🔌 Documentación de API REST

🔐 Autenticación

La API utiliza sesiones de Flask para autenticación. Todas las rutas requieren autenticación excepto `/auth/login`.

Login

Endpoint: `POST /auth/login`

Body:

```
{
    "username": "admin",
    "password": "admin123",
    "remember": false
}
```

Response:

```
{
    "success": true,
    "message": "Login exitoso",
    "user": {
        "id": 1,
        "username": "admin",
        "full_name": "Administrador Principal"
    }
}
```

Logout

Endpoint: `GET /auth/logout`

Response: Redirect a `/auth/login`

👥 Empleados

Listar Empleados

Endpoint: `GET /employees/api/employees`

Query Parameters:
- `active_only` (boolean): Filtrar solo activos (default: true)
- `search` (string): Búsqueda por nombre/código/email

Response:

```
{
    "success": true,
    "employees": [
        {
            "id": 1,
            "employee_code": "EMP001",
            "full_name": "Juan Pérez",
            "email": "juan@empresa.com",
            "phone": "555-0001",
            "department": "Tecnología",
            "position": "Desarrollador",
            "photo_path": "/path/to/photo.jpg",
            "is_active": true,
            "created_at": "2024-01-15T10:30:00"
        }
    ]
}
```


Obtener Empleado

Endpoint: `GET /employees/api/employees/{id}`

Response:


```
{
    "success": true,
    "employee": {
        "id": 1,
        "employee_code": "EMP001",
        "full_name": "Juan Pérez",
        ...
    }
}
```


Crear Empleado

Endpoint: `POST /employees/api/employees`

Content-Type: `multipart/form-data`

Form Data:
- `employee_code` (string, required)
- `full_name` (string, required)
- `email` (string)
- `phone` (string)
- `department` (string)
- `position` (string)
- `photo` (string, base64): Foto facial en formato base64

Response:

```
{
    "success": true,
    "message": "Empleado creado exitosamente",
    "employee": {
        "id": 2,
        "employee_code": "EMP002",
        ...
    }
}
```

Actualizar Empleado

Endpoint: `PUT /employees/api/employees/{id}`

Body:

```
{
    "full_name": "Juan Pérez García",
    "email": "juan.perez@empresa.com",
    "phone": "555-0001",
    "department": "Ingeniería",
    "position": "Senior Developer",
    "is_active": true
}
```

Response:

```
{
    "success": true,
    "message": "Empleado actualizado exitosamente",
    "employee": {...}
}
```

Eliminar Empleado

Endpoint: `DELETE /employees/api/employees/{id}`

Response:

```
{
    "success": true,
    "message": "Empleado eliminado exitosamente"
}
```

Nota: Esta es una eliminación “soft” (is_active=false).

📅 Asistencia

Asistencia de Hoy

Endpoint: `GET /attendance/api/attendance/today`

Response:

```
{
    "success": true,
    "date": "2024-12-12",
    "attendances": [
        {
            "id": 1,
            "employee_id": 1,
            "employee_name": "Juan Pérez",
            "employee_code": "EMP001",
            "date": "2024-12-12",
            "check_in_time": "08:55:30",
            "check_out_time": "17:45:20",
            "status": "present",
            "is_late": false,
            "work_hours": 8.83
        }
    ]
}
```


Registros de Asistencia

Endpoint: `GET /attendance/api/attendance/records`

Query Parameters:
- `start_date` (string, YYYY-MM-DD)
- `end_date` (string, YYYY-MM-DD)
- `employee_id` (integer)
- `status` (string): present, late, absent

Response:


```
{
    "success": true,
    "attendances": [...]
}
```


Registrar Entrada (Check-in)

Endpoint: `POST /attendance/api/attendance/checkin`

Body:


```
{
    "photo": "..."
}
```


Response:

```
{
    "success": true,
    "message": "Entrada registrada: Juan Pérez",
    "attendance": {
        "id": 1,
        "employee_id": 1,
        "check_in_time": "08:55:30",
        ...
    },
    "employee": {...},
    "is_late": false
}
```

Error Response:

```
{
    "success": false,
    "message": "No se detectó ningún rostro en la imagen"
}
```

Registrar Salida (Check-out)

Endpoint: `POST /attendance/api/attendance/checkout`

Body:

```
{
    "photo": "..."
}
```

Response:

```
{
    "success": true,
    "message": "Salida registrada: Juan Pérez",
    "attendance": {...},
    "employee": {...},
    "work_hours": 8.83
}
```

📊 Reportes

Reporte Diario

Endpoint: `GET /reports/api/daily`

Query Parameters:
- `date` (string, YYYY-MM-DD): Fecha del reporte (default: hoy)

Response:

```
{
    "success": true,
    "date": "2024-12-12",
    "total_employees": 50,
    "present": 45,
    "absent": 5,
    "late": 3,
    "data": [
        {
            "employee_code": "EMP001",
            "employee_name": "Juan Pérez",
            "department": "Tecnología",
            "check_in": "08:55",
            "check_out": "17:45",
            "work_hours": 8.83,
            "status": "present",
            "is_late": false
        }
    ]
}
```


Reporte por Empleado

Endpoint: `GET /reports/api/employee/{employee_id}`

Query Parameters:
- `start_date` (string, YYYY-MM-DD)
- `end_date` (string, YYYY-MM-DD)

Response:


```
{
    "success": true,
    "employee": {...},
    "period": {
        "start_date": "2024-11-01",
        "end_date": "2024-11-30"
    },
    "statistics": {
        "total_days": 30,
        "present_days": 28,
        "absent_days": 2,
        "late_days": 3,
        "attendance_rate": 93.33,
        "punctuality_rate": 89.29,
        "total_work_hours": 224.5,
        "avg_work_hours": 8.02
    },
    "attendances": [...]
}
```


Reporte Mensual

Endpoint: `GET /reports/api/monthly`

Query Parameters:
- `year` (integer)
- `month` (integer, 1-12)

Response:


```
{
    "success": true,
    "period": {
        "year": 2024,
        "month": 12,
        "start_date": "2024-12-01",
        "end_date": "2024-12-31"
    },
    "statistics": {
        "total_employees": 50,
        "working_days": 22,
        "total_attendances": 1045,
        "present_count": 1045,
        "late_count": 67,
        "avg_attendance_rate": 95.0,
        "avg_punctuality_rate": 93.6
    },
    "daily_stats": [
        {
            "date": "2024-12-01",
            "present": 48,
            "absent": 2,
            "late": 3
        }
    ]
}
```


Exportar a Excel

Endpoint: `POST /reports/api/export/excel`

Body:

```
{
    "report_type": "daily",
    "params": {
        "date": "2024-12-12"
    }
}
```

Response: Archivo Excel descargable

Exportar a PDF

Endpoint: `POST /reports/api/export/pdf`

Body:

```
{
    "report_type": "employee",
    "params": {
        "employee_id": 1,
        "start_date": "2024-11-01",
        "end_date": "2024-11-30"
    }
}
```

Response: Archivo PDF descargable

🔄 Sincronización

Estado de Sincronización

Endpoint: `GET /sync/api/status`

Response:


```
{
    "success": true,
    "sync_enabled": true,
    "auto_sync": true,
    "last_sync": {
        "id": 1,
        "sync_type": "full",
        "status": "success",
        "started_at": "2024-12-12T10:30:00",
        "completed_at": "2024-12-12T10:31:45",
        "records_sent": 25,
        "records_received": 10
    },
    "pending": {
        "employees": 2,
        "attendances": 15,
        "total": 17
    }
}
```


Sincronización Manual

Endpoint: `POST /sync/api/manual`

Body:


```
{
    "sync_type": "full"
}
```


Opciones de sync_type:
- `employees`: Solo empleados
- `attendances`: Solo asistencias
- `full`: Sincronización completa

Response:

```
{
    "success": true,
    "message": "Sincronización completada",
    "sync_log": {...},
    "results": {
        "employees": {
            "sent": 2,
            "received": 0,
            "failed": 0
        },
        "attendances": {
            "sent": 15,
            "received": 0,
            "failed": 0
        }
    }
}
```


Logs de Sincronización

Endpoint: `GET /sync/api/logs`

Query Parameters:
- `limit` (integer): Número de logs a retornar (default: 50)

Response:


```
{
    "success": true,
    "logs": [
        {
            "id": 1,
            "sync_type": "full",
            "status": "success",
            "records_sent": 25,
            "records_received": 10,
            "started_at": "2024-12-12T10:30:00",
            "completed_at": "2024-12-12T10:31:45",
            "duration_seconds": 105
        }
    ]
}
```

⚠️ Códigos de Error

HTTP Status Codes

- `200` : OK - Solicitud exitosa
- `201` : Created - Recurso creado exitosamente
- `400` : Bad Request - Datos inválidos
- `401` : Unauthorized - No autenticado
- `403` : Forbidden - Sin permisos
- `404` : Not Found - Recurso no encontrado
- `500` : Internal Server Error - Error del servidor

Formato de Error

```
{
    "success": false,
    "message": "Descripción del error",
    "error_code": "ERROR_CODE",
    "details": {}
}
```

📦 Ejemplos de Uso

Python (requests)

```python
import requests
import base64

# Login
response = requests.post(
    'http://localhost:5000/auth/login',
    json={
        'username': 'admin',
        'password': 'admin123'
    }
)

session = requests.Session()
session.cookies = response.cookies

# Listar empleados
response = session.get('http://localhost:5000/employees/api/employees')
employees = response.json()['employees']

# Registrar entrada con foto
with open('foto.jpg', 'rb') as f:
    photo_base64 = base64.b64encode(f.read()).decode()

response = session.post(
    'http://localhost:5000/attendance/api/attendance/checkin',
    json={'photo': f'{photo_base64}'}
)

print(response.json())
```

JavaScript (Fetch API)

```javascript
// Login
fetch('/auth/login', {
    method: 'POST',
    headers: {'Content-Type': 'application/json'},
    body: JSON.stringify({
        username: 'admin',
        password: 'admin123'
    })
})
.then(response => response.json())
.then(data => console.log(data));

// Listar empleados
fetch('/employees/api/employees')
    .then(response => response.json())
    .then(data => {
        console.log(data.employees);
    });

// Registrar entrada
fetch('/attendance/api/attendance/checkin', {
    method: 'POST',
    headers: {'Content-Type': 'application/json'},
    body: JSON.stringify({
        photo: photoBase64
    })
})
.then(response => response.json())
.then(data => console.log(data));
```

📝 **Documentación actualizada**: Diciembre 2024