U-NEXT HOLDINGS

Gambling
プロジェクトドキュメント
～超簡易版～

要件定義書・基本設計書
ER図・画面遷移図

1. プロジェクト概要

U-NEXT HOLDINGS

1.1 プロジェクト名

【Gambling】

1.2 目的

プレイヤーが様々なギャンブルゲームで
借金を返済するシミュレーションを提供

1.3 背景・概要

- Flask と Flask-SocketIO を用いたウェブアプリケーション
- セッション管理によりユーザーごとの進行状況
  └(所持金／借金／運／ア　イテム)を保持
- フロントエンドは HTML/CSS/JavaScript、静的アセットとして画像と音声を利用

2. 要件定義書

U-NEXT HOLDINGS

2.1 機能要件

2.1.1 共通機能

- /start：ゲーム開始画面
- /create：難易度選択と初期設定
- /lobby：ゲーム一覧と遷移
- /item：アイテム購入画面
- /reset：進行状況リセット
- /clear：クリア画面
- /game_over：ゲームオーバー画面（専用BGM再生含む）
- セッション管理：所持金、借金、運ステータス、アイテム在庫
- BGM／効果音管理（static/audio + audio.js）

2. 要件定義書

U-NEXT HOLDINGS

2.1 機能要件

2.1.2 ゲーム別機能

- chohan（丁半）
  └賭け金入力 → 勝敗判定 → 結果表示 → セッション更新
- highlow（ハイアンドロー）
  └賭け金入力 → カード比較 → 結果表示 → セッション更新
- scratch（スクラッチくじ）
  └スクラッチ実行 → ランダム当たり判定 → 画像表示 → セッション更新
- fx（FXシミュレーション）
  └為替レート擬似乱数生成 → 売買処理 → 結果表示 → セッション更新
- poker（PvPポーカー）
  └Socket.IO マッチング → カード配布 → ベット／コール／チェック → 結果同期

2. 要件定義書

U-NEXT HOLDINGS

2.2 非機能要件

- 言語・フレームワーク
  └Python 3.13.3, Flask 2.3.0, Flask-SocketIO 5.3.2, Flask-Session 0.8.0
- 依存管理
  └package-list.txtにパッケージリストを記載
- セッション保存
  └ファイルベース（Flask-Session 標準設定）
- 同時接続数
  └ポーカーのみ複数接続対応
- パフォーマンス
  └静的ファイルはキャッシュヘッダー設定推奨
- セキュリティ
  └サーバーサイドセッション、CSRF対策は必要に応じて実装

3. 基本設計

U-NEXT HOLDINGS

3.1 全体構成

```
Gambling/
├ app.py              # エントリポイント・Blueprint登録・ルート定義
├ config.py           # 難易度・アイテム定義・calc_luck()
├ extensions.py       # Flask-SocketIO / Flask-Session 初期化
├ package-list.txt    # 依存パッケージ一覧
├ games/
│   ├ chohan.py
│   ├ highlow.py
│   ├ scratch.py
│   ├ fx.py
│   └ poker.py
├ templates/
│   ├ chohan/
│   ├ highlow/
│   ├ scratch/
│   ├ fx/
│   └ poker/
│
└ static/
    ├ css/style.css
    ├ js/
    │   ├ socket.io.js
    │   └ audio.js
    ├ img/.png
    └ audio/.mp3
```

3. 基本設計

U-NEXT HOLDINGS

3.2 モジュール設計

モジュール	役割
app.py	アプリ起動・ルート管理・Blueprint登録
config.py	ゲーム設定値・アイテム設定・運計算ロジック
extensions.py	Flask-SocketIO / Flask-Session 初期化
games/*.py	各ギャンブルゲームビジネスロジック実装
templates/	HTML レイアウト・表示ロジック
static/	CSS, JS, 画像, 音声

3. 基本設計

U-NEXT HOLDINGS

3.3 データフロー

1. ユーザーが /start へアクセス → セッション初期化
2. 難易度選択後、/lobby でゲーム選択
3. 各ゲームルートでビジネスロジックを実行→ 結果をテンプレートに渡す
4. 結果表示後、セッションに所持金/借金/運 を更新
5. ゲーム継続 or /game_over or /clear

3.4 セッション管理

Flask-Session を利用しサーバーサイドに保存

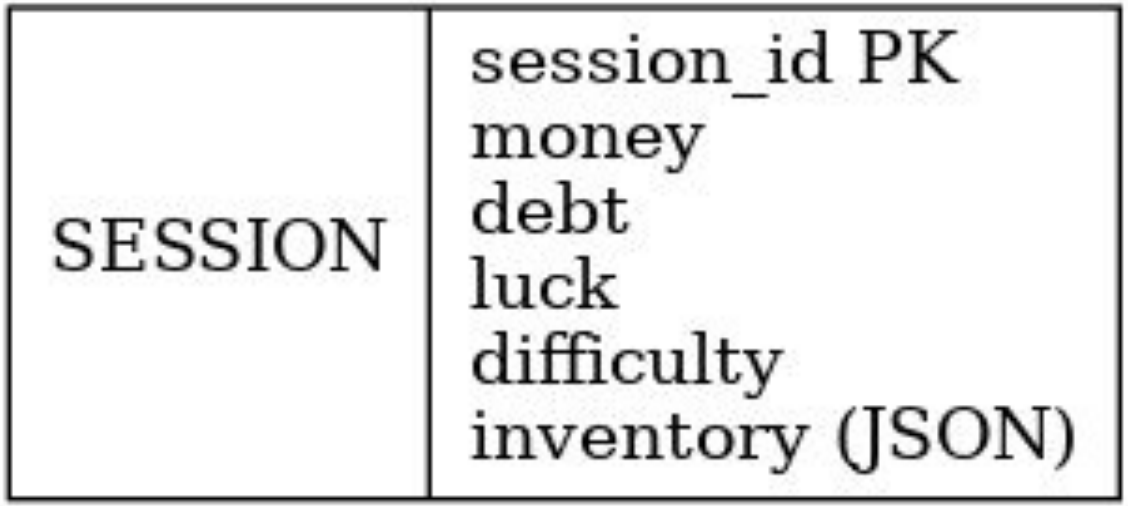
└キー: money, debt, luck, difficulty, inventory

4 README（マニュアル）

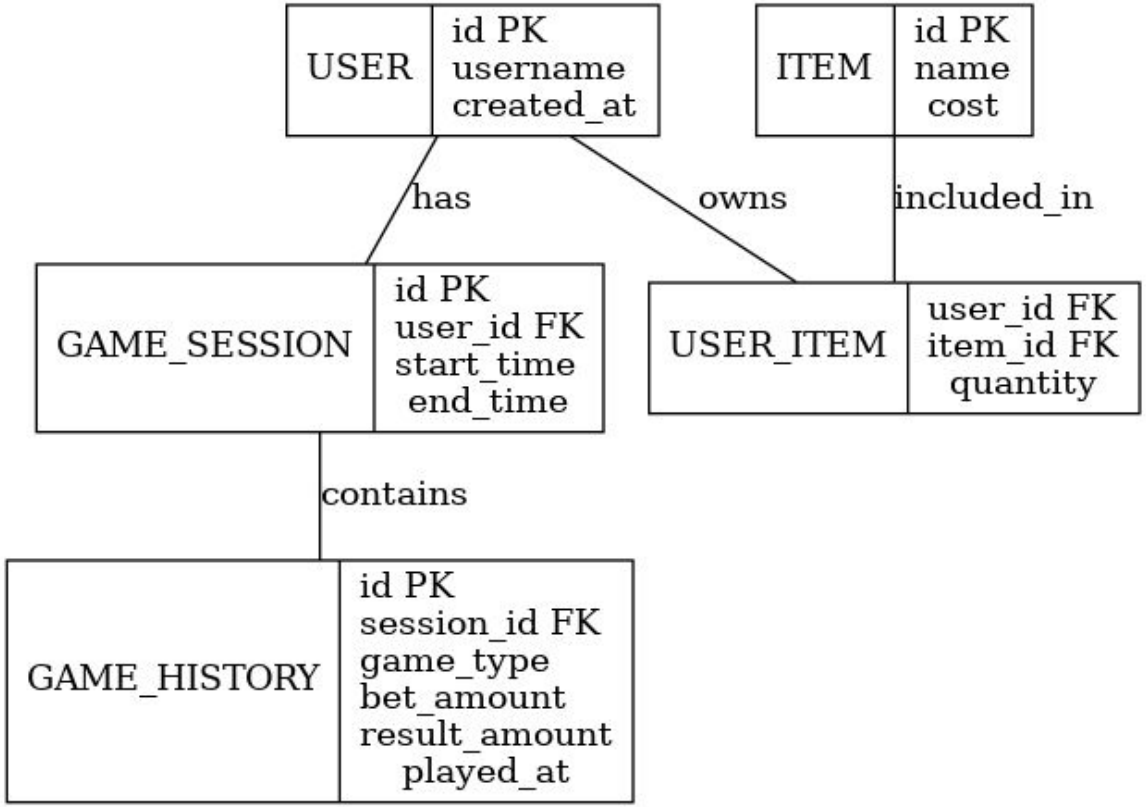
readme.txtに記載

5.1 ER図 (セッション)

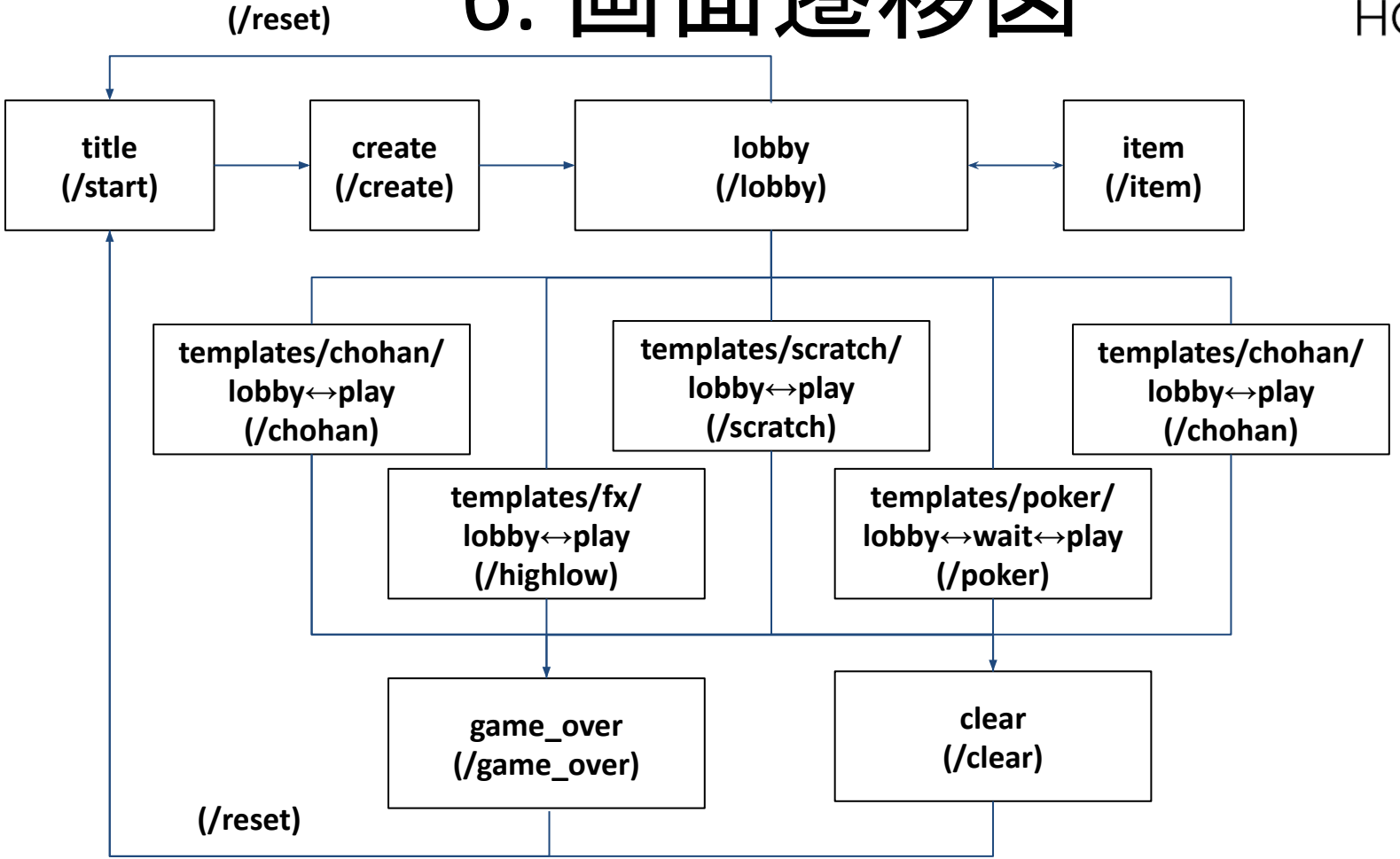
U-NEXT HOLDINGS



5.2 ER図 (DB設計用)


USER
id PK
username
created_at
ITEM
id PK
name
cost
has
owns
included_in
GAME_SESSION
id PK
user_id FK
start_time
end_time
USER_ITEM
user_id FK
item_id FK
quantity
contains
GAME_HISTORY
id PK
session_id FK
game_type
bet_amount
result_amount
played_at

6. 画面遷移図


(/reset)
title
(/start)
create
(/create)
lobby
(/lobby)
item
(/item)
templates/chohan/
lobby↔play
(/chohan)
templates/scratch/
lobby↔play
(/scratch)
templates/chohan/
lobby↔play
(/chohan)
templates/fx/
lobby↔play
(/highlow)
templates/poker/
lobby↔wait↔play
(/poker)
game_over
(/game_over)
clear
(/clear)
(/reset)

7 素材

7.1 コード

- 雛形・初期土台→生成AI（GPT）
- 詳細設計・UI→自作

7.2 画像（png）

- 全てオリジナル制作

7.3 音声（mp3）

- BGM→生成AI（https://soundraw.io/ja/）
- SFX（効果音）→フリー素材（https://maou.audio/#google_vignette）