勤怠管理システム 設計書

1. システム概要

システムの目的と概要

本システムは、従業員の勤怠管理を効率化するためのWeb APIを提供します。従業員の出退勤記録や月次レポートの生成をサポートし、勤怠データの一元管理を実現します。

主な機能一覧

- ユーザー管理
  - ユーザーの登録、一覧取得
- 勤怠記録
  - 出勤打刻（チェックイン）
  - 退勤打刻（チェックアウト）
- レポート生成
  - 月次勤怠レポートの取得
- 健全性確認
  - システムの稼働状況確認（ヘルスチェック）

---

2. アーキテクチャ設計

使用技術

- フレームワーク: Flask
- データベース: SQLite
- **ORM**: SQLAlchemy
- その他ライブラリ: Flask-Migrate, Marshmallow

コンポーネント構成

```
+--------------------+

|   クライアント     |

+--------------------+

         |

         v

+--------------------+

|   Flask API層      |

| - ルーティング     |

| - バリデーション   |
```

```
+--------------------+

         |

         v

+--------------------+

|   データベース層   |

| - SQLAlchemy ORM  |

| - SQLite          |

+--------------------+
```

---

3. データベース設計

ER図

```
[User]

+----+------------+----------+

| id | username    | PK, UNIQUE |

|    | full_name   | NOT NULL |

|    | email       | UNIQUE, NOT NULL |

|    | is_active   | DEFAULT True |

+----+------------+----------+

[Attendance]

+----+------------+----------+

| id | user_id     | FK -> User.id |

|    | date        | NOT NULL, UNIQUE(user_id, date) |

|    | check_in    | NULLABLE |

|    | check_out   | NULLABLE |

|    | note        | NULLABLE |

+----+------------+----------+
```

テーブル定義

Userテーブル

カラム名	型	制約
id	Integer	主キー
username	String(80)	一意制約, 非NULL
full_name	String(120)	非NULL
email	String(120)	一意制約, 非NULL
is_active	Boolean	デフォルト: True

Attendanceテーブル

カラム名	型	制約
id	Integer	主キー
user_id	Integer	外部キー(User.id), 非NULL
date	Date	非NULL, 一意制約(user_id, date)
check_in	Time	NULLABLE
check_out	Time	NULLABLE
note	String(255)	NULLABLE

---

4. 機能設計

API仕様

ユーザー作成

- エンドポイント: `/api/users`
- メソッド: POST

- リクエスト:
- ```
  {
    "username": "alice",
    "full_name": "Alice Tanaka",
    "email": "alice@example.com"
  ```
  ```

```
}
```

レスポンス:

```
{
  "user": {
    "id": 1,
    "username": "alice",
    "full_name": "Alice Tanaka",
    "email": "alice@example.com",
    "is_active": true
  }
}
```

出勤打刻（チェックイン）

- エンドポイント: /api/attendances/checkin
- メソッド: POST

リクエスト:

```
{
  "username": "alice",
  "date": "2025-10-01",
  "time": "09:00"
}
```

レスポンス:

```
{
  "attendance": {
    "id": 1,
    "user_id": 1,
    "date": "2025-10-01",
    "check_in": "09:00:00",
    "check_out": null,
    "note": null
  }
}
```

月次レポート

- エンドポイント: /api/reports/monthly
- メソッド: GET

リクエスト:

```
/api/reports/monthly?username=alice&year=2025&month=10
```

レスポンス:

```
{
  "username": "alice",
```

```
    "year": 2025,
    "month": 10,
    "total_days_recorded": 2,
    "total_work_minutes": 960,
    "details": [
      {
        "id": 1,
        "user_id": 1,
        "date": "2025-10-01",
        "check_in": "09:00:00",
        "check_out": "17:30:00",
        "note": "通常出勤",
        "work_minutes": 510
      }
    ]
  }
```

バリデーションとエラー処理方針

- 必須項目が欠けている場合: HTTP 400 (Bad Request)
- リソースが存在しない場合: HTTP 404 (Not Found)
- サーバーエラー: HTTP 500 (Internal Server Error)

---

5. 業務フロー

1. ユーザー登録
    - 管理者が新規ユーザーを登録。
2. 出勤打刻
    - ユーザーが出勤時にチェックイン。
3. 退勤打刻
    - ユーザーが退勤時にチェックアウト。
4. 月次レポート生成
    - 管理者が指定ユーザーの勤怠サマリを取得。

---

6. 非機能要件

セキュリティ

- APIリクエストの認証・認可（未実装だが、JWT等の導入を推奨）
- SQLインジェクション対策（SQLAlchemyを使用）

パフォーマンス

- 小規模データ向け（SQLiteを使用）

- 大規模データの場合、PostgreSQL等への移行を推奨

運用保守

- ログ出力の整備
- データベースのバックアップスクリプトの導入

以上