Nama : **Ayesha Margalla Putri**

NIM : **1202213214**

Kelas : **SI-46-06**

Tugas 2 - Perancangan API dengan Arsitektur SOA

Perusahaan seringkali menghadapi tantangan dalam mengelola data karyawan, absensi, dan penggajian secara terpisah. Untuk mempermudah manajemen data dan memisahkan setiap layanan, perusahaan dapat menggunakan pendekatan Service-Oriented Architecture (SOA). Dalam proyek ini, saya telah merancang dan mengembangkan sebuah API berbasis SOA menggunakan FastAPI untuk mengelola data karyawan, absensi, dan penggajian. Setiap layanan akan dikelola secara terpisah dan saling berkomunikasi melalui REST API. FastAPI digunakan untuk mempermudah pembuatan API ini dan menyediakan dokumentasi otomatis melalui Swagger UI. Tiga layanan API yang dimaksud termasuk berikut:

1. Service Karyawan: Untuk mengelola data karyawan (id, nama, posisi)
2. Service Absensi: Untuk mencatat absensi karyawan (waktu masuk dan waktu keluar)
3. Service Penggajian: Untuk mengelola penggajian karyawan (gaji pokok)

Dokumentasi API disediakan menggunakan Swagger yang dapat diakses melalui link http://127.0.0.1:8000/docs.

Code Program:

```python
from fastapi import FastAPI
from pydantic import BaseModel
from typing import List

app = FastAPI()

# Data models untuk Karyawan, Absensi, and Gaji
class Karyawan(BaseModel):
    id: int
    name: str
    posisi: str

class Absensi(BaseModel):
    id_karyawan: int
    waktu_masuk: str
    waktu_keluar: str

class Penggajian(BaseModel):
    id_karyawan: int
    gaji: float
```

```python
# Dummy data untuk API (tidak terhubung dengan database)
karyawan_db = [
    {"id": 1, "name": "John Doe", "posisi": "Manager"},
    {"id": 2, "name": "Jane Smith", "posisi": "Staff"}
]

absensi_db = [
    {"id_karyawan": 1, "waktu_masuk": "08:00", "waktu_keluar": "17:00"},
    {"id_karyawan": 2, "waktu_masuk": "09:00", "waktu_keluar": "18:00"}
]

gaji_db = [
    {"id_karyawan": 1, "gaji": 5000000},
    {"id_karyawan": 2, "gaji": 4000000}
]
```

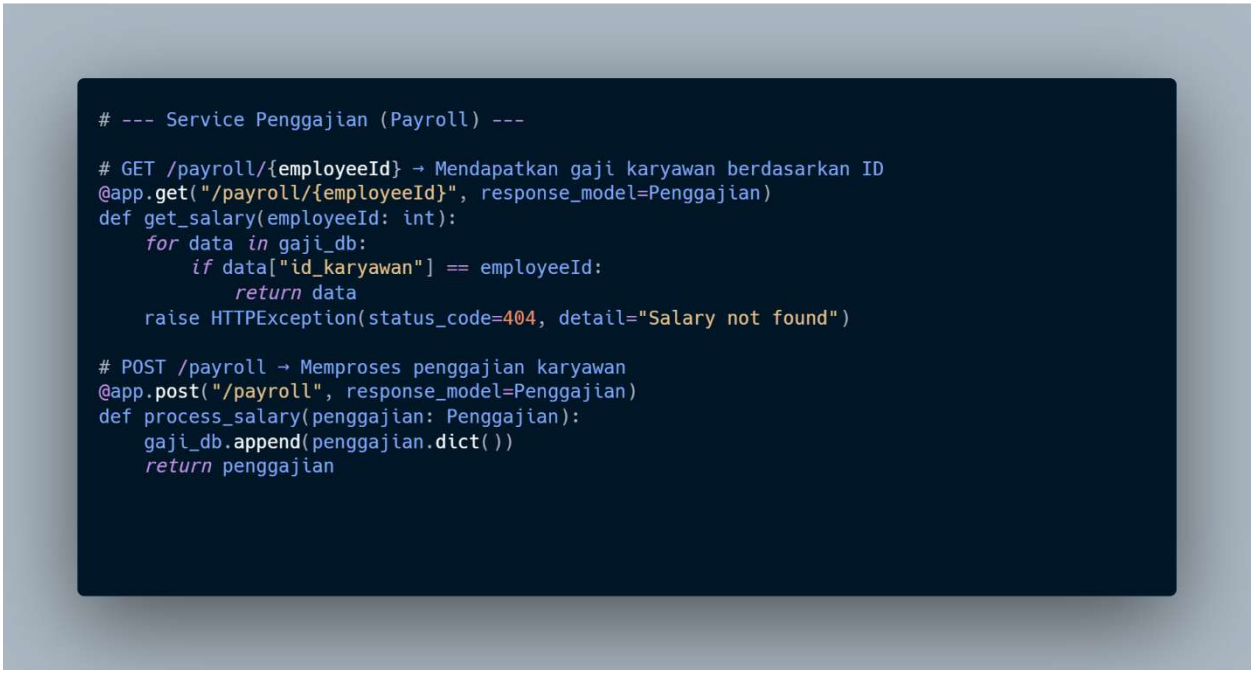
```python
# --- Service Karyawan (Employee) ---

# GET /employees → Mendapatkan daftar karyawan
@app.get("/employees", response_model=List[Karyawan])
def get_employees():
    return karyawan_db

# POST /employees → Menambahkan karyawan baru
@app.post("/employees", response_model=Karyawan)
def add_employee(karyawan: Karyawan):
    karyawan_db.append(karyawan.dict())
    return karyawan

# GET /employees/{id} → Mendapatkan detail karyawan berdasarkan ID
@app.get("/employees/{id}", response_model=Karyawan)
def get_employee(id: int):
    for emp in karyawan_db:
        if emp["id"] == id:
            return emp
    raise HTTPException(status_code=404, detail="Employee not found")
```

```python
# PUT /employees/{id} → Memperbarui data karyawan
@app.put("/employees/{id}", response_model=Karyawan)
def update_employee(id: int, updated_karyawan: Karyawan):
    for emp in karyawan_db:
        if emp["id"] == id:
            emp["name"] = updated_karyawan.name
            emp["posisi"] = updated_karyawan.posisi
            return emp
    raise HTTPException(status_code=404, detail="Employee not found")

# DELETE /employees/{id} → Menghapus karyawan
@app.delete("/employees/{id}")
def delete_employee(id: int):
    for emp in karyawan_db:
        if emp["id"] == id:
            karyawan_db.remove(emp)
            return {"message": "Employee deleted successfully"}
    raise HTTPException(status_code=404, detail="Employee not found")
```

```python
# POST /attendance/check-in → Karyawan melakukan check-in
@app.post("/attendance/check-in")
def check_in(id_karyawan: int, waktu_masuk: str):
    # Check if employee already checked-in
    for data in absensi_db:
        if data["id_karyawan"] == id_karyawan and data["waktu_keluar"] == "":
            raise HTTPException(status_code=400, detail="Employee already checked-in")

    # Add a new check-in entry
    absensi_db.append({"id_karyawan": id_karyawan, "waktu_masuk": waktu_masuk, "waktu_keluar":
""})
    return {"message": f"Karyawan {id_karyawan} checked-in at {waktu_masuk}"}
```

```python
# POST /attendance/check-out → Karyawan melakukan check-out
@app.post("/attendance/check-out")
def check_out(id_karyawan: int, waktu_keluar: str):
    for data in absensi_db:
        # Check if employee has checked-in and hasn't checked-out yet
        if data["id_karyawan"] == id_karyawan and data["waktu_keluar"] == "":
            data["waktu_keluar"] = waktu_keluar
            return {"message": f"Karyawan {id_karyawan} checked-out at {waktu_keluar}"}
    # If no matching check-in is found
    raise HTTPException(status_code=404, detail="Attendance record not found")
```

```python
# GET /attendance/{employeeId} → Mendapatkan riwayat absensi karyawan
@app.get("/attendance/{employeeId}", response_model=List[Absensi])
def get_attendance(employeeId: int):
    result = [data for data in absensi_db if data["id_karyawan"] == employeeId]
    if result:
        return result
    raise HTTPException(status_code=404, detail="Attendance data not found")
```



```python
# --- Service Penggajian (Payroll) ---

# GET /payroll/{employeeId} → Mendapatkan gaji karyawan berdasarkan ID
@app.get("/payroll/{employeeId}", response_model=Penggajian)
def get_salary(employeeId: int):
    for data in gaji_db:
        if data["id_karyawan"] == employeeId:
            return data
    raise HTTPException(status_code=404, detail="Salary not found")

# POST /payroll → Memproses penggajian karyawan
@app.post("/payroll", response_model=Penggajian)
def process_salary(penggajian: Penggajian):
    gaji_db.append(penggajian.dict())
    return penggajian
```

Penjelasan *Code*:

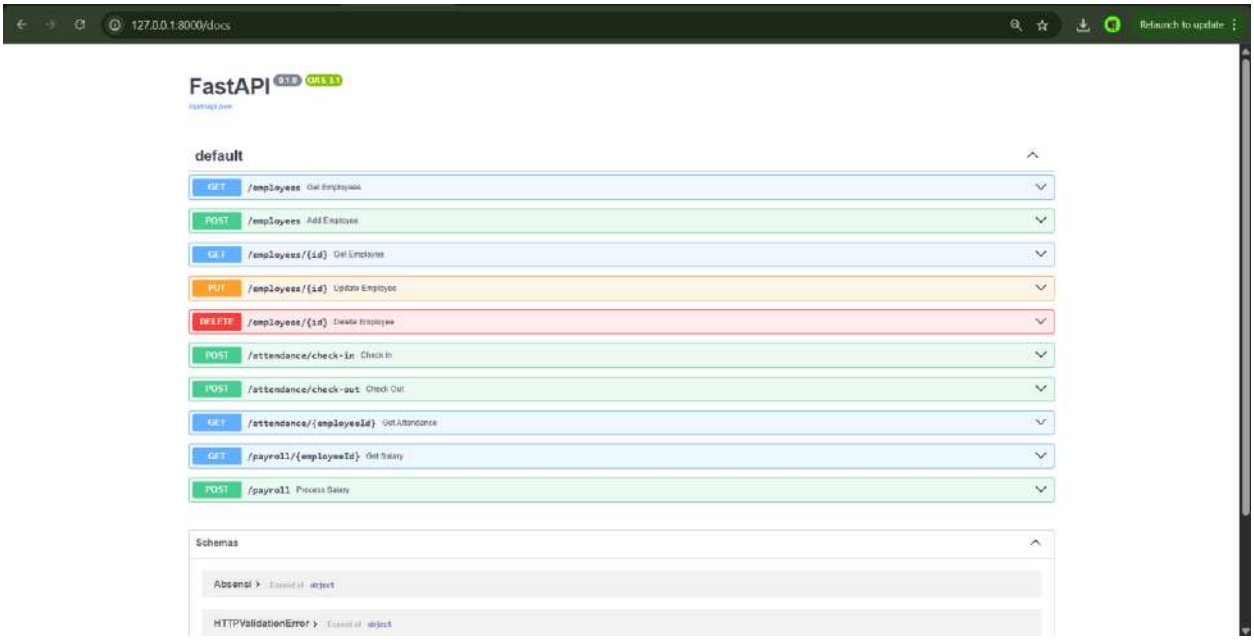
1. **FastAPI** digunakan untuk membuat server API dan mendefinisikan route dengan metode HTTP GET dan POST untuk setiap endpoint: /employees, /attendance, dan /payroll.
    - **GET** pada /employees untuk mengambil daftar karyawan atau data karyawan berdasarkan ID yang diberikan
    - **POST** pada /employees untuk menambahkan karyawan baru
    - **POST** pada /attendance/check-in dan /attendance/check-out untuk mencatat waktu kedatangan dan kepulangan karyawan
    - **GET** pada /payroll untuk melihat gaji karyawan berdasarkan ID
2. **Pydantic** digunakan untuk membuat model data yang dapat memvalidasi dan mendokumentasikan format data input dan output.
    - Model Pydantic yang digunakan adalah Karyawan, Absensi, dan Penggajian
    - Model Karyawan berisi data id, name, dan posisi
    - Model Absensi berisi data id_karyawan, waktu_masuk, dan waktu_keluar
    - Model Penggajian berisi data id_karyawan dan gaji.
3. Setiap **route** dapat menerima **parameter query**. Misalnya, jika Anda menambahkan query id_karyawan=1, API akan memfilter data sesuai dengan parameter yang diberikan.

- **GET /employees**: Bisa menerima parameter query id untuk mendapatkan data karyawan berdasarkan ID
- **GET /attendance/{employeeId}**: Bisa menerima parameter query employeeId untuk mengambil data absensi karyawan berdasarkan ID karyawan yang diberikan
- **POST /attendance/check-in** dan **POST /attendance/check-out**: Menerima parameter query id_karyawan dan waktu_masuk atau waktu_keluar untuk mencatat waktu kedatangan atau kepulangan karyawan
- **GET /payroll/{employeeId}**: Bisa menerima parameter query employeeId untuk mendapatkan data gaji karyawan berdasarkan ID karyawan yang diberikan.

Setelah menjalankan server FastAPI dengan perintah:

uvicorn app:app –reload

Swagger UI dapat diakses di http://127.0.0.1:8000/docs. Di Swagger UI, kita dapat melihat dokumentasi API secara otomatis, termasuk daftar endpoint seperti /employees, /attendance, dan /payroll.

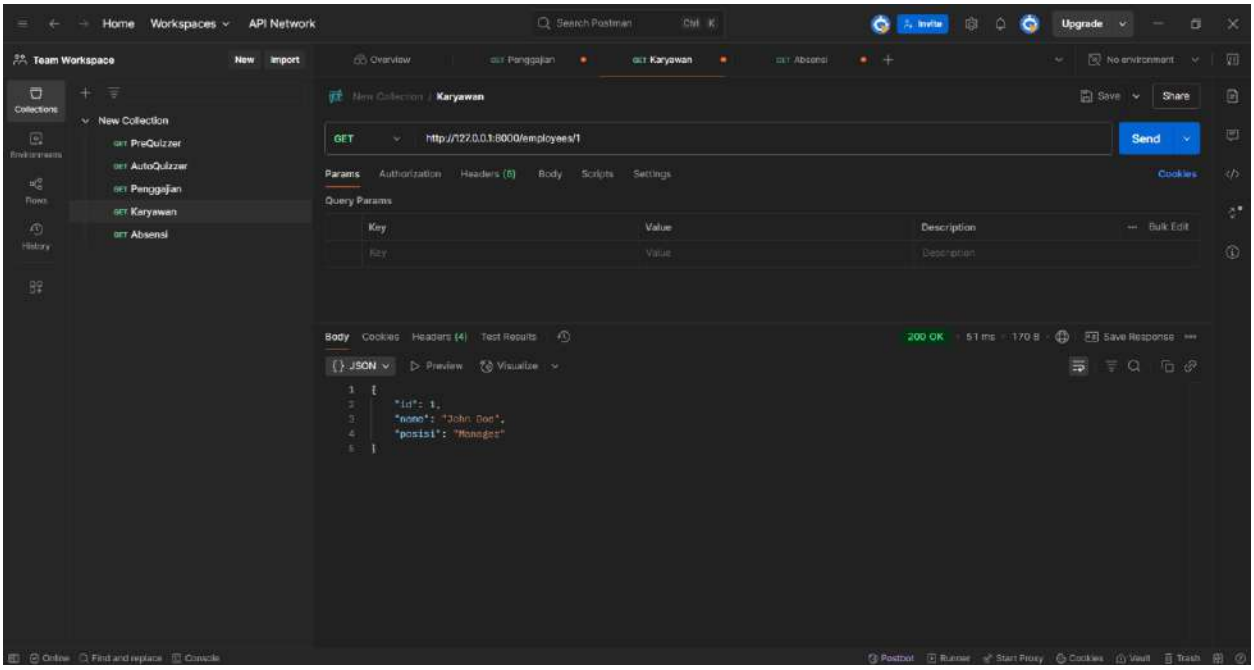


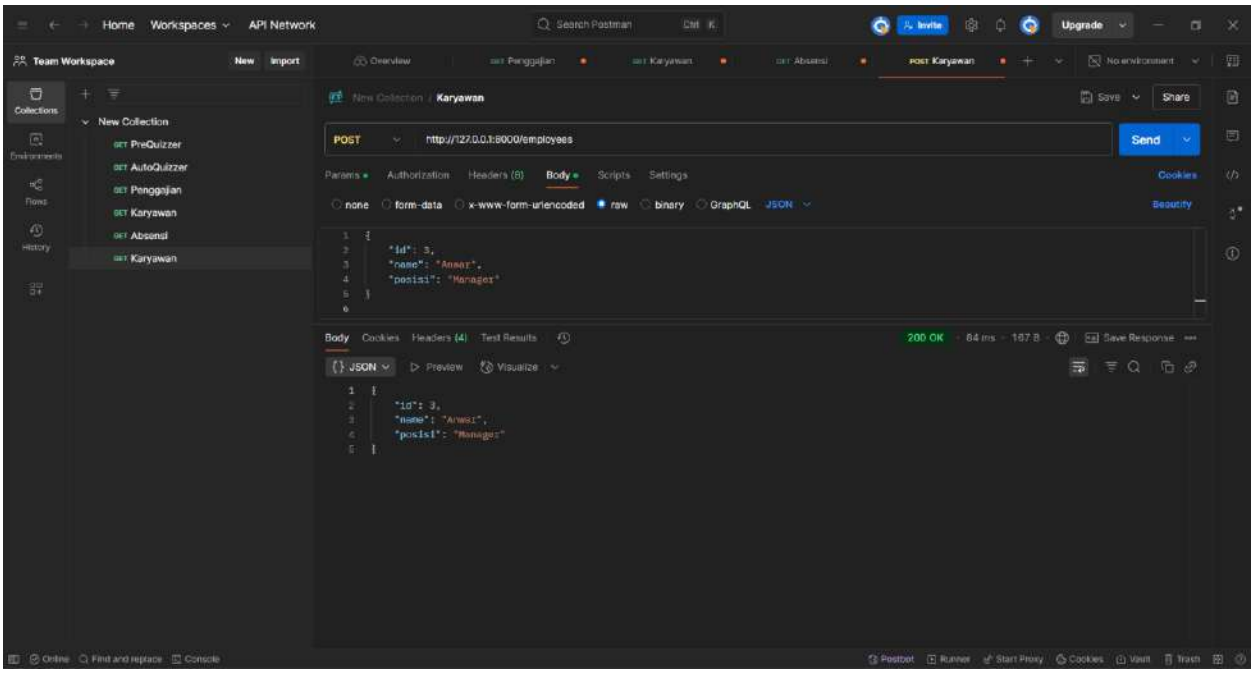
Contoh tampilan Swagger UI:

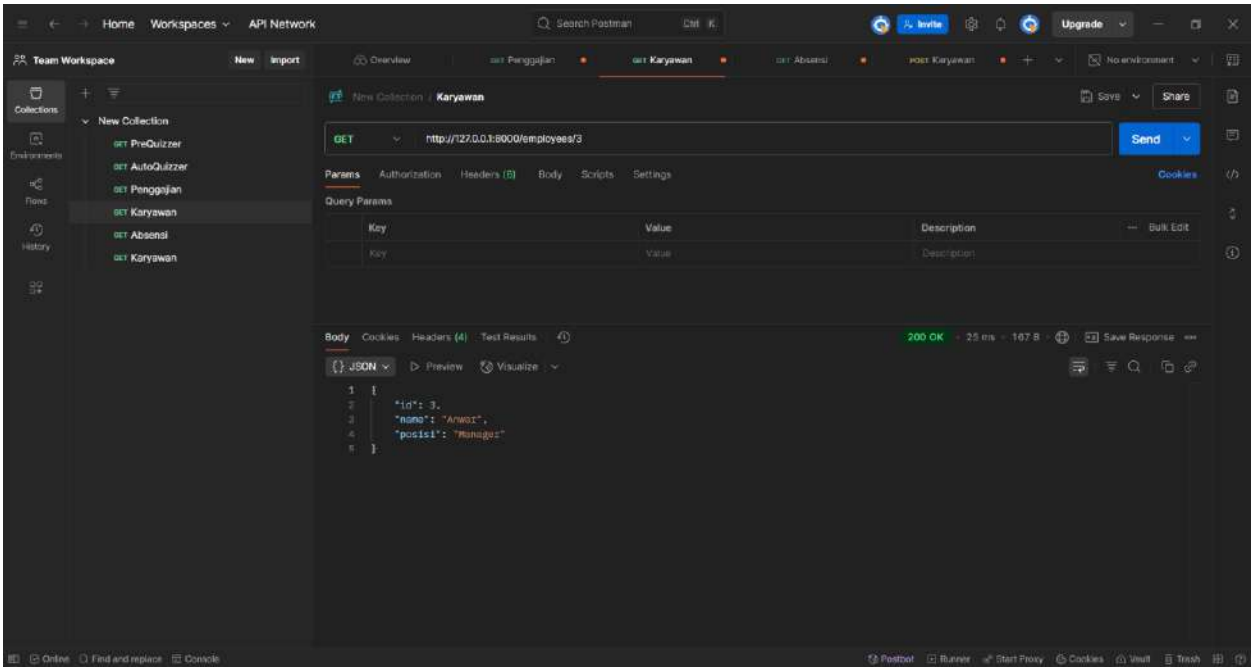
1. GET /employees: Endpoint ini akan menampilkan semua data karyawan yang terdaftar

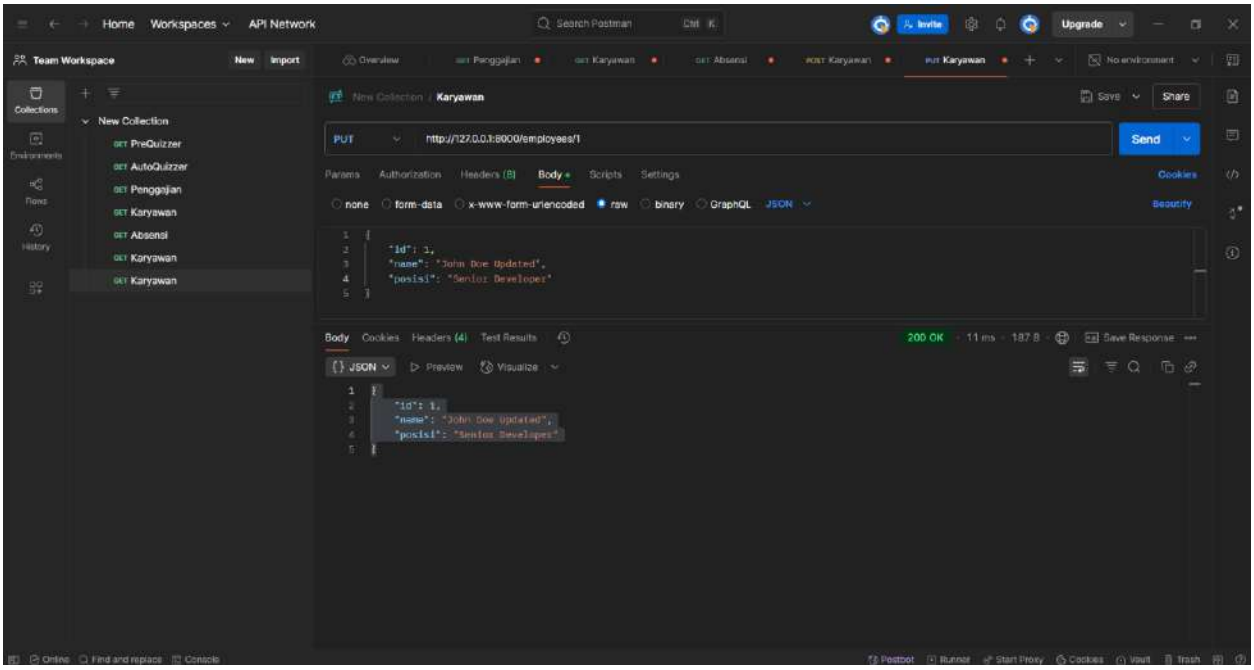
2. POST /employees: Endpoint ini digunakan untuk menambah data karyawan baru ke dalam database
3. GET /employees/{id}: Endpoint ini akan mengembalikan data karyawan berdasarkan ID yang diberikan
4. PUT /employees/{id}: Endpoint ini digunakan untuk memperbarui data karyawan yang sudah ada, berdasarkan ID
5. DELETE /employees/{id}: Endpoint ini digunakan untuk menghapus data karyawan berdasarkan ID
6. POST /attendance/check-in: Endpoint ini digunakan untuk mencatat waktu check-in karyawan
7. POST /attendance/check-out: Endpoint ini digunakan untuk mencatat waktu check-out karyawan
8. GET /attendance/{employeeId}: Endpoint ini akan mengembalikan riwayat absensi karyawan berdasarkan employeeId
9. GET /payroll/{employeeId}: Endpoint ini digunakan untuk mendapatkan data gaji karyawan berdasarkan employeeId
10. POST /payroll: Endpoint ini digunakan untuk memproses penggajian karyawan baru

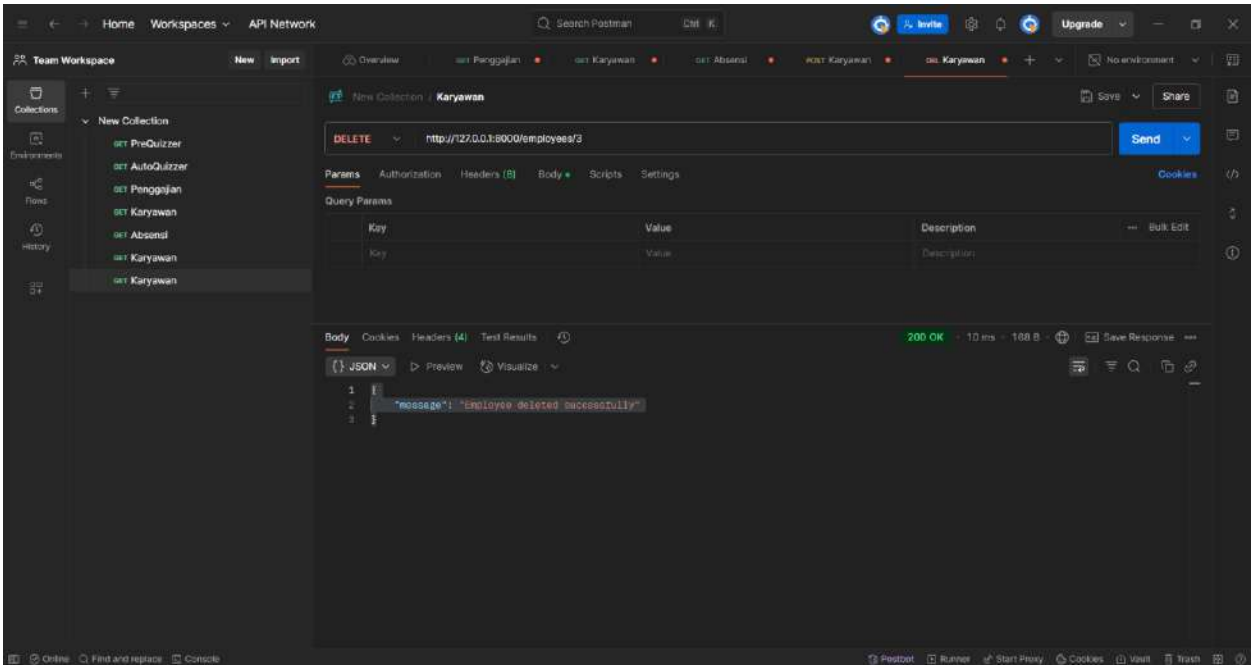
Hasil Uji Coba di Postman:

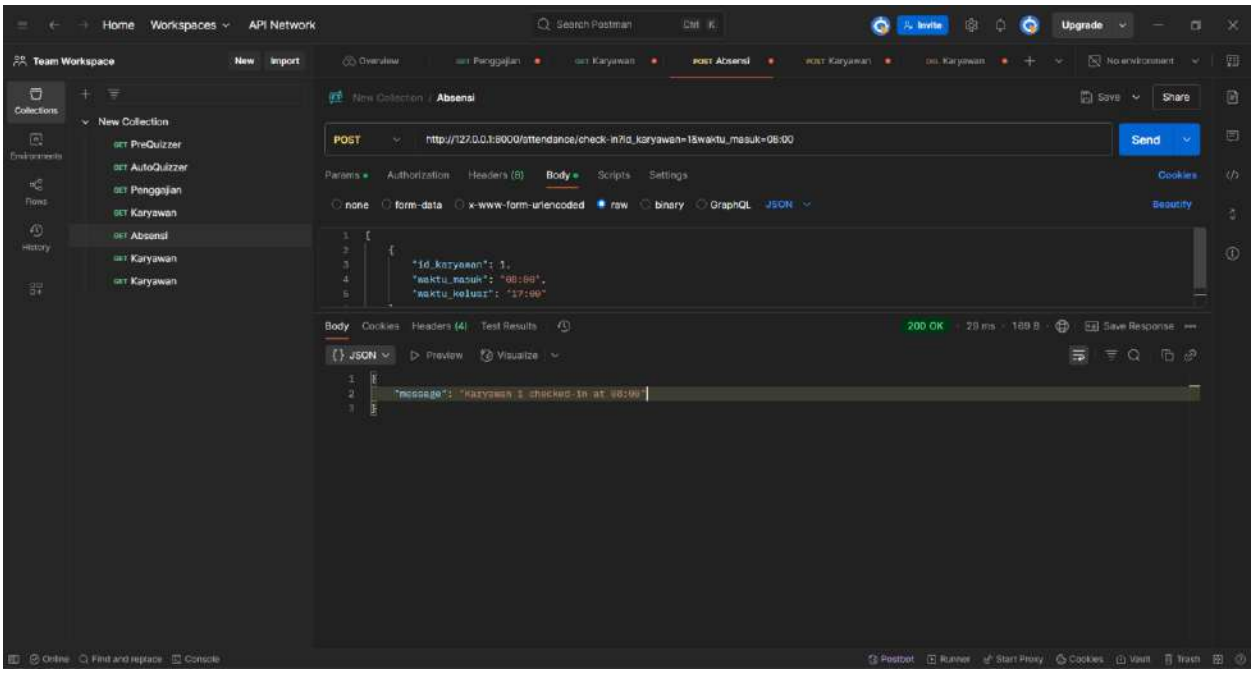


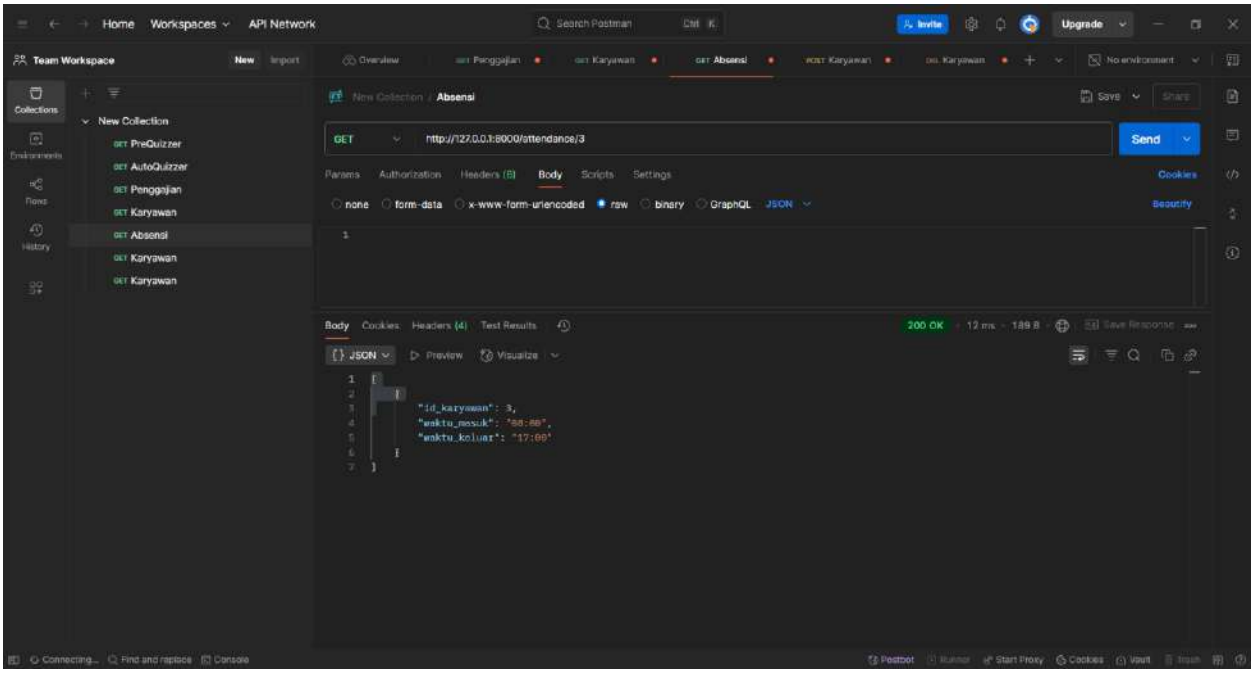

Home
Workspaces
API Network
Search Postman
Invite
Upgrade
Team Workspace
New
Import
Overview
GET Penggajian
GET Karyawan
GET Absensi
POST Karyawan
No environment
Collections
Environments
Flows
History
New Collection
GET PreQuizzer
GET AutoQuizzer
GET Penggajian
GET Karyawan
GET Absensi
GET Karyawan
New Collection / Karyawan
Save
Share
POST
http://127.0.0.1:8000/employees
Send
Params
Authorization
Headers (8)
Body
Scripts
Settings
Cookies
none
form-data
x-www-form-urlencoded
raw
binary
GraphQL
JSON
Beautify
{
"id": 3,
"name": "Anwar",
"posisi": "Manager"
}
Body
Cookies
Headers (4)
Test Results
200 OK
84 ms
167 B
Save Response
JSON
Preview
Visualize
[
{
"id": 3,
"name": "Anwar",
"posisi": "Manager"
}
]
Online
Find and replace
Console
Postbot
Runner
Start Proxy
Cookies
Vault
Trash

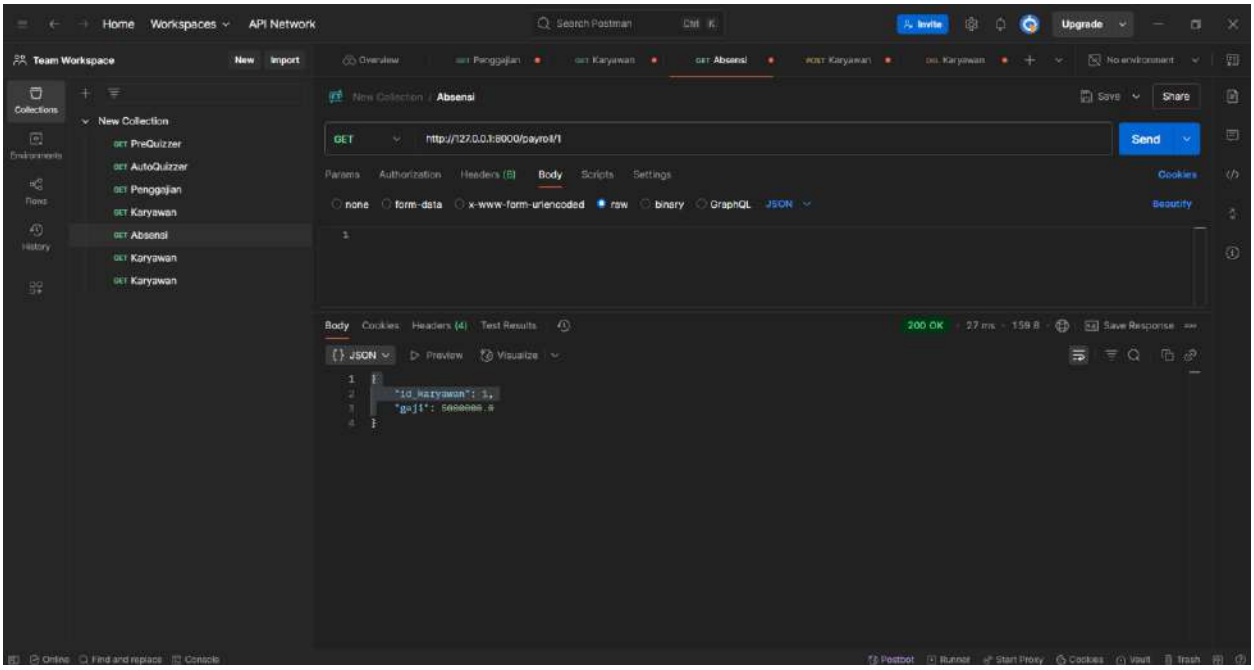

Home
Workspaces
API Network
Search Postman
Invite
Upgrade
Team Workspace
New
Import
Overview
GET Penggajian
GET Karyawan
GET Absensi
POST Karyawan
No environment
Collections
Environments
Flows
History
New Collection
GET PreQuizzer
GET AutoQuizzer
GET Penggajian
GET Karyawan
GET Absensi
GET Karyawan
New Collection / Karyawan
Save
Share
GET
http://127.0.0.1:8000/employees/3
Send
Params
Authorization
Headers (6)
Body
Scripts
Settings
Cookies
Query Params
Key
Value
Description
Bulk Edit
Body
Cookies
Headers (4)
Test Results
200 OK
25 ms
167 B
Save Response
JSON
Preview
Visualize
[
{
"id": 3,
"name": "Anwar",
"posisi": "Manager"
}
]
Online
Find and replace
Console
Postbot
Runner
Start Proxy
Cookies
Vault
Trash

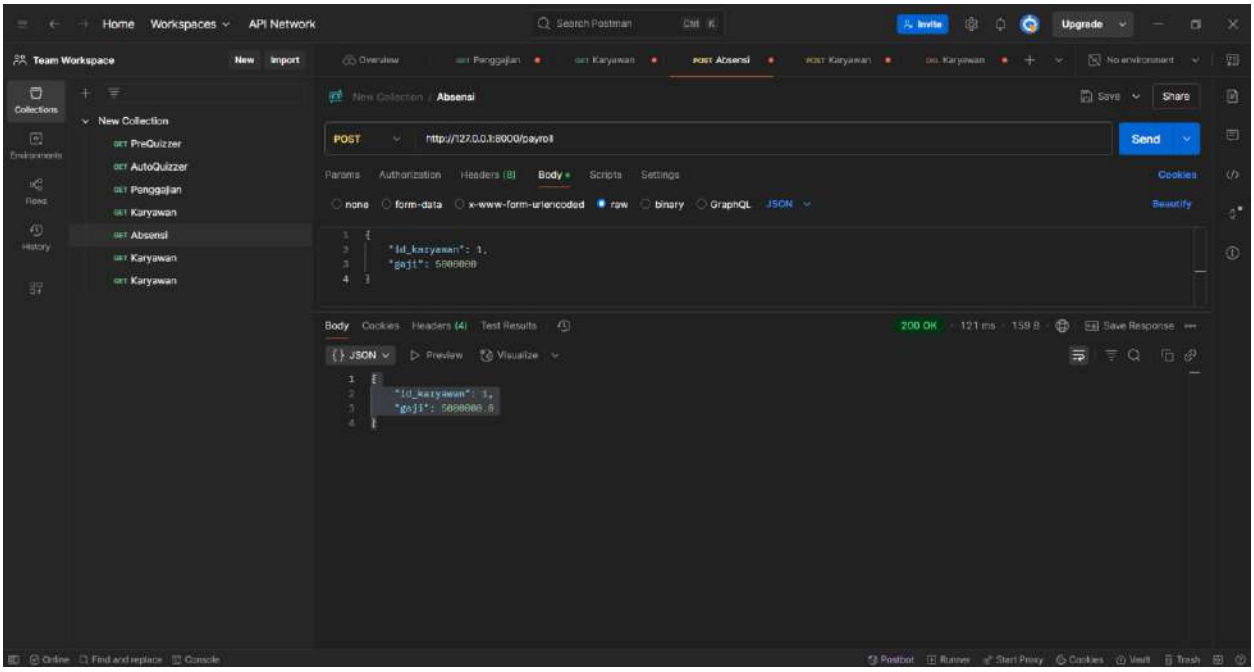

Home
Workspaces
API Network
Search Postman
Ctrl K
Invite
Upgrade
Team Workspace
New
Import
Collections
Environments
Flows
History
New Collection
GET PreQuizzer
GET AutoQuizzer
GET Penggajian
GET Karyawan
GET Absensi
GET Karyawan
GET Karyawan
Overview
GET Penggajian
GET Karyawan
GET Absensi
POST Karyawan
PUT Karyawan
No environment
New Collection / Karyawan
Save
Share
PUT
http://127.0.0.1:8000/employees/1
Send
Params
Authorization
Headers (8)
Body
Scripts
Settings
Cookies
none
form-data
x-www-form-urlencoded
raw
binary
GraphQL
JSON
Beautify
{
"id": 1,
"name": "John Doe Updated",
"posisi": "Senior Developer"
}
Body
Cookies
Headers (4)
Test Results
200 OK
11 ms
187 B
Save Response
{} JSON
Preview
Visualize
{
"id": 1,
"name": "John Doe Updated",
"posisi": "Senior Developer"
}
Online
Find and replace
Console
Postbot
Runner
Start Proxy
Cookies
Vault
Trash


Home
Workspaces
API Network
Search Postman
Ctrl K
Invite
Upgrade
Team Workspace
New
Import
Collections
Environments
Flows
History
New Collection
GET PreQuizzer
GET AutoQuizzer
GET Penggajian
GET Karyawan
GET Absensi
GET Karyawan
GET Karyawan
Overview
GET Penggajian
GET Karyawan
GET Absensi
POST Karyawan
DEL Karyawan
No environment
New Collection / Karyawan
Save
Share
DELETE
http://127.0.0.1:8000/employees/3
Send
Params
Authorization
Headers (8)
Body
Scripts
Settings
Cookies
Query Params
Key
Value
Description
Bulk Edit
Body
Cookies
Headers (4)
Test Results
200 OK
10 ms
188 B
Save Response
{} JSON
Preview
Visualize
{
"message": "Employee deleted successfully"
}
Online
Find and replace
Console
Postbot
Runner
Start Proxy
Cookies
Vault
Trash


Home
Workspaces
API Network
Search Postman
Upgrade
Team Workspace
New
Import
Collections
Environments
Flows
History
New Collection
GET PreQuizzer
GET AutoQuizzer
GET Penggajian
GET Karyawan
GET Absensi
GET Karyawan
GET Karyawan
Absensi
Save
Share
POST
http://127.0.0.1:8000/attendance/check-in?id_karyawan=1&waktu_masuk=08:00
Send
Params
Authorization
Headers (8)
Body
Scripts
Settings
Cookies
none
form-data
x-www-form-urlencoded
raw
binary
GraphQL
JSON
Beautify
{
"id_karyawan": 1,
"waktu_masuk": "08:00",
"waktu_keluar": "17:00"
}
Body
Cookies
Headers (4)
Test Results
200 OK
29 ms
189 B
Save Response
JSON
Preview
Visualize
{
"message": "Karyawan 1 checked-in at 08:00"
}
Online
Find and replace
Console
Postbot
Runner
Start Proxy
Cookies
Vault
Trash


Home
Workspaces
API Network
Search Postman
Upgrade
Team Workspace
New
Import
Collections
Environments
Flows
History
New Collection
GET PreQuizzer
GET AutoQuizzer
GET Penggajian
GET Karyawan
GET Absensi
GET Karyawan
GET Karyawan
Absensi
Save
Share
GET
http://127.0.0.1:8000/attendance/3
Send
Params
Authorization
Headers (8)
Body
Scripts
Settings
Cookies
none
form-data
x-www-form-urlencoded
raw
binary
GraphQL
JSON
Beautify
Body
Cookies
Headers (4)
Test Results
200 OK
12 ms
189 B
Save Response
JSON
Preview
Visualize
[
{
"id_karyawan": 3,
"waktu_masuk": "08:00",
"waktu_keluar": "17:00"
}
]
Connecting...
Find and replace
Console
Postbot
Runner
Start Proxy
Cookies
Vault
Trash


Home
Workspaces
API Network
Search Postman
Invite
Upgrade
Team Workspace
New
Import
Collections
Environments
Flows
History
New Collection
GET PreQuizzer
GET AutoQuizzer
GET Penggajian
GET Karyawan
GET Absensi
GET Karyawan
GET Karyawan
Overview
GET Penggajian
GET Karyawan
GET Absensi
POST Karyawan
No environment
New Collection / Absensi
Save
Share
GET
http://127.0.0.1:8000/payroll/1
Send
Params
Authorization
Headers (8)
Body
Scripts
Settings
Cookies
none
form-data
x-www-form-urlencoded
raw
binary
GraphQL
JSON
Beautify
Body
Cookies
Headers (4)
Test Results
200 OK
27 ms
159 B
Save Response
JSON
Preview
Visualize
{
"id_karyawan": 1,
"gaji": 5000000.0
}
Online
Find and replace
Console
Postbot
Runner
Start Proxy
Cookies
Vault
Trash


Home
Workspaces
API Network
Search Postman
Invite
Upgrade
Team Workspace
New
Import
Collections
Environments
Flows
History
New Collection
GET PreQuizzer
GET AutoQuizzer
GET Penggajian
GET Karyawan
GET Absensi
GET Karyawan
GET Karyawan
Overview
GET Penggajian
GET Karyawan
POST Absensi
POST Karyawan
No environment
New Collection / Absensi
Save
Share
POST
http://127.0.0.1:8000/payroll
Send
Params
Authorization
Headers (8)
Body
Scripts
Settings
Cookies
none
form-data
x-www-form-urlencoded
raw
binary
GraphQL
JSON
Beautify
{
"id_karyawan": 1,
"gaji": 5000000
}
Body
Cookies
Headers (4)
Test Results
200 OK
121 ms
159 B
Save Response
JSON
Preview
Visualize
{
"id_karyawan": 1,
"gaji": 5000000.0
}
Online
Find and replace
Console
Postbot
Runner
Start Proxy
Cookies
Vault
Trash