

LAMPIRAN

Lampiran 1 Surat Kesediaan pembimbing I

SURAT KESEDIAAN MEMBIMBING TA

Yang bertanda tangan di bawah ini:

Nama : Wildani Eko Nugroho, M.Kom.
NIDN : 0617078204
NIPY : 12.013.169
Jabatan Struktural : Dosen Tetap
Jabatan Fungsional : Lektor

Dengan ini menyatakan bersedia untuk menjadi pembimbing I pada Tugas Akhir mahasiswa berikut:

Nama : Muhamad Wisnu Solekhudin
NIM : 23040073
Program Studi : D3 Teknik Komputer

Judul TA : Rancang Bangun Sistem Parkir Otomatis Kampus
Berbasis IoT dengan Integrasi RFID Kartu Tanda
Mahasiswa dan Deteksi Plat Nomor Berbasis Machine
Learning Menggunakan Flutter dan FastAPI

Demikian pernyataan ini dibuat agar dapat dilaksanakan sebagaimana mestinya.

Tegal, 23 Februari 2026

Mengetahui

Ka. Prodi DIII Teknik Komputer,


Arif Rakhman, SE, S.Pd, M.Kom
NIPY.05.016.291

Dosen Pembimbing I,


Wildani Eko Nugroho M.Kom
NIPY. 12.013.169

Lampiran 2 Surat Kesediaan pembimbing II

SURAT KESEDIAAN MEMBIMBING TA

Yang bertanda tangan di bawah ini:

Nama : Lukmanul Khakim, S.Kom., M.Tr.T., IPP
NIDN : 0604059004
NIPY : 08.017.343
Jabatan Struktural : Dosen Tetap
Jabatan Fungsional : Lektor

Dengan ini menyatakan bersedia untuk menjadi pembimbing II pada Tugas Akhir mahasiswa berikut:

Nama : Muhamad Wisnu Solekhfudin
NIM : 23040073
Progran Studi : D3 Teknik Komputer

Judul TA : Rancang Bangun Sistem Parkir Otomatis Kampus
Berbasis IoT dengan Integrasi RFID Kartu Tanda
Mahasiswa dan Deteksi Plat Nomor Berbasis Machine
Learning Menggunakan Flutter dan FastAPI

Demikian pernyataan ini dibuat agar dapat dilaksanakan sebagaimana mestinya.

Tegal, 24 Februari 2026

Mengetahui

Ka. Prodi DIII Teknik Komputer,

Dosen Pembimbing II,



Arif Rakhman, SE, S.Pd, M.Kom
NIPY.05.016.291

Lukmanul Khakim, S.Kom., M.Tr.T., IPP
NIPY. 08.017.343

Lampiran 3 Surat Observasi Universitas Harkat Negeri



Sekolah Vokasi
Program Studi D-3 Teknik Komputer

Nomor : 655.03/D-3 T.Komp.HN/VI/2026
Lampiran : -
Perihal : Permohonan Izin Observasi Tugas Akhir (TA)

Kepada Yth.

Kepala Pengelola Parkir Kampus Universitas Harkat Negeri

Di Jl. Mataram No.9, Pesurungan Lor, Kec. Margadana, Kota Tegal, Jawa Tengah 52147

Dengan Hormat,

Sehubungan dengan tugas mata kuliah Tugas Akhir (TA) yang akan diselenggarakan di semester V Program Studi Diploma III Teknik Komputer Universitas Harkat Negeri, Maka dengan ini kami mengajukan izin observasi pengambilan data di Pengelola Parkir Kampus Universitas Harkat Negeri yang Bapak / Ibu Pimpin, untuk kepentingan dalam pembuatan produk Tugas Akhir, dengan Mahasiswa sebagai berikut:

No	NIM	NAMA	NO HP
1	23040073	MUHAMAD WISNU SOLEKHFUDIN	082242638686
2	23040074	MUHAMAD RIZAL MAULANA	082328246244

Demikian surat permohonan ini kami sampaikan atas izin dan kerjasamanya kami sampaikan terima kasih.

Tegal, 18 Januari 2026
Ka. Prodi DIII Teknik Komputer
Universitas Harkat Negeri



Arif Rakhman SE, S.Pd, M.Kom
NIPY. 05.016.291

Lampiran 4 Surat Observasi Rita SuperMall Tegal



Sekolah Vokasi
Program Studi D-3 Teknik Komputer

Nomor : 656.03/D-3 T.Komp.HN/VI/2026
Lampiran : -
Perihal : Permohonan Izin Observasi Tugas Akhir (TA)

Kepada Yth.

Pimpinan Pengelola Parkir Rita Supermall Tegal

Di Jl. Kolonel Sugiono No.155, Kemandungan, Kec. Tegal Bar., Kota Tegal, Jawa Tengah
52133

Dengan Hormat,

Sehubungan dengan tugas mata kuliah Tugas Akhir (TA) yang akan diselenggarakan di semester V Program Studi Diploma III Teknik Komputer Universitas Harkat Negeri, Maka dengan ini kami mengajukan izin observasi pengambilan data di Pengelola Parkir Rita Supermall Tegal yang Bapak / Ibu Pimpin, untuk kepentingan dalam pembuatan produk Tugas Akhir, dengan Mahasiswa sebagai berikut:

No	NIM	NAMA	NO HP
1	23040073	MUHAMAD WISNU SOLEKHFUDIN	082242638686
2	23040074	MUHAMAD RIZAL MAULANA	082328246244

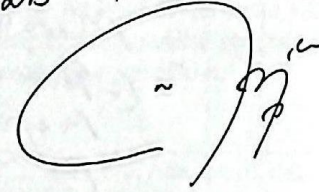
Demikian surat permohonan ini kami sampaikan atas izin dan kerjasamanya kami sampaikan terima kasih.

Tegal, 18 Januari 2026
Ka. Prodi DIII Teknik Komputer
Universitas Harkat Negeri



Arif Rakhman SE, S.Pd, M.Kom
NIPY. 05.016.291

Lampiran 5 Laporan Pembimbing I TA

No	HARI/ TANGGAL	URAIAN	TANDA TANGAN
3.	Seminar 2/3 - 2026	Revisi: - Latar belakang. - Rangkuman pengantar	
4.	Belekan 3/3 - 2026	Revisi: - Study literature. - format penulisan	
		Revisi Proposal. 3/3-26 Canjutan ke laporan Bab 1, 2, 3	
1.	Laporan - Kamirs. 12/3 - 2026	Revisi Bab 1, 2, 3 Format penulisan	

No	HARI/ TANGGAL	URAIAN	TANDA TANGAN
		12/ - 2026 13 Ju Bab 1,2,3 Langshe ke Bab 4,5,6 dan Project TA ke Pentamting 2. 	

Lampiran 6 Laporan Pembimbing II TA

Lampiran 23
Bimbingan Laporan Pembimbing I TA

IK P2M PHB d51e1

PEMBIMBING II:

BIMBINGAN LAPORAN TA

No	HARI/ TANGGAL	URAIAN	TANDA TANGAN
1.	Jumat 29/5-2026	italic bab 4, Penomoran tabel dan gambar	
2.	Rabu 10/6 2026	revisi project pada emergency gate, bab 6 dipersingkat lagi.	
3.	Rabu 24/6-2026	Penambahan batasan masalah dan perapikan penulisan.	
4.	Kamis 25/6	- Re BAB IV - Re BAB V - Re BAB VI - Project Re	

Lampiran 7 Dokumentasi Observasi



Lampiran 8 Source code

```
from fastapi import APIRouter, Depends, HTTPException
from sqlalchemy.orm import Session
from typing import List

import models
from schemas.vehicle import VehicleResponse
from core.security import get_petugas
from database import get_db
from datetime import datetime, timezone, timedelta

def to_jakarta_time(dt: datetime) -> datetime:
    if dt is None:
        return None
    if dt.tzinfo is None:
        dt = dt.replace(tzinfo=timezone.utc)
    jakarta_tz = timezone(timedelta(hours=7))
    return dt.astimezone(jakarta_tz)

router = APIRouter(prefix="/api/petugas", tags=["Petugas"],
dependencies=[Depends(get_petugas)])

@router.get("/vehicles/pending",
response_model=List[VehicleResponse])
def get_pending_vehicles(db: Session = Depends(get_db)):
    # List vehicles that need approval by officer
    return
db.query(models.Vehicle).filter(models.Vehicle.status_validasi ==
models.ValidationStatusEnum.pending).all()

@router.put("/vehicles/{vehicle_id}/verify",
response_model=VehicleResponse)
async def verify_vehicle(vehicle_id: int, status: str, db: Session
= Depends(get_db)):
    if status not in [models.ValidationStatusEnum.disetujui,
models.ValidationStatusEnum.ditolak]:
        raise HTTPException(status_code=400, detail="Invalid
status input")

    vehicle = db.query(models.Vehicle).filter(models.Vehicle.id ==
vehicle_id).first()
    if not vehicle:
        raise HTTPException(status_code=404, detail="Vehicle not
found")

    # Update state and commit
    vehicle.status_validasi = status
    db.commit()
    db.refresh(vehicle)
    try:
        from routers.iot import manager
        await manager.broadcast({"type": "update", "message":
"vehicle_verified"})
    except Exception:
        pass
```

```

        return vehicle

@router.get("/access-requests/pending")
def get_pending_access_requests(db: Session = Depends(get_db)):
    from datetime import datetime, timezone
    requests = db.query(models.AccessRequest).filter(
        models.AccessRequest.status ==
models.AccessRequestStatusEnum.pending
    ).order_by(models.AccessRequest.waktu_request.asc()).all()

    result = []
    for r in requests:
        if r.emergency_guest_id:
            guest =
db.query(models.EmergencyGuest).filter(models.EmergencyGuest.id ==
r.emergency_guest_id).first()
            result.append({
                "id": r.id,
                "user_id": None,
                "user_nama": guest.nama + " (Tamu Darurat)" if
guest else "Unknown Darurat",
                "user_nim": "Tamu Darurat",
                "rfid_uid": None,
                "vehicle_plat": guest.plat_nomor if guest else
"Unknown",
                "vehicle_jenis": "Mobil/Motor",
                "jenis_aktivitas": r.jenis_aktivitas,
                "waktu_request":
to_jakarta_time(r.waktu_request).isoformat() if r.waktu_request
else None,
                "is_flagged": False,
                "flag_reason": None,
            })
        else:
            user = db.query(models.User).filter(models.User.id ==
r.user_id).first()
            vehicle =
db.query(models.Vehicle).filter(models.Vehicle.id ==
r.vehicle_id).first()
            result.append({
                "id": r.id,
                "user_id": r.user_id,
                "user_nama": user.nama if user else "Unknown",
                "user_nim": user.nim_npp if user else "Unknown",
                "rfid_uid": user.rfid_uid if user else None,
                "vehicle_plat": vehicle.plat_nomor if vehicle else
"Unknown",
                "vehicle_jenis": vehicle.jenis_kendaraan if
vehicle else "Unknown",
                "jenis_aktivitas": r.jenis_aktivitas,
                "waktu_request":
to_jakarta_time(r.waktu_request).isoformat() if r.waktu_request
else None,
                "is_flagged": user.is_flagged == 1 if user else
False,
                "flag_reason": user.flag_reason if user else None,
            })

```

```

        })
    return result

@router.put("/access-requests/{request_id}/respond")
async def respond_to_access_request(request_id: int, action: str,
catatan: str = "", db: Session = Depends(get_db)):
    from datetime import datetime, timezone
    if action not in ["disetujui", "ditolak"]:
        raise HTTPException(status_code=400, detail="Action harus
'disetujui' atau 'ditolak'")

    req =
db.query(models.AccessRequest).filter(models.AccessRequest.id ==
request_id).first()
    if not req:
        raise HTTPException(status_code=404, detail="Request tidak
ditemukan")
    if req.status != models.AccessRequestStatusEnum.pending:
        raise HTTPException(status_code=400, detail="Request sudah
diproses")

    req.status = action
    req.waktu_respon = datetime.now(timezone.utc)
    req.catatan = catatan if catatan else None

    # If approved, create the actual ParkingLog
    if action == "disetujui":
        display_name = "Mahasiswa"
        display_plate = "MANUAL"

        if req.emergency_guest_id:
            guest =
db.query(models.EmergencyGuest).filter(models.EmergencyGuest.id ==
req.emergency_guest_id).first()
            if guest:
                guest.waktu_keluar = datetime.now(timezone.utc)
                guest.status = "sudah_keluar"
                display_name = guest.nama + " (Tamu)"
                display_plate = guest.plat_nomor

            # Log as manual for emergency
            new_log = models.ParkingLog(
                user_id=None,
                vehicle_id=None,
                emergency_guest_id=req.emergency_guest_id,
                jenis_aktivitas=req.jenis_aktivitas,

status_akses=models.AccessStatusEnum.manual_petugas
            )
            db.add(new_log)
        else:
            user = db.query(models.User).filter(models.User.id ==
req.user_id).first()
            vehicle =
db.query(models.Vehicle).filter(models.Vehicle.id ==
req.vehicle_id).first()

```

```

        if user:
            display_name = user.nama
        if vehicle:
            display_plate = vehicle.plat_nomor

        new_log = models.ParkingLog(
            user_id=req.user_id,
            vehicle_id=req.vehicle_id,
            jenis_aktivitas=req.jenis_aktivitas,

status_akses=models.AccessStatusEnum.manual_petugas
        )
        db.add(new_log)

        # Trigger physical servo via Firebase Realtime Database
        import asyncio
        from core.firebase import trigger_physical_servo
        gate_id = "GATE_MASUK_1" if req.jenis_aktivitas == "masuk"
    else "GATE_KELUAR_1"
        asyncio.create_task(trigger_physical_servo(gate_id))

        # Broadcast success log to the Live Monitor Socket for
petugas dashboard
        try:
            from routers.iot import manager
            await manager.broadcast({
                "type": "success",
                "message": f"Akses {req.jenis_aktivitas} disetujui
manual.",
                "user": display_name,
                "plate": display_plate,
                "time":
to_jakarta_time(datetime.now(timezone.utc)).isoformat()
            })
        except Exception:
            pass

        db.commit()

        try:
            from routers.iot import manager
            await manager.broadcast({"type": "update", "message":
"request_responded"})
        except Exception:
            pass

        status_msg = "disetujui dan gate dibuka" if action ==
"disetujui" else "ditolak"
        return {"status": "success", "message": f"Permintaan telah
{status_msg}"}

@router.get("/search")
def search_members(query: str, db: Session = Depends(get_db)):
    """Search users by NIM, Nama, or Vehicle Plate."""
    # Search in Users
    users = db.query(models.User).filter(

```

```

        (models.User.nim_npp.contains(query)) |
        (models.User.nama.contains(query))
    ).limit(20).all()

    # Search in Vehicles
    vehicles =
db.query(models.Vehicle).filter(models.Vehicle.plat_nomor.contains
(query)).limit(20).all()
    vehicle_owners = [v.user for v in vehicles if v.user]

    # Merge and deduplicate
    all_users = {u.id: u for u in (users +
vehicle_owners)}.values()

    result = []
    for u in all_users:
        v_list = [{"id": v.id, "plat": v.plat_nomor, "jenis":
v.jenis_kendaraan, "status": v.status_validasi} for v in
u.vehicles]
        result.append({
            "id": u.id,
            "nama": u.nama,
            "nim": u.nim_npp,
            "is_flagged": u.is_flagged == 1,
            "flag_reason": u.flag_reason,
            "vehicles": v_list
        })
    return result

@router.get("/session-stats")
def get_session_stats(db: Session = Depends(get_db), current_user:
models.User = Depends(get_petugas)):
    """Get stats of actions performed by the current officer
today."""
    from datetime import datetime, timezone, timedelta
    jakarta_tz = timezone(timedelta(hours=7))
    now_jakarta =
datetime.now(timezone.utc).astimezone(jakarta_tz)
    today_start_jakarta = now_jakarta.replace(hour=0, minute=0,
second=0, microsecond=0)
    today_start =
today_start_jakarta.astimezone(timezone.utc).replace(tzinfo=None)

    # AccessRequests handled by this officer (waktu_respon is when
it was handled)
    # Note: we need to track WHICH officer handled it.
    # For now, let's just use general log count or assume logs
handled by anyone today for simplicity
    # unless we want to add 'handled_by' to AccessRequest.
    # Let's count ParkingLogs of type 'manual_petugas' (since
these are from requests)
    logs_count = db.query(models.ParkingLog).filter(
        models.ParkingLog.status_akses ==
models.AccessStatusEnum.manual_petugas,
        models.ParkingLog.waktu >= today_start
    ).count()

```

```

        # Count STNK approvals (using general count vs this officer -
        adding 'handled_by' would be better but let's keep it lean)
        pending_stnk =
db.query(models.Vehicle).filter(models.Vehicle.status_validasi ==
models.ValidationStatusEnum.pending).count()

    return {
        "handled_count": logs_count,
        "pending_stnk": pending_stnk
    }

@router.put("/flag-user/{user_id}")
async def toggle_flag(user_id: int, is_flagged: bool, reason: str
= "", db: Session = Depends(get_db)):
    user = db.query(models.User).filter(models.User.id ==
user_id).first()
    if not user:
        raise HTTPException(status_code=404, detail="User not
found")

    user.is_flagged = 1 if is_flagged else 0
    user.flag_reason = reason if is_flagged else None
    db.commit()
    try:
        from routers.iot import manager
        await manager.broadcast({"type": "update", "message":
"user_flagged_toggled"})
    except Exception:
        pass
    return {"status": "success", "is_flagged": is_flagged}

@router.get("/activity-chart")
def get_activity_chart(db: Session = Depends(get_db)):
    from datetime import datetime, timedelta, timezone
    from sqlalchemy import func
    results = []
    for i in range(6, -1, -1):
        day = (datetime.now(timezone.utc) -
timedelta(days=i)).date()
        in_count = db.query(models.ParkingLog).filter(
            func.date(models.ParkingLog.waktu) == day,
            models.ParkingLog.jenis_aktivitas ==
models.ActivityTypeEnum.masuk
        ).count()
        out_count = db.query(models.ParkingLog).filter(
            func.date(models.ParkingLog.waktu) == day,
            models.ParkingLog.jenis_aktivitas ==
models.ActivityTypeEnum.keluar
        ).count()
        results.append({
            "day": day.strftime("%a"),
            "masuk": in_count,
            "keluar": out_count
        })
    return results

```