

- **PULSEPING — DATABASE SCHEMA DESIGN DOCUMENT :**

1. Overview :

The PulsePing monitoring system uses a PostgreSQL relational database to store application data such as users, monitored URLs, status check logs, and alert notifications.

The database is normalized, relational, and optimized for uptime/downtime monitoring workloads.

The schema ensures:

- Data integrity
- Clean relationships
- Easy querying
- Scalable log storage

2. Entity Relationship Overview:

PulsePing includes four core entities:

1. **Users** → Represents registered users
2. **URLs** → Websites users want to monitor
3. **Check Logs** → Every monitoring result (status, response time)
4. **Alerts** → Notifications sent when status changes

Relationships:

- One user → many URLs
- One URL → many check logs
- One URL → many alerts

3. Tables and Field Definitions:

Below is the detailed schema for each table:

3.1 USERS TABLE

Stores user credentials and identity information.

Field	Type	Constraints
id	SERIAL	PRIMARY KEY
name	VARCHAR(100)	NOT NULL
email	VARCHAR(255)	UNIQUE, NOT NULL
Password_hash	TEXT	NOT NULL
Created_at	TIMESTAMP	DEFAULT NOW()

3.2 URLs TABLE

Stores the list of URLs a user is tracking.

Field	Type	Constraints
id	SERIAL	PRIMARY KEY
user_id	INT	FOREIGN KEY → users(id), ON DELETE CASCADE
url	TEXT	NOT NULL
expected_status	INT	DEFAULT 200
is_active	BOOLEAN	DEFAULT TRUE
last_status	VARCHAR(50)	NULL
last_response_time	FLOAT	NULL
created_at	TIMESTAMP	DEFAULT NOW()

3.3 CHECK_LOGS TABLE

Stores every check performed by the monitoring worker.

Field	Type	Constraints
id	SERIAL	PRIMARY KEY
url_id	INT	FOREIGN KEY-> urls(id)
status	VARCHAR(50)	NOT NULL
response_time	FLOAT	NULL
checked_at	TIMESTAMP	DEFAULT NOW()

3.4 ALERTS TABLE

Stores alerts sent to users when a URL changes status.

Field	Type	Constraints
id	SERIAL	PRIMARY KEY
url_id	INT	FOREIGN KEY → urls(id)
previous_status	VARCHAR(50)	NULL
current_status	VARCHAR(50)	NULL
alert_type	VARCHAR(50)	CHECK(alert_type IN ('email', 'SMS'))
sent_at	TIMESTAMP	DEFAULT NOW()

4. SQL Implementation:

```
CREATE TABLE users (
    id SERIAL PRIMARY KEY,
    name VARCHAR(100) NOT NULL,
    email VARCHAR(255) UNIQUE NOT NULL,
    password_hash TEXT NOT NULL,
    created_at TIMESTAMP DEFAULT NOW()
);
```

```
CREATE TABLE urls (
    id SERIAL PRIMARY KEY,
    user_id INT REFERENCES users(id) ON DELETE CASCADE,
    url TEXT NOT NULL,
    expected_status INT DEFAULT 200,
    is_active BOOLEAN DEFAULT TRUE,
    last_status VARCHAR(50),
    last_response_time FLOAT,
    created_at TIMESTAMP DEFAULT NOW()
);
```

```
CREATE TABLE check_logs (
    id SERIAL PRIMARY KEY,
    url_id INT REFERENCES urls(id) ON DELETE CASCADE,
    status VARCHAR(50) NOT NULL,
    response_time FLOAT,
    checked_at TIMESTAMP DEFAULT NOW()
);
```

```
CREATE TABLE alerts (
    id SERIAL PRIMARY KEY,
```

```
url_id INT REFERENCES urls(id) ON DELETE CASCADE,  
previous_status VARCHAR(50),  
current_status VARCHAR(50),  
alert_type VARCHAR(50) CHECK (alert_type IN ('email', 'sms')),  
sent_at TIMESTAMP DEFAULT NOW()  
);
```