากพบ: ระบบจะ ตั้งค่าสถานะรถเป็น Inactive (ปิดการใช้งาน) เพื่อ กันไม่ให้รถถูกปล่อยเช่าในอนาคต แต่ยังเก็บประวัติไว้ครบถ้วน กรณี entity == "rental" ระบบค้นหา Rental ID ที่ผู้ใช้ระบุ หากไม่พบบันทึก จะแจ้งว่า “ว่าง/ไม่พบ” หากพบ: ระบบจะทำ soft-delete โดยตั้งค่า status ของรายการเช่าเป็น RENT_DELETED (เช่น = -1) เพื่อเก็บร่องรอยธุรกรรมไว้โดยไม่ลบข้อมูลออกจากไฟล์จริง เขียนบันทึกกลับลงไฟล์ เมื่อมีการเปลี่ยนแปลง (เช่น set Inactive หรือ set Deleted) ระบบจะเขียนทับระเบียนเป้าหมาย ณ ตำแหน่งเดิม เพื่อให้โครงสร้างไฟล์คงที่และการค้นหาทำได้เร็ว

```python
def delete_entity(entity: str):
    if entity in ("customer", "car"):
        rents = read_all(RENT_PATH, RENT_FMT, RENT_FIELDS, RENT_SIZE)
        key = "customer_id" if entity == "customer" else "car_id"
        target_id = ask_int(
            f"{'Customer' if entity=='customer' else 'Car'} ID ที่จะลบ (จะตั้ง Inactive): ",
            1,
        )
        if any(r[key] == target_id and r["status"] != RENT_DELETED for r in rents):
            print("  ! ยังมีรายการเชื่อมโยงอยู่ (ห้ามลบ) -> จะตั้งสถานะ Inactive")
        if entity == "customer":
            rows = read_all(CUST_PATH, CUSTOMER_FMT, CUSTOMER_FIELDS, CUSTOMER_SIZE)
            idx = index_by_key(rows, "customer_id")
            if target_id not in idx:
                print("  ! ไม่พบ")
                return
            print("  ✓ ทำเครื่องหมายแล้ว (ลูกค้าไม่มี status; ข้ามการลบ)")
        else:
            rows = read_all(CAR_PATH, CAR_FMT, CAR_FIELDS, CAR_SIZE)
            idx = index_by_key(rows, "car_id")
            if target_id not in idx:
                print("  ! ไม่พบ")
                return
            i, cur = idx[target_id]
            cur["status"] = CAR_INACTIVE
            write_record_by_index(CAR_PATH, CAR_FMT, pack_car(cur), CAR_SIZE, i)
            print("  ✓ ตั้งรถเป็น Inactive แล้ว")
    else:
        rows = read_all(RENT_PATH, RENT_FMT, RENT_FIELDS, RENT_SIZE)
        idx = index_by_key(rows, "rental_id")
        if not rows:
            print("  (ว่าง)")
            return
        k = ask_int("Rental ID ที่จะลบ: ", 1)
        if k not in idx:
            print("  ! ไม่พบ")
            return
        i, cur = idx[k]
        cur["status"] = RENT_DELETED
        write_record_by_index(RENT_PATH, RENT_FMT, pack_rent(cur), RENT_SIZE, i)
        print("  ✓ ทำเครื่องหมายลบแล้ว (-1)")
```

รูปภาพที่ 4-11 delete_entity

4.7 ฟังก์ชัน view_one()

4.7.1 ฟังก์ชัน view_one() ฟังก์ชันนี้ใช้สำหรับ แสดงรายละเอียดของข้อมูลเพียง 1 รายการจากระบบ โดยผู้ใช้สามารถเลือกได้ว่าจะดูข้อมูลลูกค้า (Customer), ข้อมูลรถ (Car) หรือข้อมูลการเช่า (Rental) เลือกประเภทข้อมูลที่จะดู เรียก entity_select() เพื่อให้ผู้ใช้เลือกว่าต้องการดูข้อมูลประเภทใด หากเลือก 1 จะดู ลูกค้า, ถ้าเลือก 2 จะดู รถ, ถ้าเลือกอย่างอื่นจะดู การเช่า กรณีดูข้อมูลลูกค้า อ่านข้อมูลลูกค้าทั้งหมดจากไฟล์ สร้างดัชนี (index) โดยใช้ customer_id รับ Customer ID ที่ต้องการค้นหา แสดงผลรายละเอียดลูกค้าหากพบ แต่ถ้าไม่พบจะแสดงว่า “Not found” กรณีดูข้อมูลรถ อ่านข้อมูลรถทั้งหมดจากไฟล์ สร้างดัชนีโดยใช้ car_id รับ Car ID ที่ต้องการค้นหา แสดงผลรายละเอียดรถหากพบ แต่ถ้าไม่พบจะแสดงว่า “Not found” กรณีดูข้อมูลการเช่า อ่านข้อมูลการเช่าทั้งหมดจากไฟล์ สร้างดัชนีโดยใช้ rental_id รับ Rental ID ที่ต้องการค้นหา ถ้าไม่พบจะแสดงว่า “Not found” ถ้าพบ จะดึงข้อมูลการเช่ามาแสดง พร้อมแปลงค่า start_ymd และ end_ymd ให้อยู่ในรูปแบบวัน (start, end) ที่อ่านง่ายขึ้น แสดงผล ข้อมูลที่ค้นหามาจะถูกพิมพ์ออกมาให้ผู้ใช้เห็นทางหน้าจอ ฟังก์ชัน view_one() ช่วยให้ผู้ใช้สามารถ ค้นหาข้อมูลเฉพาะราย ได้สะดวก ไม่ว่าจะเป็นลูกค้า รถ หรือการเช่า โดยผู้ใช้ระบุ ID ที่ต้องการค้นหา หากพบระบบจะแสดงรายละเอียด หากไม่พบจะแสดงข้อความเตือนว่า “Not found” ซึ่งช่วยให้การตรวจสอบข้อมูลเป็นไปได้อย่างรวดเร็วและแม่นยำ

```python
def view_one():
    ent = entity_select()
    if ent == 1:
        rows = read_all(CUST_PATH, CUSTOMER_FMT, CUSTOMER_FIELDS, CUSTOMER_SIZE)
        idx = index_by_key(rows, "customer_id")
        k = ask_int("Customer ID: ", 1)
        r = idx.get(k)
        print(r[1] if r else "  ! Not found")
    elif ent == 2:
        rows = read_all(CAR_PATH, CAR_FMT, CAR_FIELDS, CAR_SIZE)
        idx = index_by_key(rows, "car_id")
        k = ask_int("Car ID: ", 1)
        r = idx.get(k)
        print(r[1] if r else "  ! Not found")
    else:
        rows = read_all(RENT_PATH, RENT_FMT, RENT_FIELDS, RENT_SIZE)
        idx = index_by_key(rows, "rental_id")
        k = ask_int("Rental ID: ", 1)
        r = idx.get(k)
        if not r:
            print("  ! Not found")
            return
        row = r[1].copy()
        row["start"], row["end"] = ymd(row["start_ymd"]), ymd(row["end_ymd"])
        print(row)
```

รูปภาพที่ 4-12 view_one

4.8 ฟังก์ชัน view_all()

4.8.1 ฟังก์ชัน view_all() ฟังก์ชันนี้ใช้สำหรับ แสดงข้อมูลทั้งหมดในระบบ ตามประเภทที่ผู้ใช้เลือกว่าจะดูข้อมูลลูกค้า (Customer), รถ (Car) หรือการเช่า (Rental) โดยข้อมูลจะแสดงในรูปแบบตาราง (Table) ที่อ่านง่าย เลือกประเภทข้อมูลที่จะดู เรียก entity_select() เพื่อให้ผู้ใช้เลือกประเภทข้อมูล หากเลือก 1 ⟶ แสดงข้อมูลลูกค้า หากเลือก 2 ⟶ แสดงข้อมูลรถ หากเลือกอย่างอื่น ⟶ แสดงข้อมูลการเช่า กรณีแสดงข้อมูลลูกค้า อ่านข้อมูลลูกค้าทั้งหมดจากไฟล์ กำหนดหัวตาราง (headers) ได้แก่: ID, ID Card, Name, Tel สร้างชุดข้อมูล (data) โดยดึงค่าจากฟิลด์ customer_id, id_card, name, และ tel ใช้ print_table(headers, data) เพื่อพิมพ์ข้อมูลออกมาเป็นตาราง กรณีแสดงข้อมูลรถ อ่านข้อมูลรถทั้งหมดจากไฟล์ กำหนดหัวตาราง ได้แก่: ID, Plate, Brand, Model, Year, Rate, Status, Rented สร้างชุดข้อมูล (data) โดยดึงค่าจากฟิลด์ต่าง ๆ เช่น car_id, plate, brand, model, year, rate แปลงสถานะรถ (status) ด้วยฟังก์ชัน car_status_label() เพื่อให้อ่านง่าย เช่น Active / Inactive แปลงสถานะการถูกเช่า (is_rented) เป็น “Yes” หรือ “No” แสดงผลออกมาเป็นตาราง กรณีแสดงข้อมูลการเช่า อ่านข้อมูลการเช่าทั้งหมดจากไฟล์ กำหนดหัวตาราง ได้แก่: RID, Car, Cust, Start, End, Days, Status, Amount สร้างชุดข้อมูล (data) โดยดึงค่าจากฟิลด์ เช่น rental_id, car_id, customer_id, start_ymd, end_ymd, total_days, status, total_amount วันที่ (start_ymd และ end_ymd) จะถูกแปลงให้อยู่ในรูปแบบวันที่ที่อ่านง่ายด้วย ymd() ค่า status จะถูกแปลงด้วย rent_status_label() เช่น Open, Closed, Deleted จำนวนเงิน (total_amount) ถูกจัดรูปแบบให้แสดงทศนิยม 2 ตำแหน่ง แสดงผลออกมาเป็นตาราง ฟังก์ชัน view_all() ช่วยให้ผู้ใช้สามารถ ดูข้อมูลทั้งหมดในระบบในรูปแบบตาราง โดยไม่ต้องค้นหาเป็นรายรายการ ทำให้ตรวจสอบและวิเคราะห์ข้อมูลได้ง่ายขึ้น ไม่ว่าจะเป็นข้อมูลลูกค้า ข้อมูลรถ หรือข้อมูลการเช่า ทั้งนี้ ระบบยังมีการแปลงสถานะและวันที่ให้อยู่ในรูปแบบที่อ่านง่าย ช่วยเพิ่มความสะดวกต่อการใช้งาน

```python
def view_all():
    ent = entity_select()
    if ent == 1:
        rows = read_all(CUST_PATH, CUSTOMER_FMT, CUSTOMER_FIELDS, CUSTOMER_SIZE)
        headers = ["ID", "ID Card", "Name", "Tel"]
        data = [[r["customer_id"], r["id_card"], r["name"], r["tel"]] for r in rows]
    elif ent == 2:
        rows = read_all(CAR_PATH, CAR_FMT, CAR_FIELDS, CAR_SIZE)
        headers = ["ID", "Plate", "Brand", "Model", "Year", "Rate", "Status", "Rented"]
        data = [
            [
                r["car_id"],
                r["plate"],
                r["brand"],
                r["model"],
                r["year"],
                r["rate"],
                car_status_label(r["status"]),
                "Yes" if r["is_rented"] else "No",
            ]
            for r in rows
        ]
    else:
        rows = read_all(RENT_PATH, RENT_FMT, RENT_FIELDS, RENT_SIZE)
        headers = ["RID", "Car", "Cust", "Start", "End", "Days", "Status", "Amount"]
        data = [
            [
                r["rental_id"],
                r["car_id"],
                r["customer_id"],
                ymd(r["start_ymd"]),
                ymd(r["end_ymd"]),
                r["total_days"],
                rent_status_label(r["status"]),
                f"{r['total_amount']:,.2f}",
            ]
            for r in rows
        ]
    print_table(headers, data)
```

รูปภาพที่ 4-13 view_all

4.9 ฟังก์ชัน view_stats()

4.9.1 ฟังก์ชัน view_stats() ฟังก์ชันนี้ใช้สำหรับ สรุปสถิติข้อมูลในระบบ เพื่อให้ผู้ใช้เห็นภาพรวมของจำนวนลูกค้า รถ และการเช่า พร้อมรายละเอียดสถานะต่าง ๆ ของรถและการเช่า อ่านข้อมูลทั้งหมดจากไฟล์ อ่านข้อมูลรถ (Cars) อ่านข้อมูลการเช่า (Rentals) อ่านข้อมูลลูกค้า (Customers) คำนวณจำนวนรถตามสถานะ Active ⟶ รถที่พร้อมให้เช่า Inactive ⟶ รถที่ถูกปิดการใช้งาน คำนวณจำนวนการเช่าตามสถานะ Open ⟶ รายการเช่าที่ยังเปิดอยู่ (ยังไม่สิ้นสุด) Closed ⟶ รายการเช่าที่สิ้นสุดแล้ว Deleted ⟶ รายการเช่าที่ถูกลบ (soft-delete) แสดงผลลัพธ์สรุป จำนวนลูกค้าทั้งหมด จำนวนรถทั้งหมด พร้อมแจกแจงว่า Active กี่คัน และ Inactive กี่คัน จำนวนการเช่าทั้งหมด พร้อมแจกแจงว่า Open, Closed, Deleted กี่รายการ ฟังก์ชัน view_stats() ทำหน้าที่เป็น แดชบอร์ดสรุปข้อมูล ที่ช่วยให้ผู้ใช้ทราบภาพรวมของระบบในทันที โดยไม่ต้องเปิดดูข้อมูลทีละรายการ เหมาะสำหรับการตรวจสอบสภาพโดยรวม เช่น มีลูกค้ากี่ราย รถที่พร้อมใช้งานกี่คัน และปริมาณการเช่าที่เกิดขึ้นในสถานะต่าง ๆ

```
def view_stats():
    cars = read_all(CAR_PATH, CAR_FMT, CAR_FIELDS, CAR_SIZE)
    rents = read_all(RENT_PATH, RENT_FMT, RENT_FIELDS, RENT_SIZE)
    custs = read_all(CUST_PATH, CUSTOMER_FMT, CUSTOMER_FIELDS, CUSTOMER_SIZE)
    car_active = sum(1 for c in cars if c["status"] == CAR_ACTIVE)
    car_inact = sum(1 for c in cars if c["status"] == CAR_INACTIVE)
    rent_open = sum(1 for r in rents if r["status"] == RENT_OPEN)
    rent_close = sum(1 for r in rents if r["status"] == RENT_CLOSED)
    rent_del = sum(1 for r in rents if r["status"] == RENT_DELETED)
    print("\n--- Summary Stats ---")
    print(f"Customers : {len(custs)}")
    print(f"Cars      : {len(cars)} (Active {car_active}, Inactive {car_inact})")
    print(
        f"Rentals   : {len(rents)} (Open {rent_open}, Closed {rent_close}, Deleted {rent_del})"
    )
```

รูปภาพที่ 4-14 view_stats

4.10 ฟังก์ชัน generate_report()

4.10.1 ฟังก์ชัน generate_report() ฟังก์ชันนี้มีหน้าที่ สร้างรายงานสรุป (Summary Report) เกี่ยวกับการเช่ารถ โดยรวมข้อมูลลูกค้า รถ และรายการเช่าเข้าด้วยกัน จากนั้นจัดรูปแบบให้อยู่ในตารางที่อ่านง่าย พร้อมทั้งสรุปสถิติที่สำคัญ และบันทึกเป็นไฟล์รายงาน อ่านข้อมูลจากไฟล์ข้อมูลลูกค้า (Customers) ข้อมูลรถ (Cars) ข้อมูลการเช่า (Rentals) และสร้างดัชนี (dict) สำหรับค้นหาลูกค้าและรถด้วย customer_id และ car_id กำหนดหัวตาราง (Table Headers) Rental ID Customer Name เบอร์โทร (Tel.) License Plate (ทะเบียนรถ) Brand, Model, Year Rate (ราคาต่อวัน) Rental Date (วันเช่า–คืน) Total Day (จำนวนวัน) Total Price (ราคารวม) ประมวลผลข้อมูลการเช่า ข้ามรายการที่ถูกลบ (RENT_DELETED) ดึงข้อมูลลูกค้าและรถที่เกี่ยวข้องกับการเช่า คำนวณวันเริ่มต้น (start_ymd), วันสิ้นสุด (end_ymd) และจำนวนวัน (total_days) คำนวณราคาทั้งหมด (total_amount) บันทึกข้อมูลในรูปแบบแถวตาราง (table row) การจัดรูปแบบ (Formatting) คำนวณความกว้างของแต่ละคอลัมน์ เพื่อให้จัดพิมพ์ตารางได้ตรงตำแหน่ง กำหนดรูปแบบวันที่ (ต่างกันระหว่าง Windows/Linux) กำหนดรูปแบบจำนวนเงินทศนิยม 2 หลัก สรุปสถิติ จำนวนลูกค้าทั้งหมด จำนวนรถทั้งหมด (Active/Inactive) จำนวนการเช่าทั้งหมด (Open, Closed, Deleted) อัตราค่าเช่ารถ (Rate) ⟶ ค่าต่ำสุด, สูงสุด, และค่าเฉลี่ย ความนิยมยี่ห้อรถ (นับจำนวน brand) เขียนไฟล์รายงาน แปลงผลลัพธ์ทั้งหมดเป็นข้อความ (string buffer) บันทึกออกเป็นไฟล์ข้อความ (.txt) โดยใช้ encoding UTF-8 สุดท้ายแจ้งผู้ใช้ว่า “สร้างรายงานแล้ว” พร้อมชื่อไฟล์ ฟังก์ชัน generate_report() มีบทบาทสำคัญในระบบ คือการ สรุปข้อมูลและสถิติทั้งหมด ของลูกค้า รถ และการเช่า ให้อยู่ในรูปแบบรายงานเดียว ทำให้ผู้ใช้หรือผู้บริหารสามารถเข้าใจภาพรวมของระบบได้ในครั้งเดียว ทั้งในเชิงรายละเอียดธุรกรรมและเชิงสถิติ

```python
def generate_report():
    customers = read_all(CUST_PATH, CUSTOMER_FMT, CUSTOMER_FIELDS, CUSTOMER_SIZE)
    cars = read_all(CAR_PATH, CAR_FMT, CAR_FIELDS, CAR_SIZE)
    rents = read_all(RENT_PATH, RENT_FMT, RENT_FIELDS, RENT_SIZE)

    cust_by = {c["customer_id"]: c for c in customers}
    car_by = {c["car_id"]: c for c in cars}

    headers = [
        "Rental_ID",
        "Customer name",
        "Tel.",
        "License_plate",
        "Brand",
        "Model",
        "Rate",
        "Date Rent",
        "Return date",
        "Rental Day",
        "Total Price",
    ]
    table = []
    for r in sorted(rents, key=lambda x: x["rental_id"]):
        if r["status"] == RENT_DELETED:
            continue
        cu = cust_by.get(r["customer_id"], {})
        ca = car_by.get(r["car_id"], {})
        sd, ed = ymd(r["start_ymd"]), ymd(r["end_ymd"])
        date_fmt = "%m/%d/%Y" if sys.platform == "win32" else "%-m/%-d/%Y"
        table.append(
            [
                r["rental_id"],
                cu.get("name", ""),
                cu.get("tel", ""),
                ca.get("plate", ""),
                ca.get("brand", ""),
                ca.get("model", ""),
                f"{ca.get('rate',0):,}",
                sd.strftime(date_fmt),
                ed.strftime(date_fmt),
                r["total_days"],
                f"{r['total_amount']:,.2f}",
            ]
        )

    widths = [len(h) for h in headers]
    for row in table:
        for i, c in enumerate(row):
            widths[i] = max(widths[i], len(str(c)))

    def fmt_row(cols):
        return " | ".join(str(c).ljust(widths[i]) for i, c in enumerate(cols))

    act_rates = [c["rate"] for c in cars if c["status"] == CAR_ACTIVE]
    minr = min(act_rates) if act_rates else 0
    maxr = max(act_rates) if act_rates else 0
    avgr = int(sum(act_rates) / len(act_rates)) if act_rates else 0
    brand_cnt = Counter(c["brand"] for c in cars)

    buf = io.StringIO()
    buf.write("Car Rent System - Summary Report\n")
    buf.write(f"Generated At: {datetime.now().strftime('%Y-%m-%d %H:%M:%S')}\n")
    buf.write(
        "App Version: 1.0\nEndianness: Little-Endian\nEncoding: UTF-8 (fixed-length)\n\n"
    )

    if table:
        buf.write(fmt_row(headers) + "\n")
        buf.write("-" * (sum(widths) + 3 * (len(widths) - 1)) + "\n")
        for row in table:
            buf.write(fmt_row(row) + "\n")
    else:
        buf.write("(no rentals)\n")

    rent_open = sum(1 for x in rents if x["status"] == RENT_OPEN)
    rent_close = sum(1 for x in rents if x["status"] == RENT_CLOSED)
    rent_del = sum(1 for x in rents if x["status"] == RENT_DELETED)

    buf.write("\n--- Summary ---\n\n")
    buf.write(f"Customers : {len(customers)}\n")
    buf.write(
        f"Cars      : {len(cars)} (Active {sum(1 for c in cars if c['status']==CAR_ACTIVE)}, Inactive {sum(1 for c in cars if c['status']==CAR_INACTIVE)})\n"
    )
    buf.write(
        f"Rentals   : {len(rents)} (Open {rent_open}, Closed {rent_close}, Deleted {rent_del})\n\n"
    )
    buf.write("Rate Statistics (Active cars only)\n")
    buf.write(
        f"- Min Rate : {minr:,}\n- Max Rate : {maxr:,}\n- Avg Rate : {avgr:,}\n\n"
    )
    buf.write("Cars by Brand\n")
    for b, n in sorted(brand_cnt.items()):
        buf.write(f"- {b} : {n}\n")

    with open(REPORT_PATH, "w", encoding="utf-8") as f:
        f.write(buf.getvalue())
    print(f"✓ สร้างรายงานแล้ว -> {REPORT_PATH}")
```

รูปภาพที่ 4-15 generate_report

4.11 ฟังก์ชัน main()

4.11.1 ฟังก์ชัน main() เป็น จุดเริ่มต้นและตัวควบคุมการทำงานหลัก (Program Controller / Main Loop) ของระบบเช่ารถ ทำหน้าที่เตรียมไฟล์ข้อมูล แสดงเมนูหลัก รับคำสั่งจากผู้ใช้ และเรียกใช้ฟังก์ชันย่อยที่เกี่ยวข้องตามตัวเลือก โดยโครงสร้างทำงานแบบ ลูปไม่สิ้นสุด จนกว่าผู้ใช้จะเลือกออกจากโปรแกรม เตรียมสภาพแวดล้อมข้อมูล เรียก ensure_files() เพื่อตรวจสอบ/สร้างไฟล์ฐานข้อมูลที่จำเป็นให้พร้อมก่อนเริ่มใช้งาน ลูปเมนูหลัก แสดงเมนูหลักด้วย main_menu() รับค่าตัวเลือกจากผู้ใช้ (Choose: 0–5) ด้วยการป้องกันความผิดพลาดของชนิดข้อมูล การทำงานตามตัวเลือก0 = ออกจากโปรแกรม ระบบจะพยายาม สร้างรายงานสรุป (generate_report()) เป็นขั้นตอนสุดท้ายเสมอ และพิมพ์ “Bye!” จากนั้นออกจากลูปและจบโปรแกรม 1 = เพิ่มข้อมูล (Create) เรียก entity_select() ให้ผู้ใช้ระบุชนิดข้อมูลที่ต้องการเพิ่ม เลือก 1 = customer ⟶ add_customer()เลือก 2 = car ⟶ add_car() เลือก 3 = ⟶ add_rental() แนวคิดคือ เมนูเดียวครอบคลุมการเพิ่มของทุกเอนทิตี 2 = แก้ไขข้อมูล (Update) เลือกชนิดข้อมูลด้วย entity_select() แล้วเรียก update_entity(<"customer"|"car"|"rental">) มีการตรวจสอบ ID, ตรวจเงื่อนไขความสอดคล้องของข้อมูล และคำนวณค่าที่เกี่ยวข้องใหม่ (เช่น จำนวนวัน/ยอดเงินของ rental) ก่อนบันทึกกลับ 3 = ลบ/ปิดการใช้งาน (Delete / Inactivate) เลือกชนิดข้อมูลด้วย entity_select() แล้วเรียก delete_entity(<"customer"|"car"|"rental">) ใช้แนวทาง ปลอดภัยต่อความเชื่อมโยงข้อมูล: rental ⟶ soft-delete (เปลี่ยนสถานะเป็น Deleted) car ⟶ ตั้งสถานะ Inactive หากยังเชื่อมกับรายการเช่า customer ⟶ ไม่มีสถานะให้ลบจริง จึงทำเครื่องหมาย/แจ้งเตือนแทน 4 = ดูข้อมูล (View) เข้าสู่ เมนูย่อย (ลูปภายใน) ด้วย view_menu() 0 ⟶ กลับเมนูหลัก 1 ⟶ view_one() แสดงรายการเดียวตาม ID 2 ⟶ view_all() แสดงข้อมูลทั้งหมดในรูปแบบตาราง อื่น ๆ ⟶ view_stats() แสดงสรุปสถิติภาพรวม (จำนวนลูกค้า, รถ Active/Inactive, เช่า Open/Closed/Deleted) 5 — สร้างรายงานทันที (ในโค้ดจัดเป็นกรณี “อื่น ๆ” ของเมนูหลัก) หากผู้ใช้เลือกเลขนอกช่วง 0–4 ที่เมนูแสดงไว้ ระบบจะตีความเป็น สั่งสร้างรายงาน (generate_report()) แล้วกลับเข้าสู่ลูปเมนูต่อไป main() คือหัวใจการทำงานของระบบ: ตรวจไฟล์ ⟶ วนเมนู ⟶ แยกการทำงาน (เพิ่ม/แก้ไข/ลบ/ดู/สรุปรายงาน) ตามที่ผู้ใช้เลือก และรับประกันว่าก่อนปิดโปรแกรมจะมีรายงานอัปเดตล่าสุด ถูกสร้างไว้เรียบร้อย ทำให้ทั้งการใช้งานประจำวันและการส่งมอบผลงานเป็นระบบและตรวจสอบย้อนหลังได้ง่าย

```python
def main():
    ensure_files()
    while True:
        main_menu()
        choice = ask_int("Choose: ", 0, 5)
        if choice == 0:
            try:
                generate_report()
            finally:
                print("Bye!")
            break
        elif choice == 1:
            ent = entity_select()
            (add_customer if ent == 1 else add_car if ent == 2 else add_rental)()
        elif choice == 2:
            ent = entity_select()
            update_entity("customer" if ent == 1 else "car" if ent == 2 else "rental")
        elif choice == 3:
            ent = entity_select()
            delete_entity("customer" if ent == 1 else "car" if ent == 2 else "rental")
        elif choice == 4:
            while True:
                sub = view_menu()
                if sub == 0:
                    break
                if sub == 1:
                    view_one()
                elif sub == 2:
                    view_all()
                else:
                    view_stats()
        else:
            generate_report()
```

รูปภาพที่ 4-16 main menu

บทที่ 5

สรุปผลการดำเนินงานและข้อเสนอแนะ

5.1 สรุปผลการดำเนินงาน

ระบบเช่ารถ ที่พัฒนาขึ้นสามารถช่วยจัดการข้อมูลรถ ข้อมูลสมาชิก และข้อมูลการเช่าได้อย่างมีประสิทธิภาพ โดยใช้การจัดเก็บข้อมูลแบบไฟล์ไบนารี พร้อมเมนู สำหรับเพิ่ม แก้ไข ลบ และแสดงผลข้อมูล ระบบยังรองรับการตรวจสอบสถานะรถที่ถูกเช่าแล้วหรือยังว่างอยู่ ตลอดจนการสร้างรายงานสรุปผลการดำเนินงาน เช่น จำนวนรถที่ถูกเช่า รายชื่อผู้เช่าปัจจุบัน และสถิติการใช้งานโดยรวม ซึ่งช่วยให้การบริหารจัดการ การเช่ารถให้สะดวก รวดเร็ว และลดความผิดพลาดจากการบันทึกแบบเดิมที่ใช้เอกสารกระดาษ

5.2 ปัญหาและอุปสรรคในการดำเนินงาน

ในการพัฒนาระบบเช่ารถ พบปัญหาหลักคือ ความซับซ้อนของการจัดการ ไฟล์ไบนารีที่ต้องใช้โครงสร้างข้อมูลคงที่ (struct) ซึ่งอาจเกิดข้อผิดพลาดหากการเข้ารหัสหรือถอดรหัสไม่ถูกต้อง นอกจากนี้ยังพบข้อจำกัดด้านการแสดงผลข้อมูล เช่น ความยาวของข้อมูล ความยากของระบบ การกำหนดขนาดของข้อมูลต่างๆ ซึ่งมีหลายรูปแบบและทำให้การทำงานซับซ้อน

5.3 ข้อเสนอแนะ

เพื่อให้ระบบสมบูรณ์และพร้อมใช้งานจริงในอนาคต ควรปรับปรุงดังนี้

5.3.1 พัฒนาให้รองรับฐานข้อมูลเชิงสัมพันธ์ (Relational Database) เช่น MySQL หรือ SQLite เพื่อรองรับข้อมูลจำนวนมากและการเข้าถึงหลายผู้ใช้งาน

5.3.2 พัฒนาเป็นโปรแกรมที่มีส่วนติดต่อผู้ใช้แบบกราฟิก (GUI) หรือเว็บแอปพลิเคชัน เพื่อ ความสะดวกในการใช้งานจริง

5.3.3 พัฒนาโปรแกรมเพิ่มระบบความปลอดภัยให้มากยิ่งขึ้น

5.3.4 พัฒนาโปรแกรม code ให้ทำงานง่าย ไม่ซับซ้อน ให้ครอบคลุมมากยิ่งขึ้น

5.4 สิ่งที่ผู้จัดทำได้รับในการพัฒนาโครงงาน

จากการพัฒนาโครงงานครั้งนี้ ผู้จัดทำได้รับความรู้และประสบการณ์ด้านการออกแบบระบบ การเขียนโปรแกรมด้วยภาษา Python การใช้โครงสร้างข้อมูลแบบไบนารี รวมถึงการคิดวิเคราะห์และแก้ไขปัญหาเชิงตรรกะ นอกจากนี้ยังได้ฝึกทักษะการทำงานเป็นทีม การแบ่งหน้าที่รับผิดชอบ และการจัดการเวลาให้สอดคล้องกับแผนงาน ทำให้ผู้จัดทำมีความเข้าใจในกระบวนการพัฒนาระบบซอฟต์แวร์มากยิ่งขึ้น และสามารถนำไปประยุกต์ใช้ในโครงการหรืองานจริงในอนาคตได้