SULIT

SECP2523 DATABASE

SESSION 2024/2025 - SEMESTER 1

ALTERNATIVE ASSESSMENT REPORT: PHASE 2

Name	Teh Ru Qian
IC No. / Matric No.	040517040256 / A23CS0191
Year / Program	2/SECPH
Section	Section 01
Lecturer Name	Ms. Rozilawati Binti Dollah @ Md. Zain
Case Study	Improvement of KKK Online System
System Name	Koperasi Kakitangan KADA (KKK) Online System
Group Name	Tech Hi-Five

Table of Contents

3.0 SQL and Interface Implementation

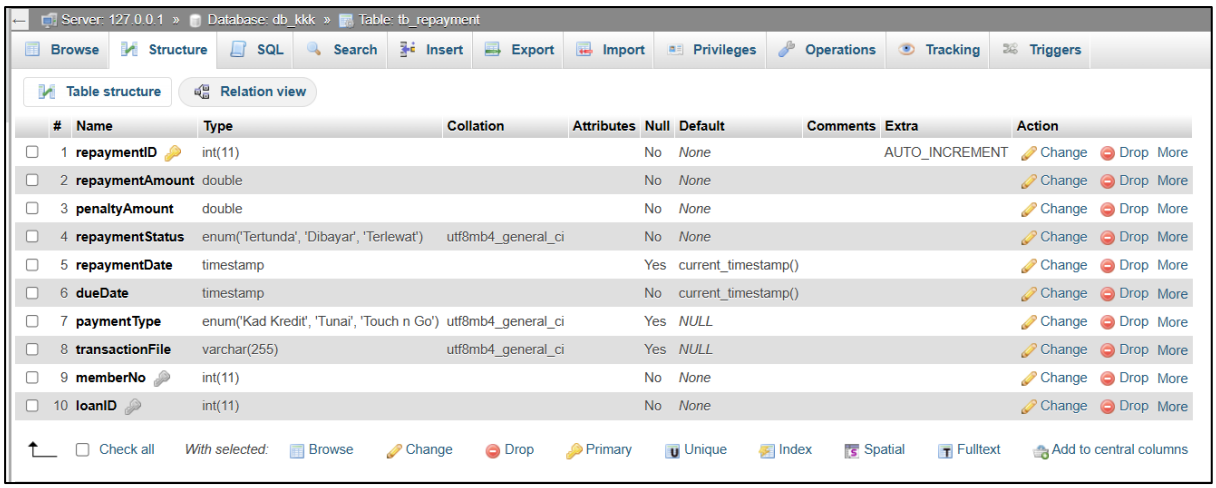
3.1 SQL Statement (DDL & DML)

Data Definition Language (DDL)

1. tb_repayment

```sql
-- Table structure for table `tb_repayment`
--

CREATE TABLE `tb_repayment` (
  `repaymentID` int(11) NOT NULL,
  `repaymentAmount` double NOT NULL,
  `penaltyAmount` double NOT NULL,
  `repaymentStatus` enum('Tertunda','Dibayar','Terlewat') NOT NULL,
  `repaymentDate` timestamp NULL DEFAULT current_timestamp(),
  `dueDate` timestamp NOT NULL DEFAULT current_timestamp(),
  `paymentType` enum('Kad Kredit','Tunai','Touch n Go') DEFAULT NULL,
  `transactionFile` varchar(255) DEFAULT NULL,
  `memberNo` int(11) NOT NULL,
  `loanID` int(11) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;

-- Indexes for table `tb_repayment`
--
ALTER TABLE `tb_repayment`
  ADD PRIMARY KEY (`repaymentID`),
  ADD KEY `memberNo` (`memberNo`),
  ADD KEY `loanID` (`loanID`);

-- Constraints for table `tb_repayment`
--
ALTER TABLE `tb_repayment`
  ADD CONSTRAINT `tb_repayment_ibfk_1` FOREIGN KEY (`memberNo`)
REFERENCES `tb_member` (`m_memberNo`),
  ADD CONSTRAINT `tb_repayment_ibfk_2` FOREIGN KEY (`loanID`)
REFERENCES `tb_loan` (`l_loanApplicationID`);

-- AUTO_INCREMENT for table `tb_repayment`
--
ALTER TABLE `tb_repayment`
  MODIFY `repaymentID` int(11) NOT NULL AUTO_INCREMENT;
```

Result for tb_repayment

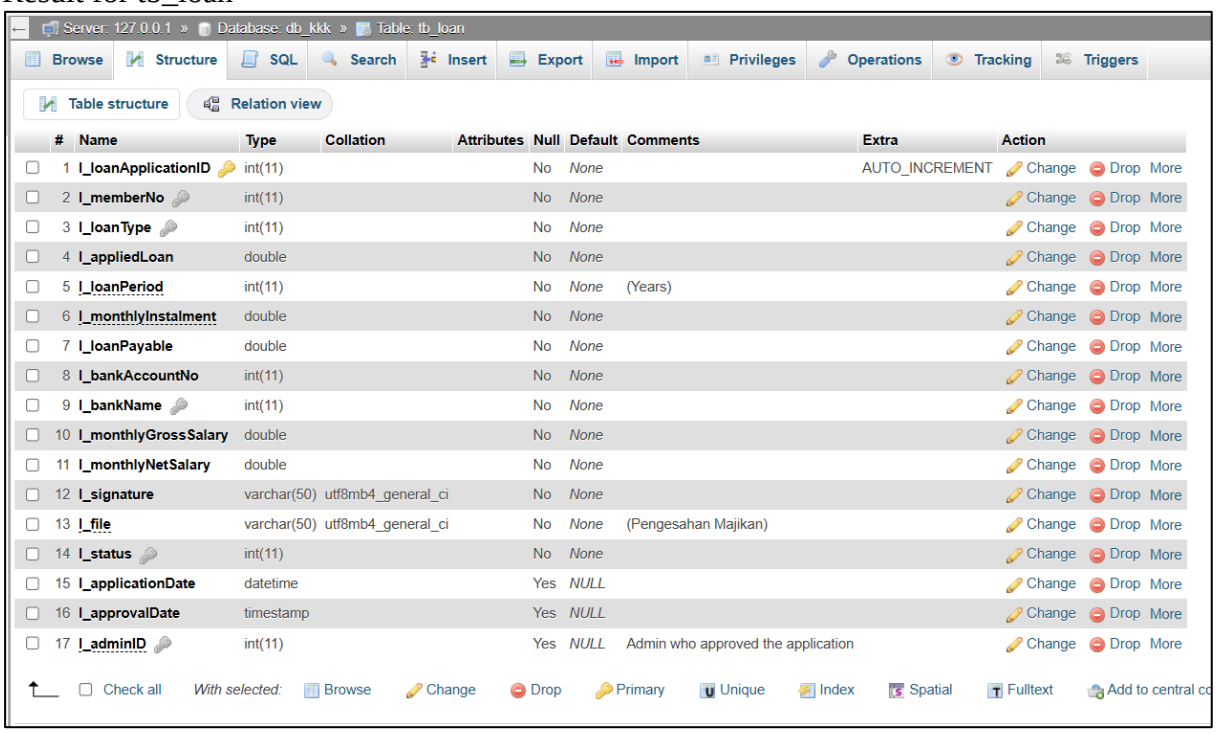


2. tb_loan

```sql
-- Table structure for table `tb_loan`
--

CREATE TABLE `tb_loan` (
  `l_loanApplicationID` int(11) NOT NULL,
  `l_memberNo` int(11) NOT NULL,
  `l_loanType` int(11) NOT NULL,
  `l_appliedLoan` double NOT NULL,
  `l_loanPeriod` int(11) NOT NULL COMMENT '(Years)',
  `l_monthlyInstalment` double NOT NULL COMMENT '\r\n',
  `l_loanPayable` double NOT NULL,
  `l_bankAccountNo` int(11) NOT NULL,
  `l_bankName` int(11) NOT NULL,
  `l_monthlyGrossSalary` double NOT NULL,
  `l_monthlyNetSalary` double NOT NULL,
  `l_signature` varchar(50) NOT NULL,
  `l_file` varchar(50) NOT NULL COMMENT '(Pengesahan Majikan)',
  `l_status` int(11) NOT NULL,
  `l_applicationDate` datetime DEFAULT NULL,
  `l_approvalDate` timestamp NULL DEFAULT NULL,
  `l_adminID` int(11) DEFAULT NULL COMMENT 'Admin who approved the
application'
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;

-- Indexes for table `tb_loan`
--
ALTER TABLE `tb_loan`
  ADD PRIMARY KEY (`l_loanApplicationID`),
  ADD KEY `l_loanType` (`l_loanType`),
  ADD KEY `l_adminID` (`l_adminID`),
  ADD KEY `l_status` (`l_status`),
  ADD KEY `l_memberNo` (`l_memberNo`),
  ADD KEY `l_bankName` (`l_bankName`);

-- Constraints for table `tb_loan`
--
ALTER TABLE `tb_loan`
  ADD CONSTRAINT `tb_loan_ibfk_1` FOREIGN KEY (`l_loanType`)
REFERENCES `tb_ltype` (`lt_lid`),
```

```
  ADD CONSTRAINT `tb_loan_ibfk_2` FOREIGN KEY (`l_status`) REFERENCES
`tb_status` (`s_sid`),
  ADD CONSTRAINT `tb_loan_ibfk_3` FOREIGN KEY (`l_memberNo`)
REFERENCES `tb_member` (`m_memberNo`),
  ADD CONSTRAINT `tb_loan_ibfk_4` FOREIGN KEY (`l_bankName`)
REFERENCES `tb_lbank` (`lb_id`);

-- AUTO_INCREMENT for table `tb_loan`
--
ALTER TABLE `tb_loan`
  MODIFY `l_loanApplicationID` int(11) NOT NULL AUTO_INCREMENT;

```

Result for tb_loan



Server: 127.0.0.1 » Database: db_kkk » Table: tb_loan

Browse | Structure | SQL | Search | Insert | Export | Import | Privileges | Operations | Tracking | Triggers

Table structure | Relation view

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	l_loanApplicationID	int(11)			No	None		AUTO_INCREMENT	Change Drop More
2	l_memberNo	int(11)			No	None			Change Drop More
3	l_loanType	int(11)			No	None			Change Drop More
4	l_appliedLoan	double			No	None			Change Drop More
5	l_loanPeriod	int(11)			No	None	(Years)		Change Drop More
6	l_monthlyInstalment	double			No	None			Change Drop More
7	l_loanPayable	double			No	None			Change Drop More
8	l_bankAccountNo	int(11)			No	None			Change Drop More
9	l_bankName	int(11)			No	None			Change Drop More
10	l_monthlyGrossSalary	double			No	None			Change Drop More
11	l_monthlyNetSalary	double			No	None			Change Drop More
12	l_signature	varchar(50)	utf8mb4_general_ci		No	None			Change Drop More
13	l_file	varchar(50)	utf8mb4_general_ci		No	None	(Pengesahan Majikan)		Change Drop More
14	l_status	int(11)			No	None			Change Drop More
15	l_applicationDate	datetime			Yes	NULL			Change Drop More
16	l_approvalDate	timestamp			Yes	NULL			Change Drop More
17	l_adminID	int(11)			Yes	NULL	Admin who approved the application		Change Drop More

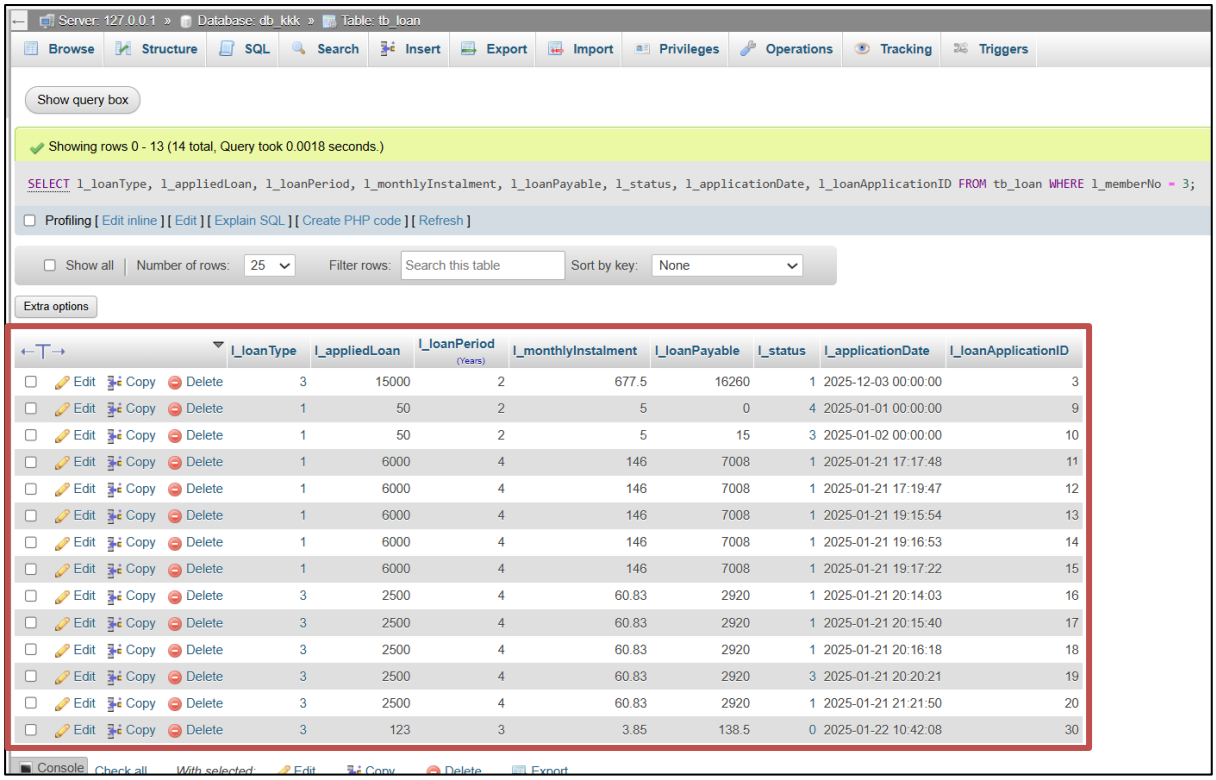
Check all With selected: Browse Change Drop Primary Unique Index Spatial Fulltext Add to central co

Data Manipulation Language (DML)

1. Retrieve loan details for a specific member, identified by $memberNo(3), from the tb_loan table in the database.

```
SELECT
    l_loanType,
    l_appliedLoan,
    l_loanPeriod,
    l_monthlyInstalment,
    l_loanPayable,
    l_status,
    l_applicationDate,
    l_loanApplicationID
FROM
    tb_loan
WHERE
    l_memberNo = 3;
```

Result

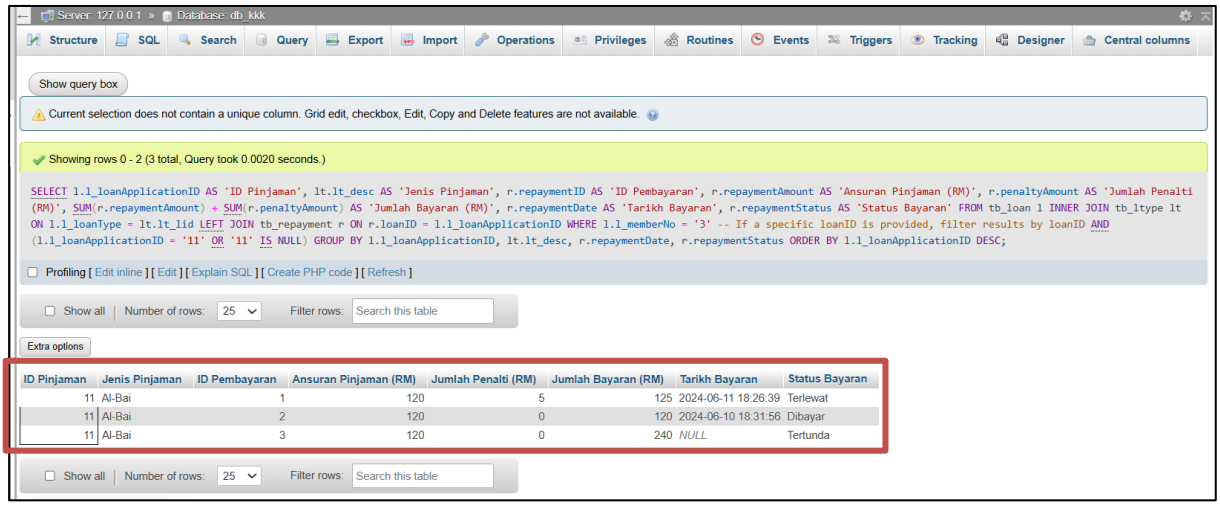


2. Retrieves detailed loan repayment information, including amounts, penalties, and statuses, for a specific member, $memberNo (3) with optional filtering by $loanID (11) and sorted by the most recent loans.

```
SELECT
    l.l_loanApplicationID AS 'ID Pinjaman',
    lt.lt_desc AS 'Jenis Pinjaman',
    r.repaymentID AS 'ID Pembayaran',
    r.repaymentAmount AS 'Ansuran Pinjaman (RM)',
    r.penaltyAmount AS 'Jumlah Penalti (RM)',
    SUM(r.repaymentAmount) + SUM(r.penaltyAmount) AS 'Jumlah Bayaran
(RM)',
    r.repaymentDate AS 'Tarikh Bayaran',
    r.repaymentStatus AS 'Status Bayaran'
FROM
    tb_loan l
INNER JOIN
    tb_ltype lt ON l.l_loanType = lt.lt_lid
LEFT JOIN
    tb_repayment r ON r.loanID = l.l_loanApplicationID
WHERE
    l.l_memberNo = '3'

    -- If a specific loanID is provided, filter results by loanID
    AND (l.l_loanApplicationID = '3' OR '$loanID' IS NULL)

GROUP BY
    l.l_loanApplicationID,
    lt.lt_desc,
    r.repaymentDate,
    r.repaymentStatus
```

```
28
29  ORDER BY
30      l.l_loanApplicationID DESC;
```

Result

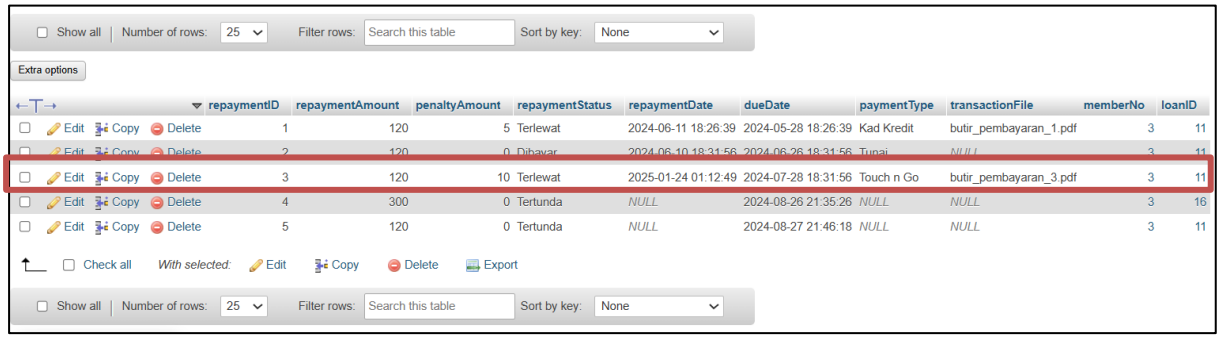


3. Updates the repayment details:
    - $repaymentDate (timestamp)
    - $transactionFile (butir_pembayaran_3.pdf)
    - $paymentType (Touch n Go)
    - $status (Terlewat)
    - $jumlahPenalti (10.00)

    in the tb_repayment table for a specific repayment record identified by $repaymentID (3)

```
1  UPDATE tb_repayment
2  SET
3      repaymentDate = 'timestamp',
4      transactionFile = 'butir_pembayaran_3.pdf',
5      paymentType = 'Touch n Go',
6      repaymentStatus = 'Terlewat',
7      penaltyAmount = '10.00'
8  WHERE
9      repaymentID = '3';
```

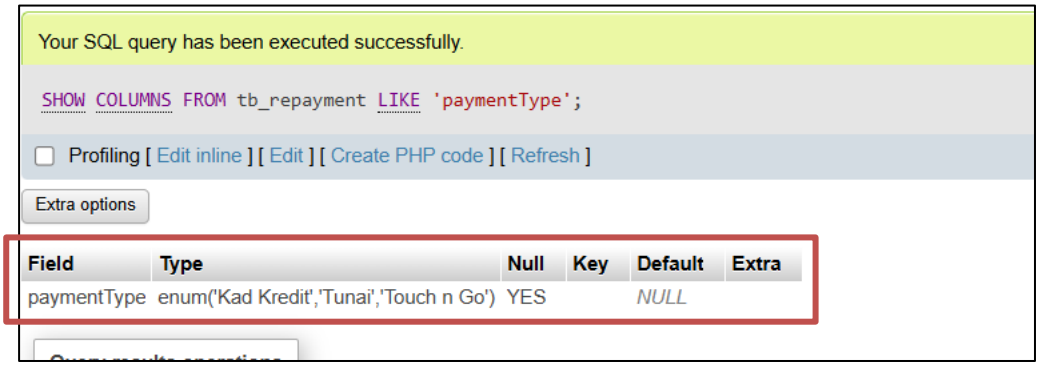
Result



4. Shows the paymentType column in the tb_repayment table and retrieves its possible ENUM values.

```
1  SHOW COLUMNS FROM tb_repayment LIKE 'paymentType';
```

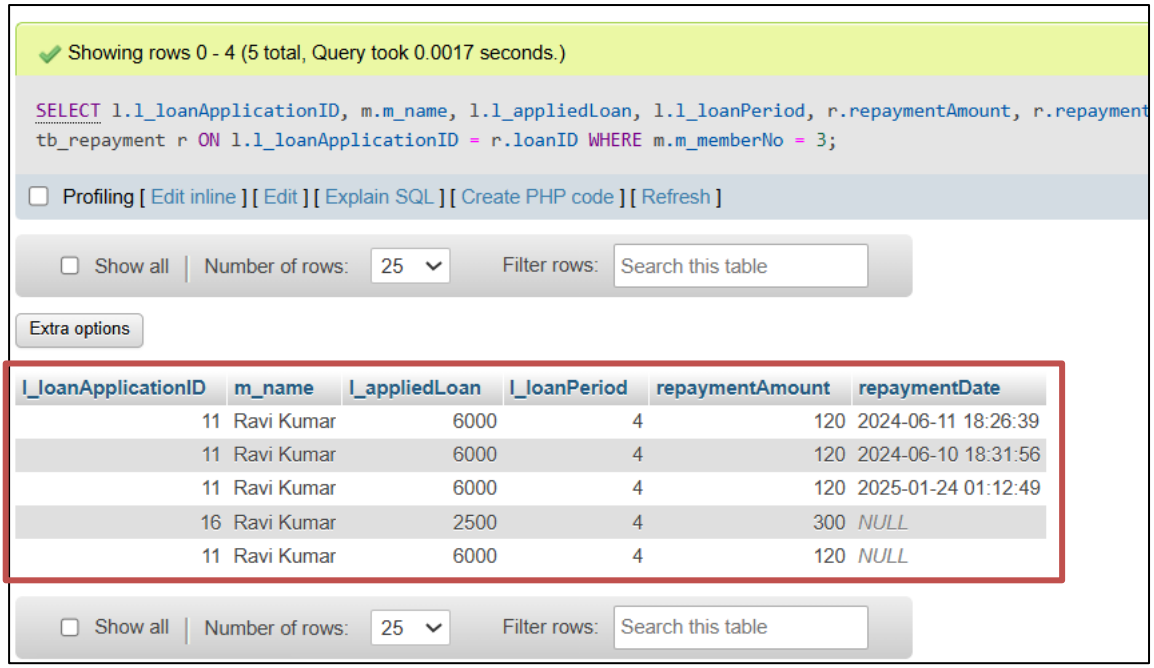
Result



5. Retrieve data about loans and their associated repayments for a specific member, $memberNo (3).

```
1  SELECT
2      l.l_loanApplicationID,
3      m.m_name,
4      l.l_appliedLoan,
5      l.l_loanPeriod,
6      r.repaymentAmount,
7      r.repaymentDate
8  FROM
9      tb_loan l
10 INNER JOIN
11     tb_member m ON l.l_memberNo = m.m_memberNo
12 INNER JOIN
13     tb_repayment r ON l.l_loanApplicationID = r.loanID
14 WHERE
15         m.m_memberNo = '3';
```

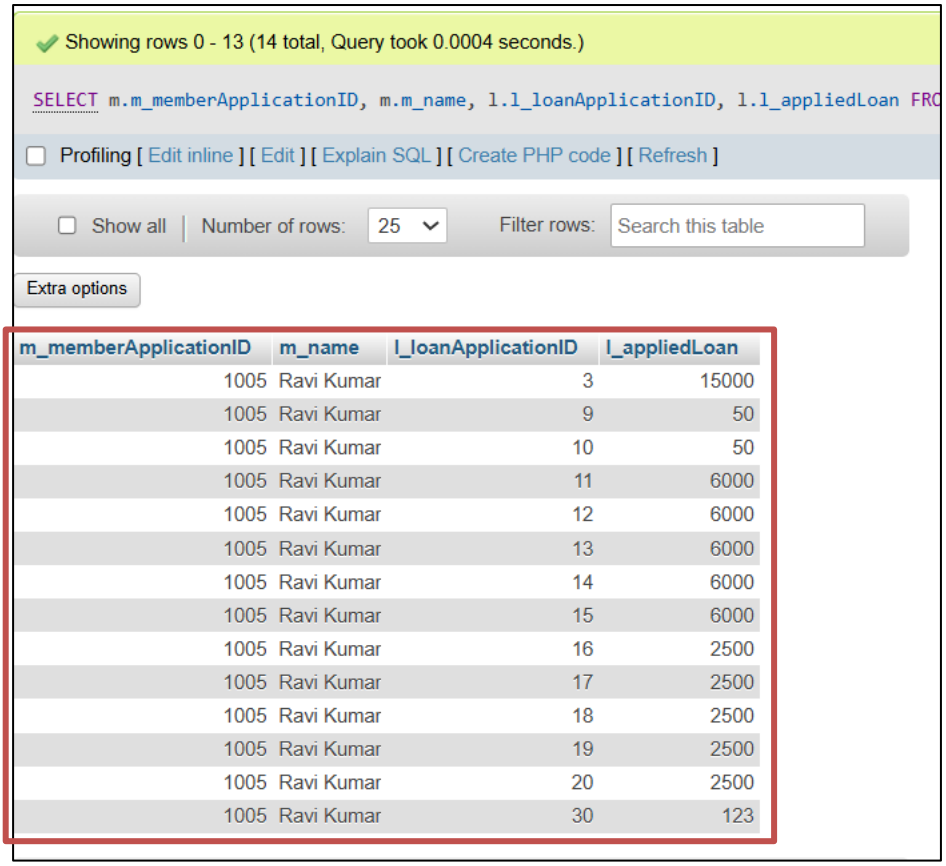
Result



6. Retrieves member details along with their loan information by performing a left join between the tb_member and tb_loan tables based on $memberNo (3).

```
SELECT
    m.m_memberApplicationID,
    m.m_name,
    l.l_loanApplicationID,
    l.l_appliedLoan
FROM
    tb_member m
LEFT JOIN
    tb_loan l ON m.m_memberNo = l.l_memberNo
WHERE
    m.m_memberNo = '3';
```

Result

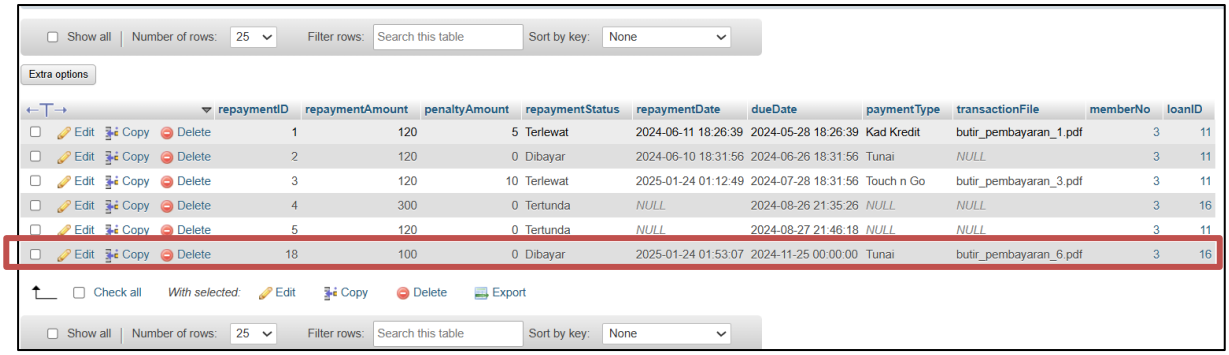


m_memberApplicationID	m_name	l_loanApplicationID	l_appliedLoan
1005	Ravi Kumar	3	15000
1005	Ravi Kumar	9	50
1005	Ravi Kumar	10	50
1005	Ravi Kumar	11	6000
1005	Ravi Kumar	12	6000
1005	Ravi Kumar	13	6000
1005	Ravi Kumar	14	6000
1005	Ravi Kumar	15	6000
1005	Ravi Kumar	16	2500
1005	Ravi Kumar	17	2500
1005	Ravi Kumar	18	2500
1005	Ravi Kumar	19	2500
1005	Ravi Kumar	20	2500
1005	Ravi Kumar	30	123

7. Inserts a new repayment record into the tb_repayment table with details such as the repayment amount, penalty amount, status, due date, payment type, transaction file, member number, and loan ID.

```
INSERT INTO tb_repayment (
    repaymentAmount,
    penaltyAmount,
    repaymentStatus,
    dueDate,
    paymentType,
    transactionFile,
    memberNo,
    loanID
)
VALUES (
    '100.00',
    '0.00',
    'Dibayar',
    '2024-11-25 00:00:00',
    'Tunai',
    'butir_pembayaran_6.pdf',
    '3',
    '16'
);
```

Result

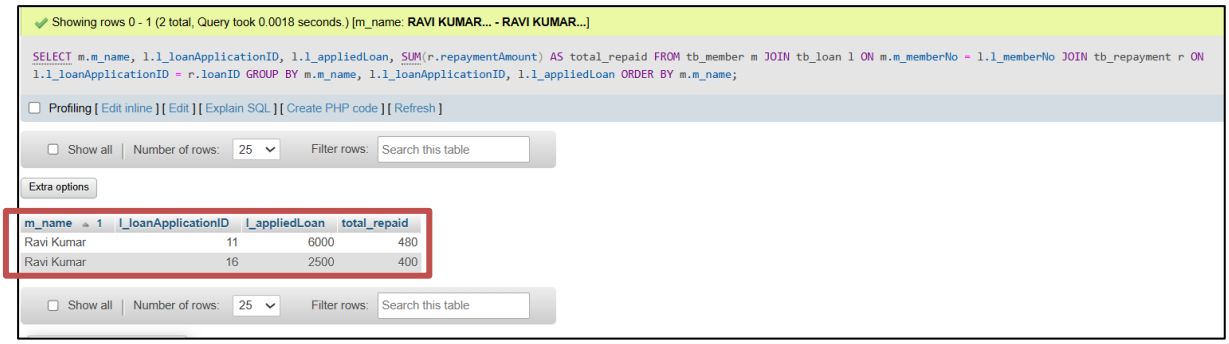


repaymentID	repaymentAmount	penaltyAmount	repaymentStatus	repaymentDate	dueDate	paymentType	transactionFile	memberNo	loanID
1	120	5	Terlewat	2024-06-11 18:26:39	2024-05-28 18:26:39	Kad Kredit	butir_pembayaran_1.pdf	3	11
2	120	0	Dibayar	2024-06-10 18:31:56	2024-06-26 18:31:56	Tunai	NULL	3	11
3	120	10	Terlewat	2025-01-24 01:12:49	2024-07-28 18:31:56	Touch n Go	butir_pembayaran_3.pdf	3	11
4	300	0	Tertunda	NULL	2024-08-26 21:35:26	NULL	NULL	3	16
5	120	0	Tertunda	NULL	2024-08-27 21:46:18	NULL	NULL	3	11
18	100	0	Dibayar	2025-01-24 01:53:07	2024-11-25 00:00:00	Tunai	butir_pembayaran_6.pdf	3	16

8. Retrieves the member's name, loan application ID, applied loan amount, and total repayment amount by joining the tb_member, tb_loan, and tb_repayment tables, grouped by member and loan, and ordered by member name.

```sql
SELECT
    m.m_name,
    l.l_loanApplicationID,
    l.l_appliedLoan,
    SUM(r.repaymentAmount) AS total_repaid
FROM
    tb_member m
JOIN
    tb_loan l ON m.m_memberNo = l.l_memberNo
JOIN
    tb_repayment r ON l.l_loanApplicationID = r.loanID
GROUP BY
    m.m_name,
    l.l_loanApplicationID,
    l.l_appliedLoan
ORDER BY
    m.m_name;
```

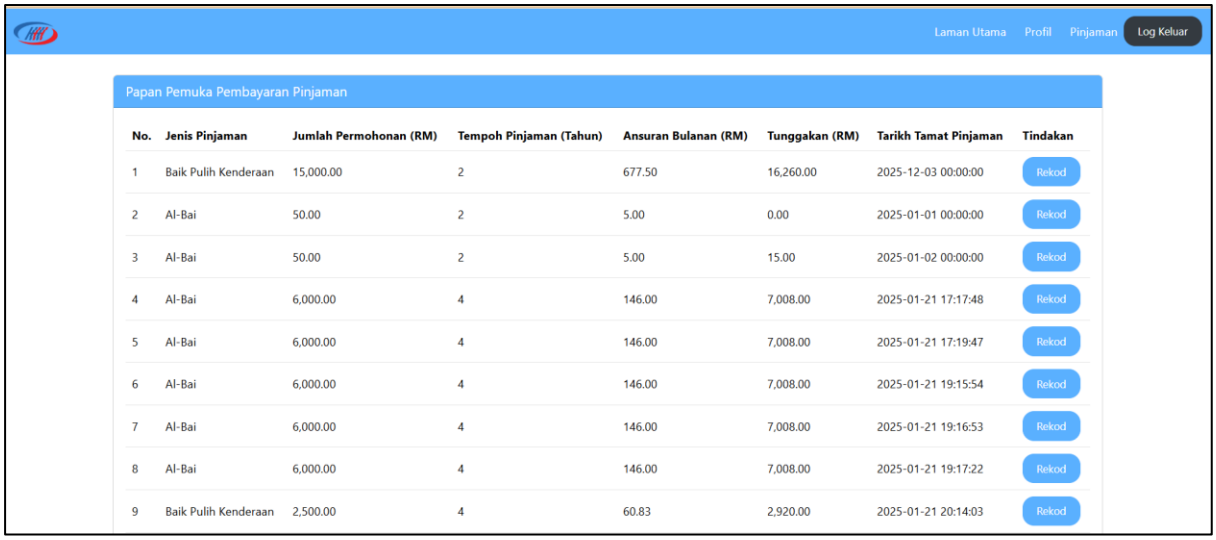
Result



m_name	l_loanApplicationID	l_appliedLoan	total_repaid
Ravi Kumar	11	6000	480
Ravi Kumar	16	2500	400

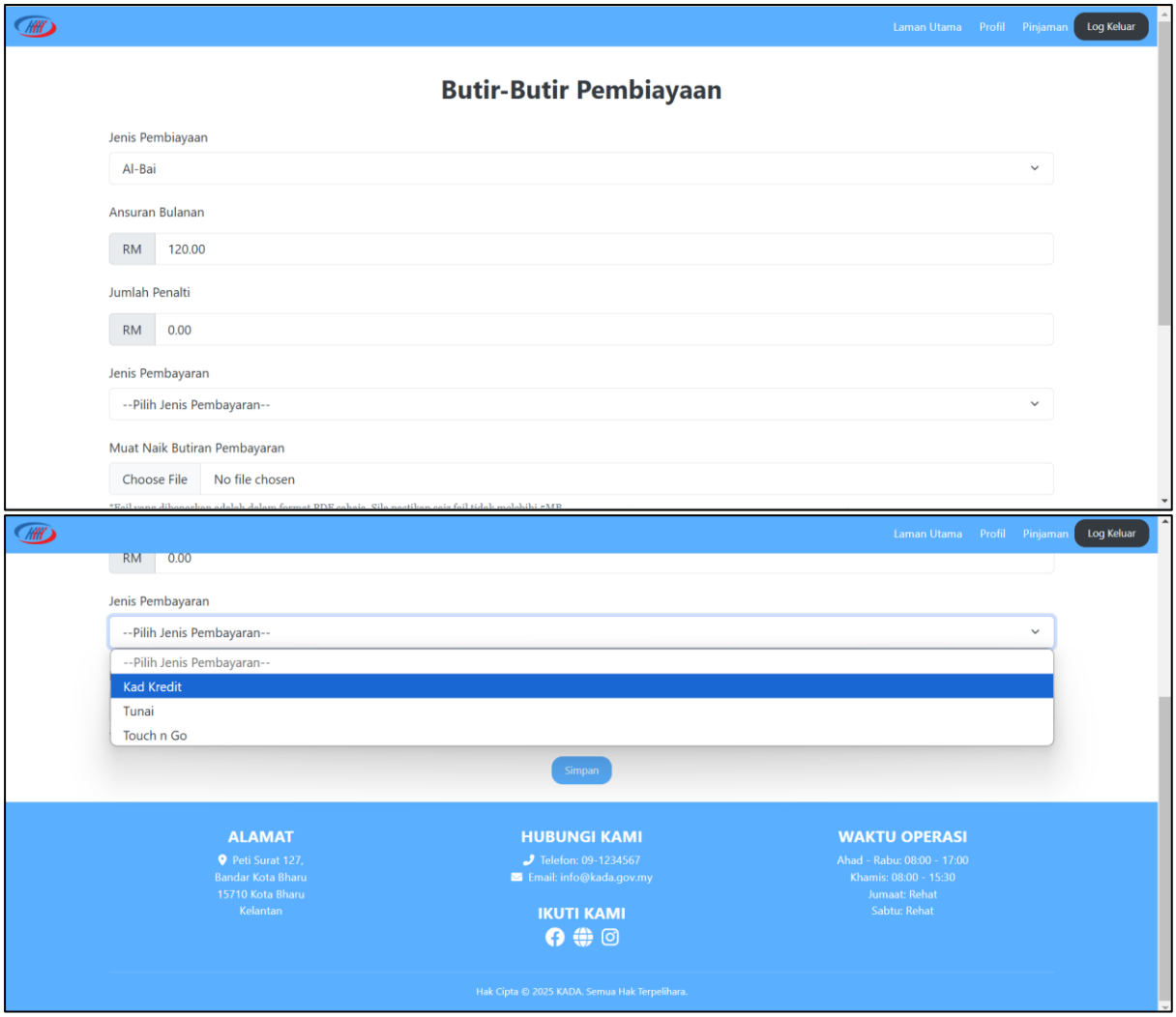
3.2 Interface Implementation

1. Loan Payment History Dashboard

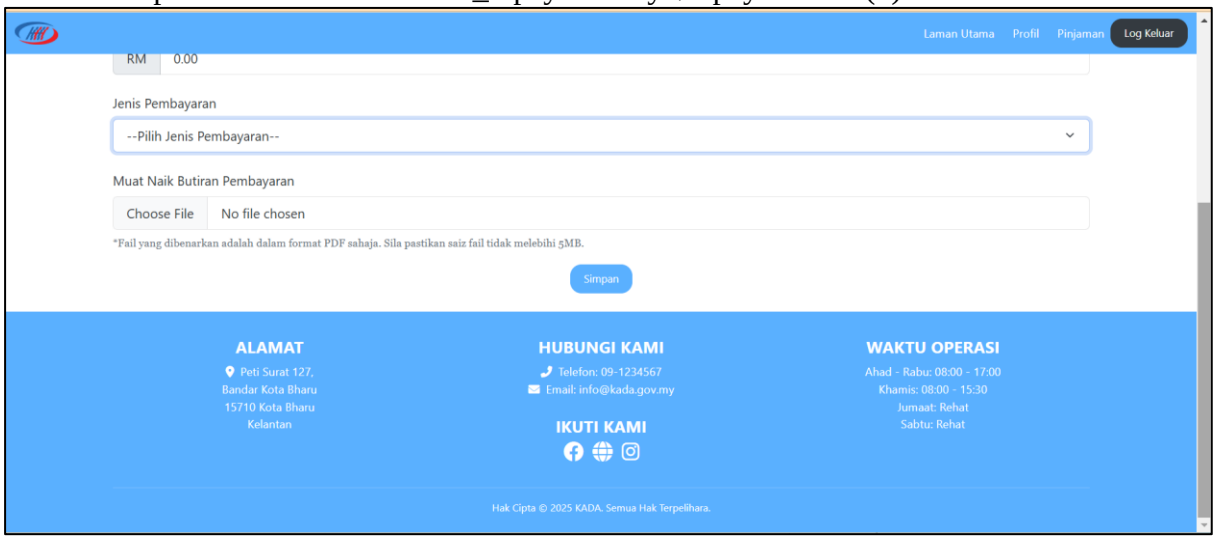


2. After click ‘Rekod’, it will retrieve loan repayment information for a member who had log in, $memberNo (3) by $loanID (11).

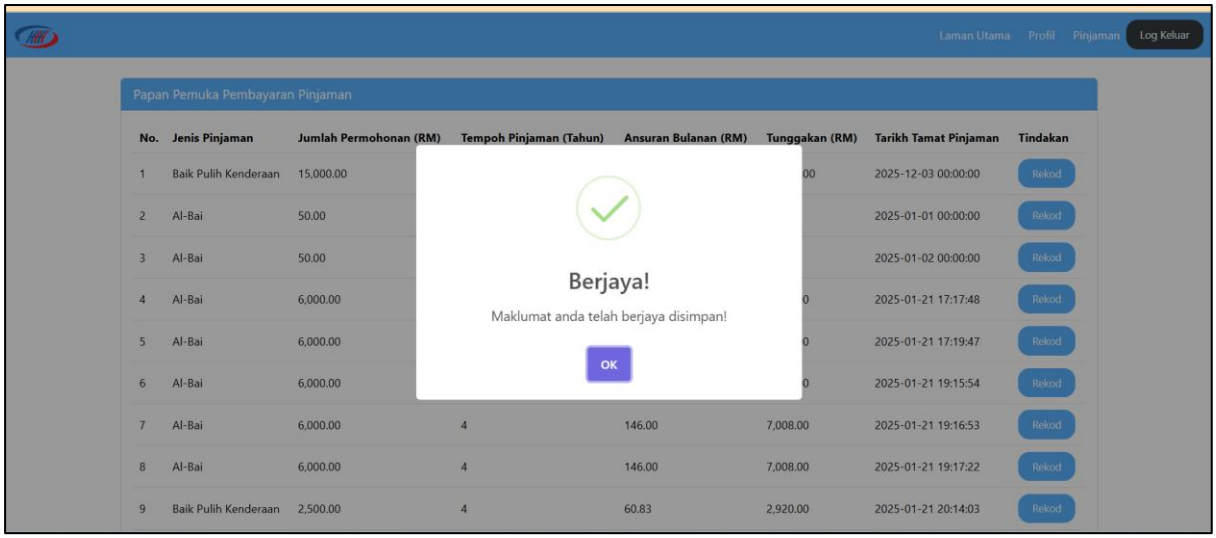
3. After click ‘Bayaran’, it will retrieve loan type and monthly instalment from tb_repayment by $loanID (11).



4. Update information into tb_repayment by $repaymentID (4).



5. Information update successfully.



4.0 Conclusion and Reflection

4.1 Conclusion

The Loan Repayment Module has successfully enhanced loan repayment management by allowing users to check repayment history and upload transaction files to make payments, reducing errors and saving time. However, it has several limitations, such as only supporting PDF files for uploads and requiring users to properly format their files. Future improvements to the module might include more file format compatibility, verification that uploaded files are accurate, and tools to assist users stay on track, like automated reminders and payment predictions. With these improvements, the module would be even more effective and user-friendly for everyone involved.

4.2 Reflection

The KKK Online System project aims to replace manual management procedures with an online platform, increasing efficiency and offering real-time data access to members and administrators. I responsible for Loan Module, which handles loan applications, repayment plans, and calculations of dynamic interest. Other team members contributed to the development of system functionalities by working on modules such as the Admin Approval, Admin Updating, User, and Member modules. We worked together to improve the system by simplifying procedures, enhancing communication, and assuring more efficient.

At the start of the project, I felt overwhelmed and confused where to begin. We had trouble understanding KADA's project requirements during our first discussions, which made it challenging to proceed. I felt more understood when we done many meetings as the project went on, but there were many challenges to overcome. Our progress was delayed by technical issues and the need to frequently reinstall PhpMyAdmin. Additionally, balancing this project with other projects and tests increased stress, making it more difficult to stay on track and complete tasks perfectly.

Despite these challenges, there were many positive aspects to the project. I learned how to retrieve, insert, and update data while gaining practical experience in the database design and website integration. However, the quality of our work was limited by time limitations and technical issues, especially during the testing phase. Better planning and organization would have prevented mistakes caused by the database system's problems and the lack of time.

In reflecting on this project, I've realized the value of time management, effective planning, and clear communication. I came to the realization that balancing different tasks can be

challenging, and I need to better manage my time so that I can complete each task very well. In order to make sure that every part of a project is finished to the greatest possible standard, I will give priority going forward to early testing, frequent backups, and improved time management. I've learned a lot from this experience that I'll use in my next projects, and I now understand how important proactive problem-solving and thorough preparation.