NAMA : Rionando Soeksin Putra
NIM : 11221063
Progress API MBD

script.sql

```sql
-- DB TABLE SETUP

-- USERS TABLE
CREATE TABLE users (
    id INT PRIMARY KEY,
    username VARCHAR(50) UNIQUE NOT NULL,
    fullname VARCHAR(100) NOT NULL,
    password VARCHAR(255) NOT NULL,
    is_admin BOOLEAN DEFAULT FALSE,
    registered_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);

-- TIPE_KENDARAAN TABLE
CREATE TABLE tipe_kendaraan (
    id INT PRIMARY KEY,
    tipe VARCHAR(50) UNIQUE NOT NULL
);

-- KENDARAAN TABLE
CREATE TABLE kendaraan (
    id INT PRIMARY KEY AUTO_INCREMENT,
    plat_nomor VARCHAR(50) UNIQUE NOT NULL,
    tersedia BOOLEAN DEFAULT TRUE,
    harga DECIMAL(10, 2) NOT NULL,
    tipe INT NOT NULL,

    FOREIGN KEY (tipe) REFERENCES tipe_kendaraan(id)
);

-- METODE_PEMBAYARAN TABLE
CREATE TABLE metode_pembayaran (
    id INT PRIMARY KEY AUTO_INCREMENT,
    nama VARCHAR(20) NOT NULL
);
```

```sql
-- SEWA TABLE
CREATE TABLE sewa (
    id INT PRIMARY KEY AUTO_INCREMENT,
    id_kendaraan INT NOT NULL,
    id_pelanggan INT NOT NULL,
    tanggal_mulai DATE NOT NULL,
    tanggal_selesai DATE NOT NULL,
    rusak BOOLEAN,
    tanggal_kembali DATE,
    biaya_denda DECIMAL(10, 2),
    total_bayar DECIMAL(10, 2),
    metode_pembayaran INT,

    FOREIGN KEY (id_kendaraan) REFERENCES kendaraan(id),
    FOREIGN KEY (id_pelanggan) REFERENCES users(id),
    FOREIGN KEY (metode_pembayaran) REFERENCES metode_pembayaran(id)
);



-- STORED FUNCTION SETUP
DELIMITER //

-- get or create tipe kendaraan berdasarkan input
DROP FUNCTION IF EXISTS get_or_create_tipe_kendaraan //
CREATE FUNCTION get_or_create_tipe_kendaraan(tipe_input VARCHAR(50))
RETURNS INT
DETERMINISTIC
BEGIN
    DECLARE tipe_id INT;

    SELECT id INTO tipe_id
    FROM tipe_kendaraan
    WHERE tipe = tipe_input
    LIMIT 1;

    IF tipe_id IS NULL THEN
        INSERT INTO tipe_kendaraan (tipe)
        VALUES (tipe_input);

        SELECT LAST_INSERT_ID() INTO tipe_id;
```

```
    END IF;

    RETURN tipe_id;
END //


-- get or create metode pembayaran berdasarkan user input
DROP FUNCTION IF EXISTS get_or_create_metode_pembayaran //
CREATE FUNCTION get_or_create_metode_pembayaran(nama_input
VARCHAR(20))
RETURNS INT
DETERMINISTIC
BEGIN
    DECLARE metode_id INT;

    SELECT id INTO metode_id
    FROM metode_pembayaran
    WHERE nama = nama_input
    LIMIT 1;

    IF metode_id IS NULL THEN
        INSERT INTO metode_pembayaran (nama)
        VALUES (nama_input);

        SELECT LAST_INSERT_ID() INTO metode_id;
    END IF;

    RETURN metode_id;
END //


-- fetch id kendaraan berdasarkan plat nomor
DROP FUNCTION IF EXISTS get_kendaraan_by_plat_nomor //
CREATE FUNCTION get_kendaraan_by_plat_nomor(plat_nomor_input
VARCHAR(50))
RETURNS INT
DETERMINISTIC
BEGIN
    DECLARE kendaraan_id INT;

    SELECT id INTO kendaraan_id
    FROM kendaraan
```

```
    WHERE plat_nomor = plat_nomor_input
    LIMIT 1;

    RETURN kendaraan_id;
END //




-- GET TOTAL REVENUE (KENDARAAN)
DROP FUNCTION IF EXISTS get_total_revenue_by_kendaraan //

CREATE FUNCTION get_total_revenue_by_kendaraan(
    plat_nomor_input VARCHAR(50)
)
RETURNS DECIMAL(10, 2)
DETERMINISTIC
BEGIN
    DECLARE total_revenue DECIMAL(10, 2);

    DECLARE id_kendaraan_input INT;
    SET id_kendaraan_input =
get_kendaraan_by_plat_nomor(plat_nomor_input);

    IF id_kendaraan_input IS NULL THEN
        RETURN 0;
    END IF;

    SELECT IFNULL(SUM(total_bayar), 0) INTO total_revenue
    FROM sewa
    WHERE id_kendaraan = id_kendaraan_input;

    RETURN total_revenue;
END //



-- GET TOTAL REVENUE (USER)
DROP FUNCTION IF EXISTS get_total_revenue_by_user //
CREATE FUNCTION get_total_revenue_by_user(
    id_pelanggan_input INT
)
RETURNS DECIMAL(10, 2)
```

```
DETERMINISTIC
BEGIN
    DECLARE total_spending DECIMAL(10, 2);

    SELECT IFNULL(SUM(total_bayar), 0) INTO total_spending
    FROM sewa
    WHERE id_pelanggan = id_pelanggan_input;

    RETURN total_spending;
END //

DELIMITER ;




-- STORED PROCEDURE SETUP
DELIMITER //

-- Fetch user from db
DROP PROCEDURE IF EXISTS get_user //
CREATE PROCEDURE get_user(IN in_email VARCHAR(50))
BEGIN
    START TRANSACTION;
    SELECT id, username, password, is_admin
    FROM users
    WHERE username = in_email;
    COMMIT;
END //

-- Create new user
DROP PROCEDURE IF EXISTS create_user //
CREATE PROCEDURE create_user(
    IN in_email VARCHAR(50),
    IN in_fullname VARCHAR(100),
    IN in_password VARCHAR(255),
    IN in_is_admin BOOLEAN
)
BEGIN
    START TRANSACTION;
    INSERT INTO users (email, fullname, password, is_admin)
    VALUES (in_email, in_fullname, in_password, in_is_admin);
```

```
    COMMIT;
END //

-- Tambah kendaraan baru
DROP PROCEDURE IF EXISTS tambah_kendaraan //
CREATE PROCEDURE tambah_kendaraan(
    IN plat_nomor_input VARCHAR(50),
    IN harga_input DECIMAL(10, 2),
    IN tipe_input VARCHAR(50)
)
BEGIN
    DECLARE tipe_id INT;

    SET tipe_id = get_or_crate_tipe_kendaraan(tipe_input);

    START TRANSACTION;
    INSERT INTO kendaraan (plat_nomor, harga, tipe)
    VALUES (plat_nomor_input, harga_input, tipe_id);
    COMMIT;
END //

-- Hapus kendaraan
DROP PROCEDURE IF EXISTS hapus_kendaraan //
CREATE PROCEDURE hapus_kendaraan(
    IN plat_nomor_input VARCHAR(50)
)
BEGIN
    START TRANSACTION;
    DELETE FROM kendaraan
    WHERE plat_nomor = plat_nomor_input;
    COMMIT;
END //




-- Sewa kendaraan
DROP PROCEDURE IF EXISTS sewa_kendaraan //
CREATE PROCEDURE sewa_kendaraan(
    IN plat_nomor_input VARCHAR(50),
    IN id_pelanggan_input INT,
    IN tanggal_mulai_input DATE,
```

```
    IN tanggal_selesai_input DATE
)
BEGIN
    DECLARE id_kendaraan_input INT;
    DECLARE harga_kendaraan DECIMAL(10, 2);
    DECLARE total_hari INT;
    DECLARE total_bayar DECIMAL(10, 2);
    DECLARE kendaraan_tersedia BOOLEAN;

    SET id_kendaraan_input =
get_kendaraan_by_plat_nomor(plat_nomor_input);

    IF id_kendaraan_input IS NULL THEN
        SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Kendaraan tidak
ditemukan.';
    END IF;

    SELECT tersedia INTO kendaraan_tersedia
    FROM kendaraan
    WHERE id = id_kendaraan_input;

    IF NOT kendaraan_tersedia THEN
        SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Kendaraan tidak
tersedia untuk periode sewa ini.';
    END IF;

    SELECT harga INTO harga_kendaraan
    FROM kendaraan
    WHERE id = id_kendaraan_input;

    SET total_hari = DATEDIFF(tanggal_selesai_input,
tanggal_mulai_input);

    SET total_bayar = harga_kendaraan * total_hari;

    START TRANSACTION;
    INSERT INTO sewa (
        id_kendaraan,
        id_pelanggan,
        tanggal_mulai,
        tanggal_selesai,
        total_bayar
```

```
    )
    VALUES (
        id_kendaraan_input,
        id_pelanggan_input,
        tanggal_mulai_input,
        tanggal_selesai_input,
        total_bayar
    );

    UPDATE kendaraan
    SET tersedia = FALSE
    WHERE id = id_kendaraan_input;

    COMMIT;

END //




-- Kembalikan kendaraan
DROP PROCEDURE IF EXISTS kembalikan_kendaraan //

CREATE PROCEDURE kembalikan_kendaraan(
    IN plat_nomor_input VARCHAR(50),
    IN tanggal_kembali_input DATE,
    IN rusak_input BOOLEAN,
    IN metode_pembayaran_input VARCHAR(50)
)
BEGIN
    DECLARE id_kendaraan_input INT;
    DECLARE harga_kendaraan DECIMAL(10, 2);
    DECLARE tanggal_sewa_mulai DATE;
    DECLARE tanggal_sewa_selesai DATE;
    DECLARE total_bayar DECIMAL(10, 2);
    DECLARE biaya_denda DECIMAL(10, 2);
    DECLARE total_hari INT;
    DECLARE total_hari_terlambat INT;
    DECLARE sewa_id INT;
    DECLARE metode_pembayaran_id INT;
```

```
    SET id_kendaraan_input =
get_kendaraan_by_plat_nomor(plat_nomor_input);

    IF id_kendaraan_input IS NULL THEN
        SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Kendaraan tidak
ditemukan.';
    END IF;

    SELECT id, tanggal_mulai, tanggal_selesai INTO sewa_id,
tanggal_sewa_mulai, tanggal_sewa_selesai
    FROM sewa
    WHERE id_kendaraan = id_kendaraan_input AND tanggal_kembali IS
NULL;

    IF tanggal_sewa_mulai IS NULL THEN
        SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Sewa kendaraan
tidak ditemukan.';
    END IF;

    SELECT harga INTO harga_kendaraan
    FROM kendaraan
    WHERE id = id_kendaraan_input;

    SET total_hari_terlambat = DATEDIFF(tanggal_kembali_input,
tanggal_sewa_selesai);

    IF total_hari_terlambat > 0 THEN
        SET biaya_denda = total_hari_terlambat * 50000;
    ELSE
        SET biaya_denda = 0;
    END IF;

    IF rusak_input THEN
        SET biaya_denda = biaya_denda + 100000;
    END IF;

    SET total_hari = DATEDIFF(tanggal_sewa_selesai,
tanggal_sewa_mulai);

    SET total_bayar = (harga_kendaraan * total_hari) + biaya_denda;

    SET metode_pembayaran_id =
```

```
get_or_create_metode_pembayaran(metode_pembayaran_input);

    START TRANSACTION;
    UPDATE sewa
    SET tanggal_kembali = tanggal_kembali_input,
        total_bayar = total_bayar,
        biaya_denda = biaya_denda,
        metode_pembayaran = metode_pembayaran_id,
        rusak = rusak_input
    WHERE id = sewa_id;

    UPDATE kendaraan
    SET tersedia = TRUE
    WHERE id = id_kendaraan_input;
    COMMIT;

END //




-- GET SEWA HISTORY (Kendaraan)
DROP PROCEDURE IF EXISTS get_sewa_history_kendaraan //

CREATE PROCEDURE get_sewa_history_kendaraan(
    IN plat_nomor_input VARCHAR(50)
)
BEGIN
    DECLARE id_kendaraan_input INT;

    SET id_kendaraan_input =
get_kendaraan_by_plat_nomor(plat_nomor_input);

    IF id_kendaraan_input IS NULL THEN
        SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Kendaraan tidak
ditemukan.';
    END IF;

    SELECT
        s.id AS sewa_id,
        s.id_kendaraan,
        k.plat_nomor,
```

```
        s.id_pelanggan,
        s.tanggal_mulai,
        s.tanggal_selesai,
        s.tanggal_kembali,
        s.total_bayar,
        s.biaya_denda,
        m.nama as metode_pembayaran,
        s.rusak
    FROM
        sewa s
    JOIN
        kendaraan k ON s.id_kendaraan = k.id
    LEFT JOIN
        metode_pembayaran m ON s.metode_pembayaran = m.id
    WHERE
        k.plat_nomor = plat_nomor_input
    ORDER BY
        s.tanggal_mulai DESC;

END //


-- GET SEWA HISTORY (User)
DROP PROCEDURE IF EXISTS get_sewa_history_user //

CREATE PROCEDURE get_sewa_history_user(
    IN id_pelanggan_input INT
)
BEGIN

    SELECT
        k.plat_nomor,
        s.tanggal_mulai,
        s.tanggal_selesai,
        s.rusak,
        s.total_bayar,
        m.nama
    FROM
        sewa s
    JOIN
        kendaraan k ON s.id_kendaraan = k.id
    LEFT JOIN
```

```
        metode_pembayaran m ON s.metode_pembayaran = m.id
    WHERE
        s.id_pelanggan = id_pelanggan_input
    ORDER BY
        s.tanggal_mulai DESC;

END //



DELIMITER ;




-- USAGE

-- CALL get_sewa_history_kendaraan('KU 2713 GI')
-- CALL get_sewa_history_user(3)

-- CALL sewa_kendaraan('KU 2713 GI', 3, '2024-11-20', '2024-11-25')
-- CALL kembalikan_kendaraan('KU 2713 GI', '2024-11-25', FALSE,
'cash')

-- SELECT get_total_revenue_by_user(3)
-- SELECT get_total_revenue_by_kendaraan('KU 2713 GI')
```

main.py

```
from flask import Flask
from flask_restful import Api
from db import mysql
from resources.user import User
from resources.auth import Login
from resources.kendaraan import Kendaraan
from resources.tipe_kendaraan import TipeKendaraan
from resources.sewa import Sewa
from resources.kembali import Kembali
from resources.history import History
```

```python
app = Flask(__name__)
api = Api(app)

app.config['MYSQL_HOST'] = 'localhost'
app.config['MYSQL_USER'] = 'root'
app.config['MYSQL_PASSWORD'] = ''
app.config['MYSQL_DB'] = 'kendaraan_db'

mysql.init_app(app)

api.add_resource(User, '/users/')
api.add_resource(Login, '/auth/login/')
api.add_resource(Kendaraan, '/kendaraan/')
api.add_resource(Sewa, '/sewa/')
api.add_resource(Kembali, '/kembali/')
api.add_resource(History, '/history/<string:history_type>/')


if __name__ == '__main__':
    for rule in app.url_map.iter_rules():
        print(f"{rule.endpoint}: {rule.methods} {rule}")
    app.run(debug=True)
```

db.py

```python
from flask_mysqldb import MySQL

mysql = MySQL()
```

auth.py

```python
from flask_restful import Resource
from db import mysql
from flask import jsonify, request, make_response
import bcrypt
import jwt


SECRET_KEY = "rio_sangat_hebat"
```

```
class Login(Resource):
    def post(self):
        data = request.json

        cursor = mysql.connection.cursor()

        # Fetching specific user
        cursor.execute("CALL get_user(%s)", (data['name'],))
        user = cursor.fetchone()
        cursor.close()

        # Verify User
        if user:
            print(user)
            if bcrypt.checkpw(data['password'].encode('utf-8'),
user[2].encode('utf-8')):
                token = jwt.encode({'id': user[0], 'role': user[3]},
SECRET_KEY, algorithm='HS256')
                response = make_response(jsonify({'message': 'Login
successful'}))
                response.set_cookie(
                    'TOKEN',
                    token,
                    httponly=True,
                    secure=True,
                    samesite='Lax',
                    max_age=24*60*60
                )
                return response
            return jsonify({'message': 'Wrong password'})

        return jsonify({'message': 'User not found'})
```

user.py

```
from flask_restful import Resource
from db import mysql
from flask import jsonify, request
import bcrypt
```

```
class User(Resource):
    def post(self):
        # DATA : email, fullname, password, is_admin
        data = request.json

        # Hash password dari user
        password = bcrypt.hashpw(data['password'].encode('utf-8'),
bcrypt.gensalt())

        # Execute query
        cursor = mysql.connection.cursor()
        cursor.execute(
            "CALL create_user(%s, %s, %s, %s)",
            (data['email'], data['fullname'], password,
data['is_admin'])
        )

        mysql.connection.commit()
        cursor.close()
        return jsonify({"message": "User created"})
```

tipe_kendaraan.py

```
from flask_restful import Resource
from db import mysql
from flask import jsonify, request

class TipeKendaraan(Resource):
    def get(self, id=None):
        cursor = mysql.connection.cursor()
        data = request.json

        cursor.execute('CALL get_or_create_tipe_kendaraan(%s)',
(data['tipe'],))
        row = cursor.fetchone()
        cursor.close()

        print(row)
```

```
        return jsonify({'message': f'Tipe Kendaraan {data['tipe']}'})
```

sewa.py

```
from flask_restful import Resource
from db import mysql
from flask import jsonify, request
from . import service

class Sewa(Resource):
    def post(self):
        # URUTAN : PLAT, ID USER, TANGGAL MULAI, TANGGAL SELESAI
        data = request.json

        print(data)

        cursor = mysql.connection.cursor()

        user = service.decode_jwt_token()

        print(user)

        # Get price for the car
        cursor.execute("CALL sewa_kendaraan(%s, %s, %s, %s)",
(data['plat_nomor'], user['id'], data['tanggal_mulai'],
data['tanggal_selesai']))
        row = cursor.fetchone()
        cursor.close()

        return jsonify({'message': 'Kendaraan Berhasilar', 'data':
row})
```

service.py

```
import jwt
from flask import request, jsonify

SECRET_KEY = 'rio_sangat_hebat'
# JWT Decode Utility
```

```
def decode_jwt_token():
    token = request.json.get('token')
    if not token:
        return None

    try:
        decoded_data = jwt.decode(token, SECRET_KEY,
algorithms=['HS256'])
        return decoded_data
    except jwt.ExpiredSignatureError:
        return jsonify({'message': 'Token expired'}), 401
    except jwt.InvalidTokenError:
        return jsonify({'message': 'Invalid token'}), 401
```

kendaraan.py

```
from flask_restful import Resource
from db import mysql
from flask import jsonify, request
from . import service

class Kendaraan(Resource):
    def post(self):
        # URUTAN : PLAT, HARGA TIPE
        data = request.json
        cursor = mysql.connection.cursor()

        # Execute query
        cursor.execute("CALL tambah_kendaraan(%s, %s, %s)",
(data['plat_nomor'], data['harga_per_hari'], data['tipe']))
        mysql.connection.commit()
        cursor.close()

        return jsonify({'message': f'Kendaraan ditambahkan
({data['plat_nomor']})'})

    def delete(self):
        # INPUT : PLAT

        user = service.decode_jwt_token()
```

```
        print(user)

        if not user['role']:
            return jsonify({ "message":"Kamu tidak memiliki akses" })

        data = request.json
        cursor = mysql.connection.cursor()

        # Execute query
        cursor.execute("CALL hapus_kendaraan(%s)",
(data['plat_nomor'],))
        mysql.connection.commit()
        cursor.close()

        return jsonify({'message': f'Kendaraan dengan plat nomor
{data['plat_nomor']} berhasil dihapus'})
```

kembali.py

```
from flask_restful import Resource
from db import mysql
from flask import jsonify, request
from . import service

class Kembali(Resource):
    def post(self):
        # URUTAN : PLAT, TANGGAL KEMBALI, RUSAK, METODE_PEMBAYARAN
        data = request.json

        cursor = mysql.connection.cursor()

        user = service.decode_jwt_token()

        print(user)

        # Executing procedure
        cursor.execute("CALL kembalikan_kendaraan(%s, %s, %s, %s)",
(data['plat_nomor'], data['tanggal_kembali'], data['kondisi'],
data['metode_pembayaran']))
        row = cursor.fetchone()
        cursor.close()
```

```
        return jsonify({'message': f'Berhasil mengembalikan kendaraan
{data['plat_nomor']}', 'data': row})
```

history.py

```
from flask_restful import Resource
from db import mysql
from flask import jsonify, request
from . import service

class History(Resource):
    def post(self, history_type):
        cursor = mysql.connection.cursor()
        if history_type == 'kendaraan':
            plat_nomor = request.json['plat_nomor']
            cursor.execute('CALL get_sewa_history_kendaraan(%s)',
(plat_nomor,))
            row = cursor.fetchall()

            cursor.execute('SELECT
get_total_revenue_by_kendaraan(%s)', (plat_nomor,))
            total_revenue = cursor.fetchone()[0]

            cursor.close()

            return jsonify({"message": f"Berhasil fetch history
kendaraan ({plat_nomor})", "history": row, "total_revenue":
total_revenue})

        elif history_type == 'user':
            data = request.json
            return jsonify({"message": "Retrieved user history"})

        else:
            return jsonify({"message": "Invalid parameter"})
```