

科目名	年度	レポート番号	クラス	学籍番号	名前
API 実習	2023	4	B	20122077	ROGER MARVIN

レポート(5)で開発する API を使ったシステムについて企画、要件定義を行ってください。ページ数や文字数よりも、読んでわかりやすく書けているかどうかが、点数アップの分かれ目です。本レポートにおける要件定義項目は、本来の要件定義項目から抜粋した簡易な内容になっています。

種別（API 開発＋アプリ開発なのか、API 連携からのサービス開発なのか識別するため必須）

どちらかに○をつけること

独自 API 開発 / API 連携

業務要件

概要

サービスはシングルアクションゲームです。このゲームは PC でやるゲームです。もちろん、ゲームの操作はキーボードとマウスでします。このゲームは API を利用して、プレイヤーのキャラや敵のステータス情報をデータベースから取れます。このゲームを勝つ際に、プレイヤーの達成情報をゲームの公式 web サイトに表示できます。

背景

現在のゲームの開発では、プレイヤーと敵のステータス情報をゲットするのは、一個一個設定しないといけないです。特に、G develop などオープンソースゲームエンジンはデータを入力のはまだ不便です。私はゲームの開発に興味がもっています。現在、使っているゲームエンジンは Gdevelop というエンジンです。このエンジンでは URL から API にリクエスト機能が持っています。そこで、API と連携するゲームを作りたいと思います。

目的

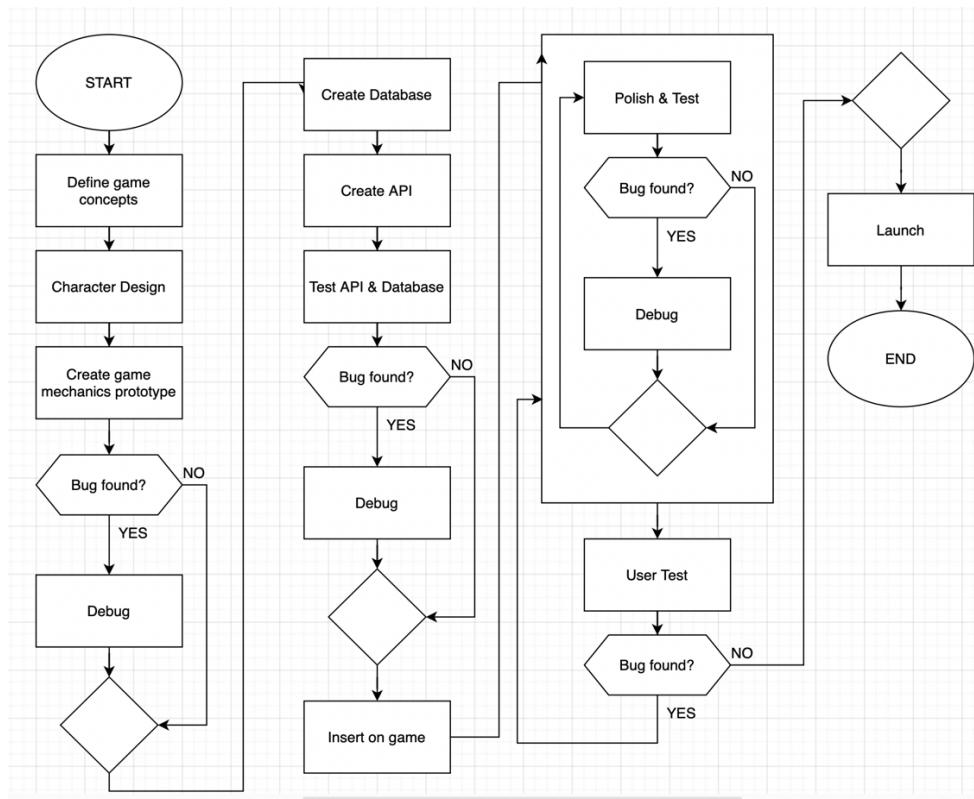
- ガクチカになるため。
- 全員が楽しめるゲームを作りたいです。
- ゲームを開発したい。
- ゲームを売るか、使った API を売ること。
- 就職活動のポートフォリオのためです。

想定利用対象者

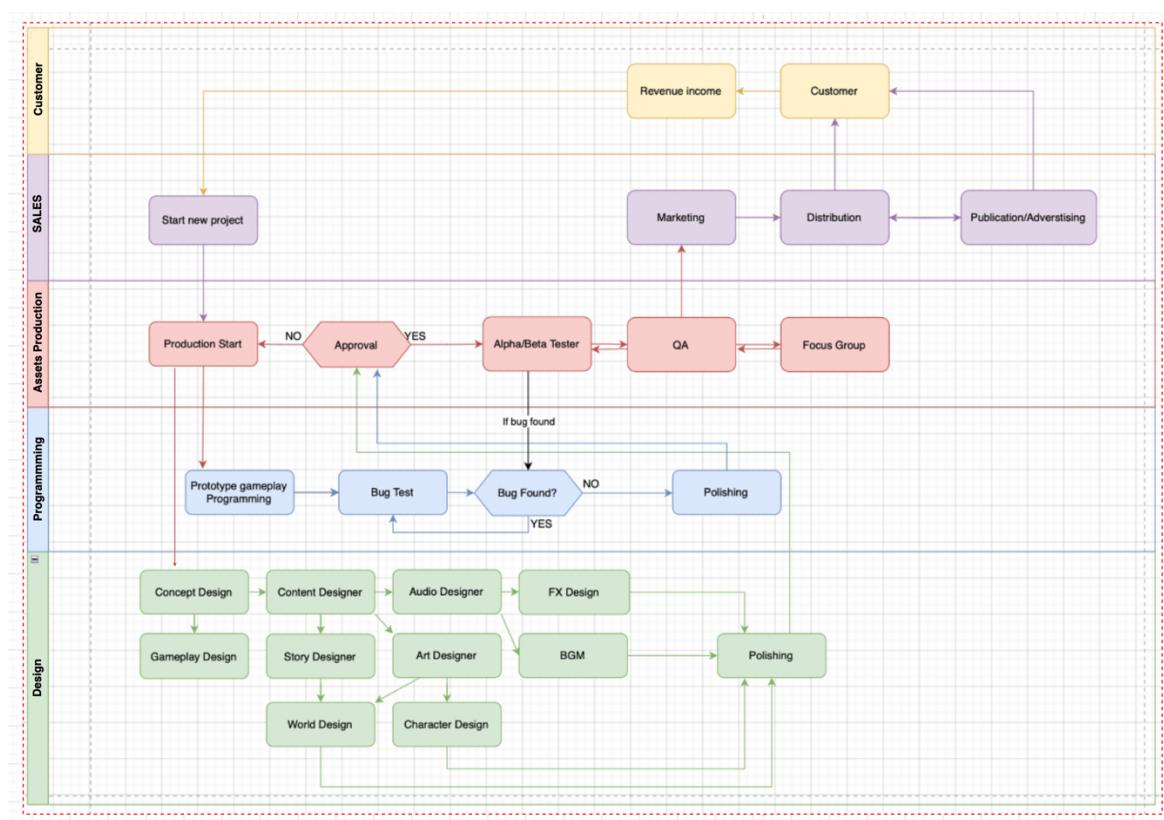
想定利用対象者は 5 歳以上の方々です。

業務フロー

業務フロー（開発過程）

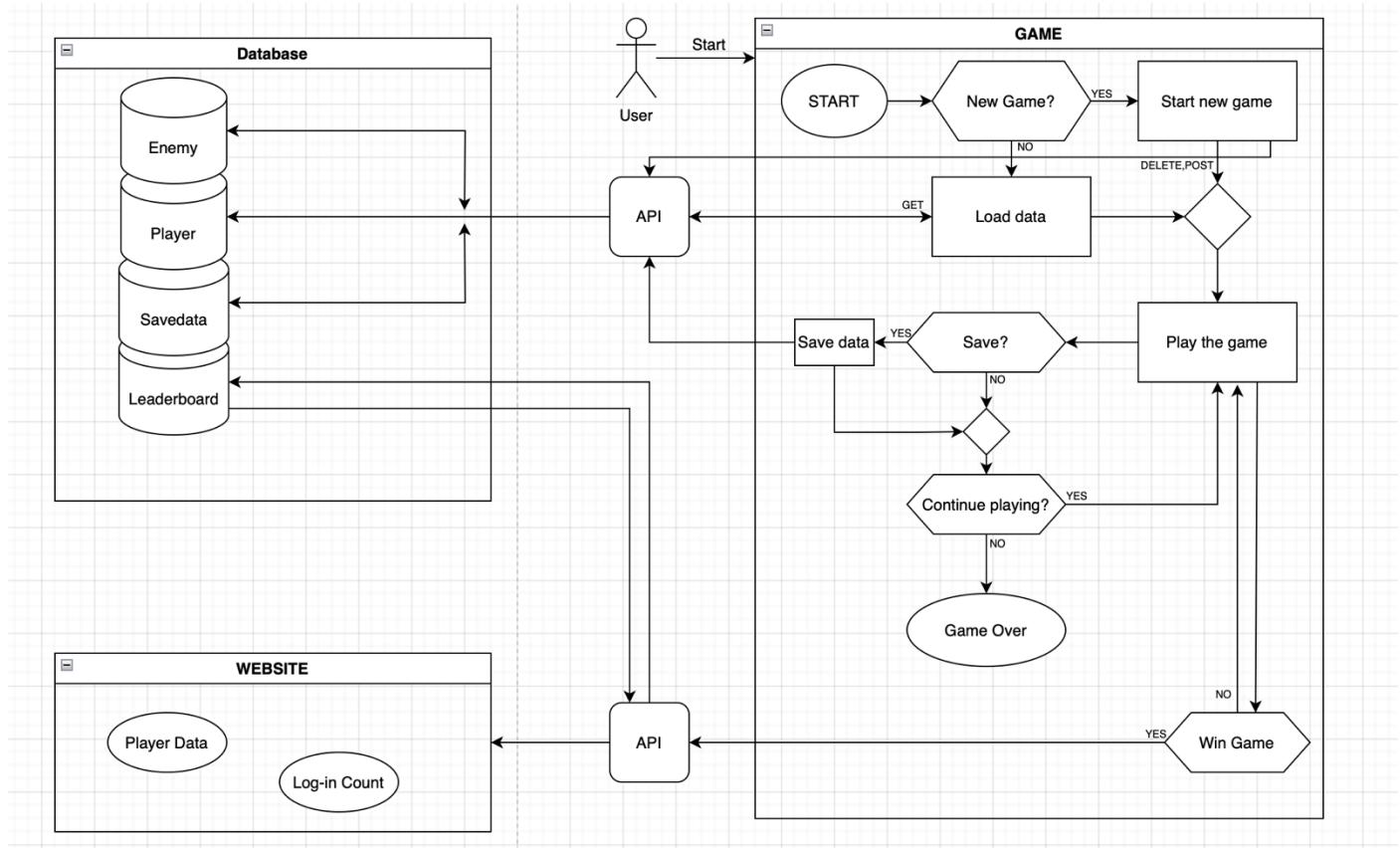


業務フロー（セールス過程）



機能要件

システム構成図



使用外部サービス一覧

ゲームのホームページ :

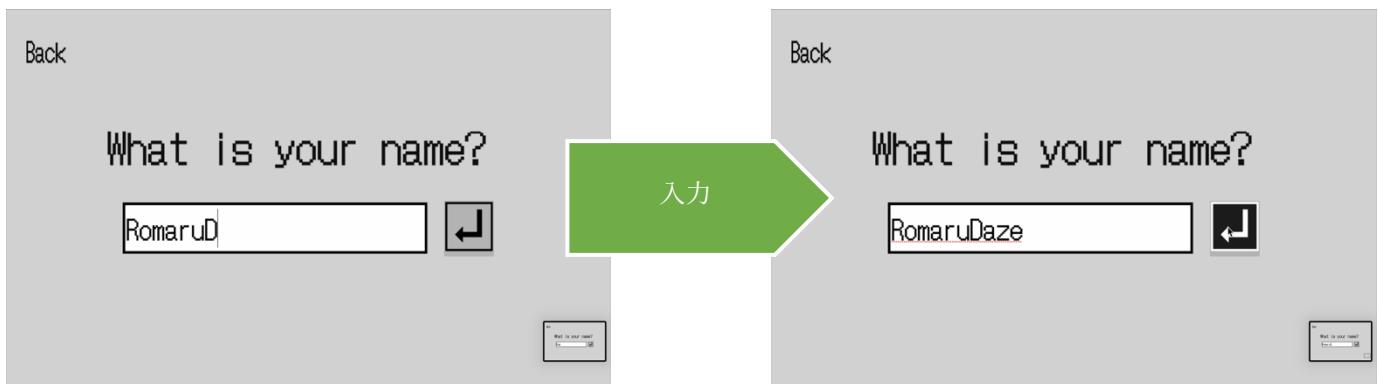
- Email.js → ウェブサイトから管理者のメールメッセージを送るため

想定画面

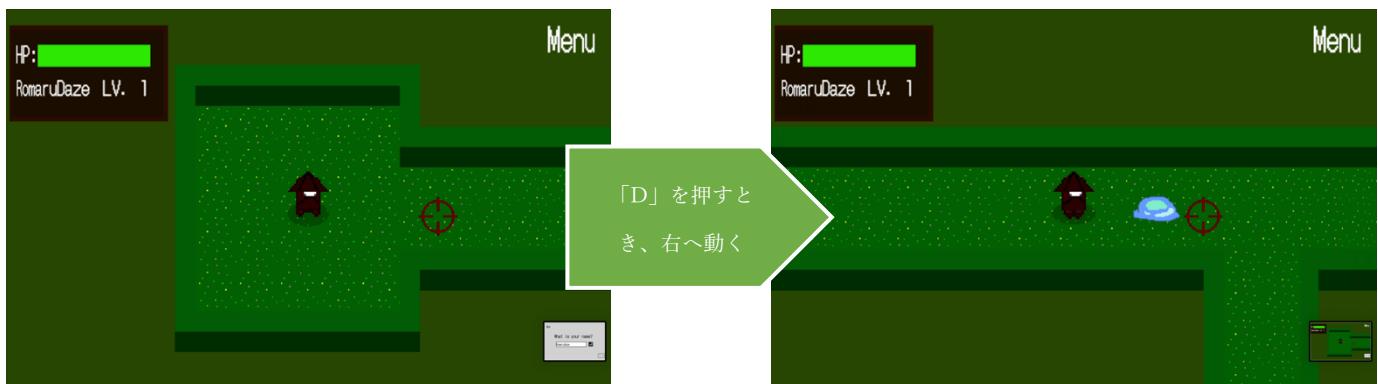
画面一覧

No.	画面
1	スタート画面
2	入力画面
3	クレディット画面
4	ゲームプレイ画面
5	メニュー画面
6	ステータス画面
7	インベントリー画面
8	図鑑画面
9	設定画面

プレイヤーネームを入力画面



キャラを動く画面



敵を攻撃する画面



敵を倒す画面



キャラがレベル UP する画面



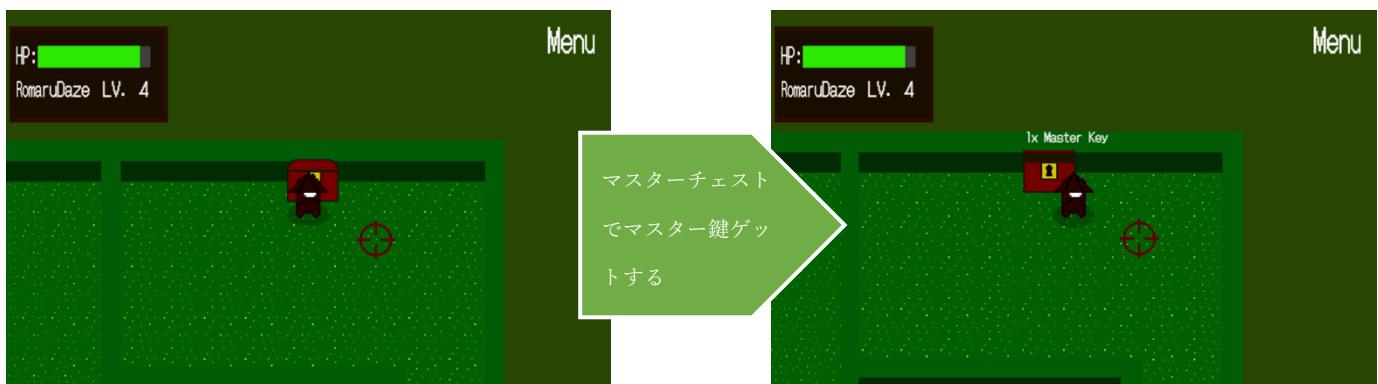
チェスを開ける画面



ドアを開く画面



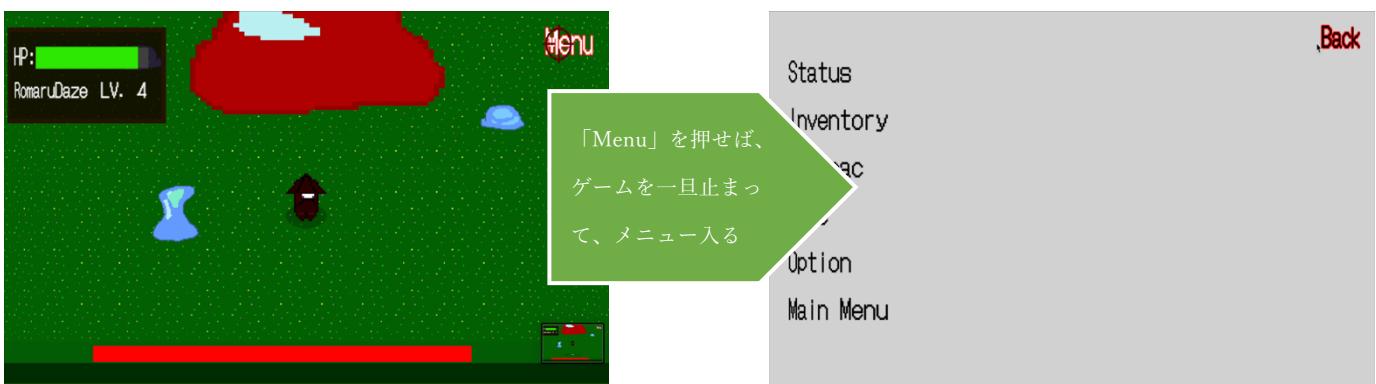
マスターチェスを開ける画面



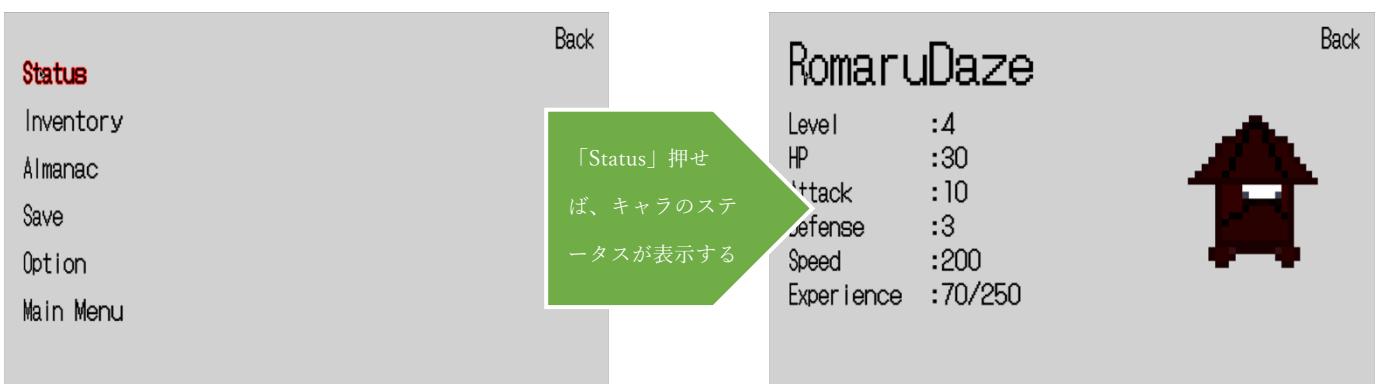
ボス部屋に入る画面



Pause の画面



キャラのステータス画面



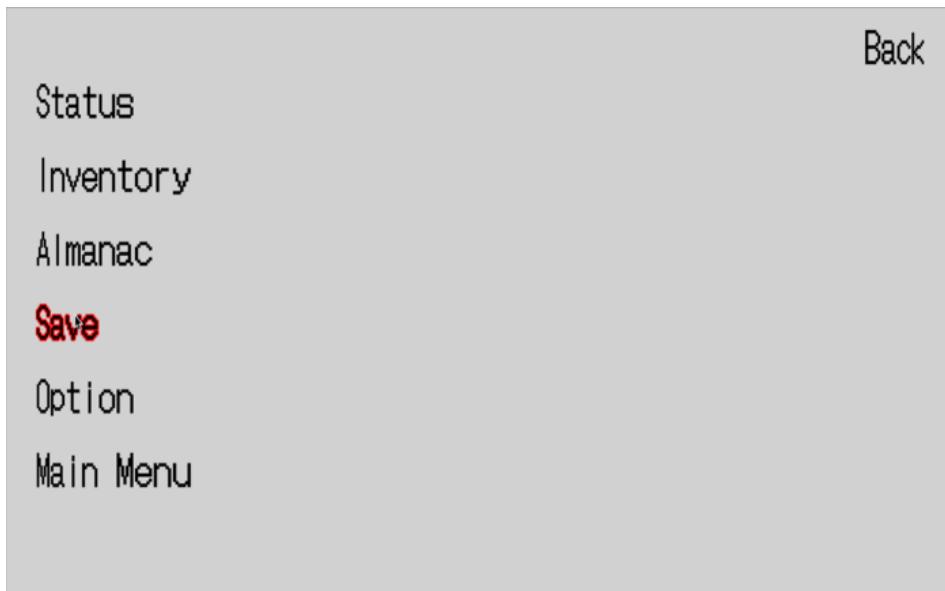
敵の図鑑画面



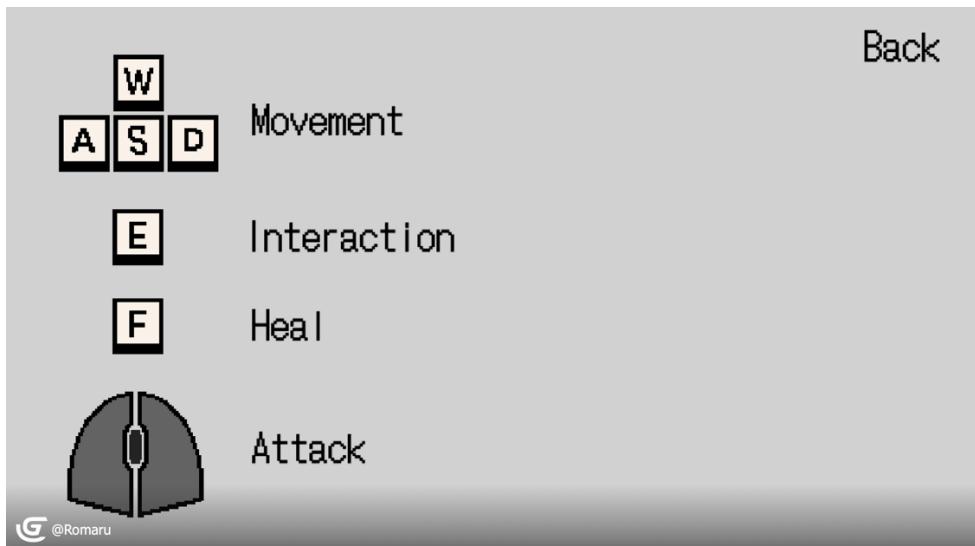
敵のステータス画面



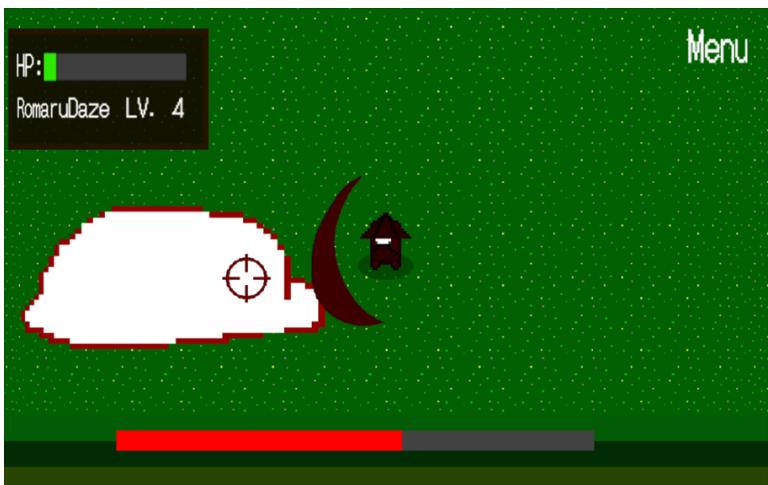
セーブ画面



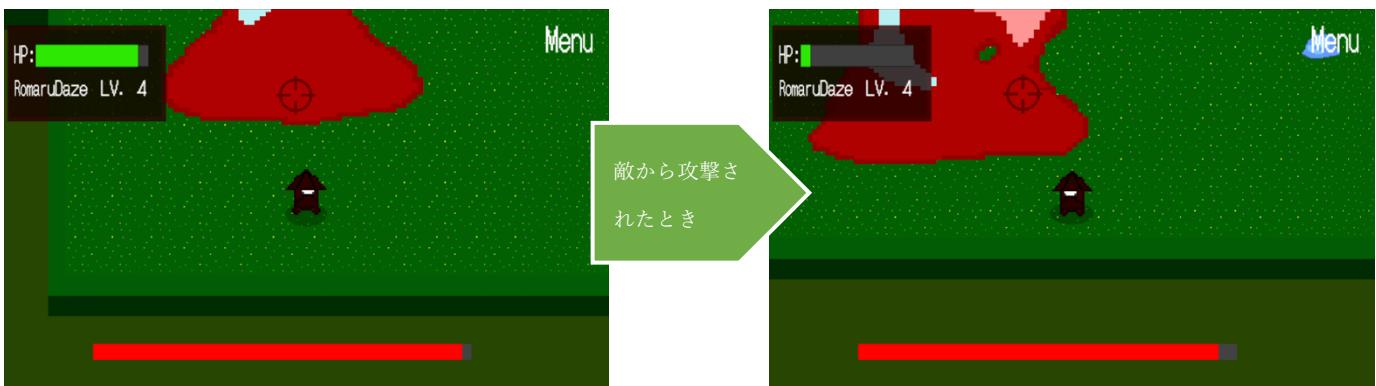
Option 画面



敵をダメージ擧げる画面



敵からダメージもらう画面



次のステージ行く画面

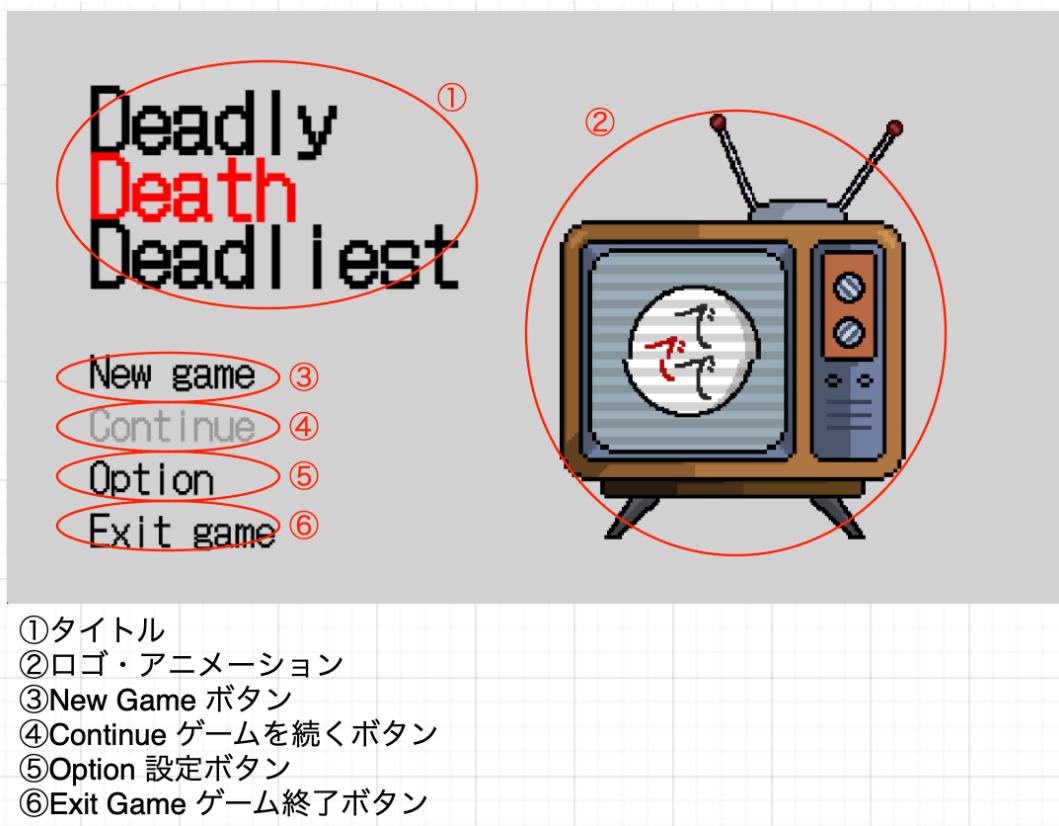


キャラが死んだ画面

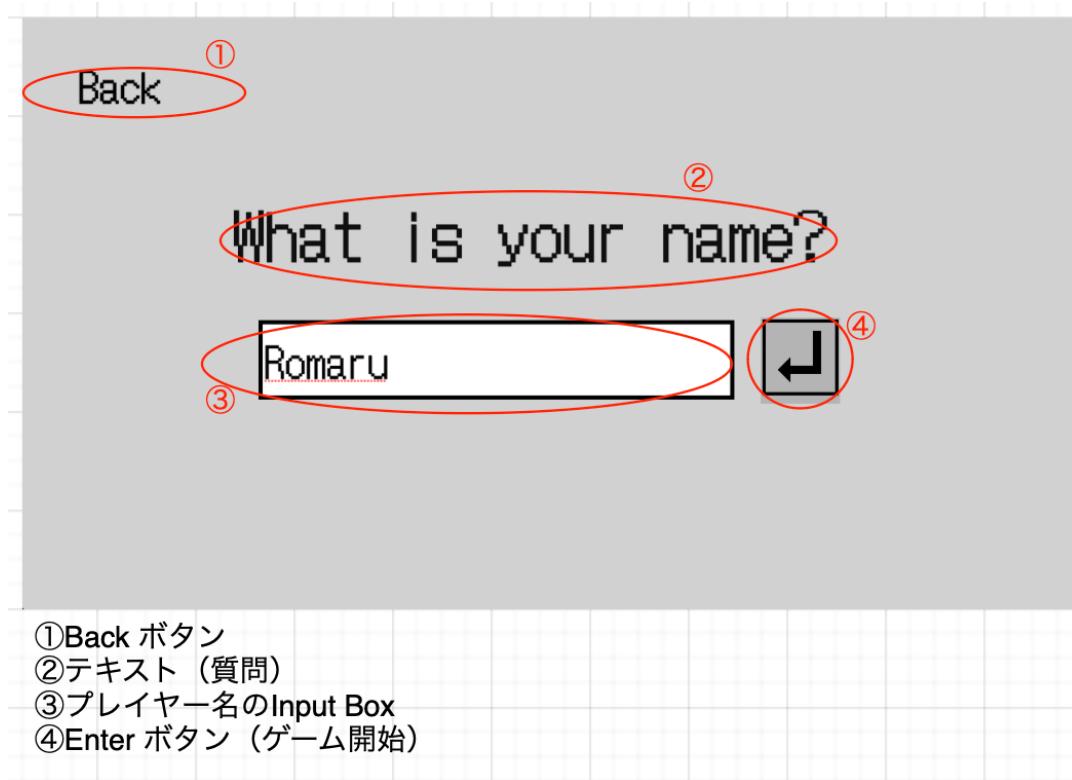


画面レイアウト

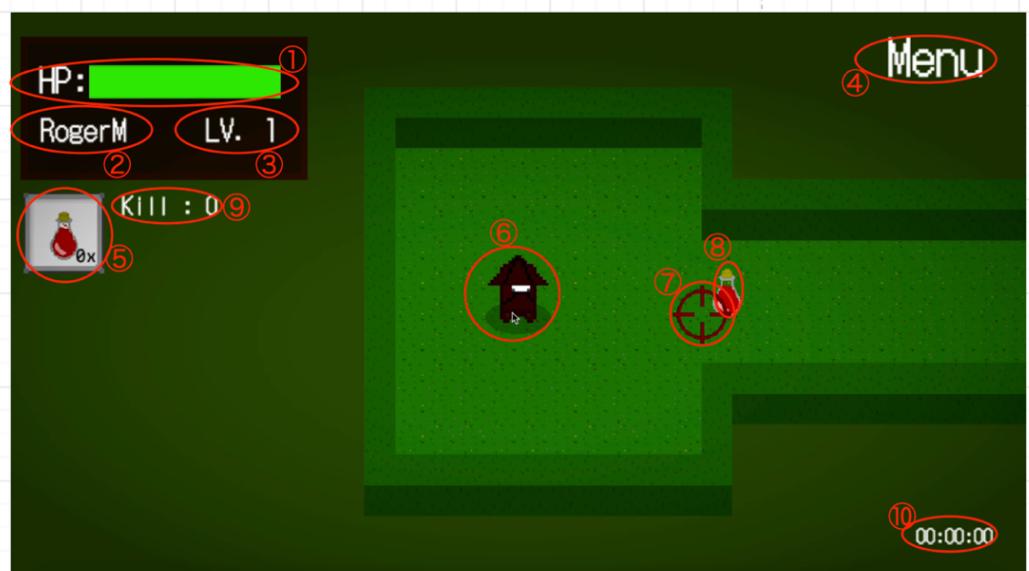
Main Menu 画面・ホーム画面



New Game 画面・新しいゲームを開始する画面

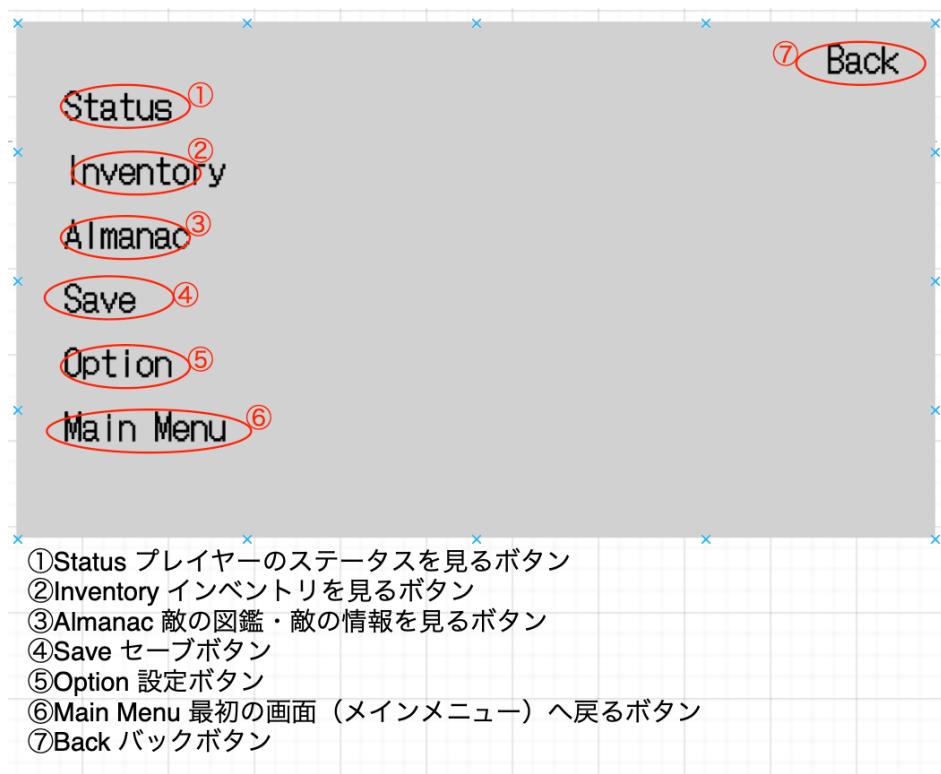


ゲームプレイ画面



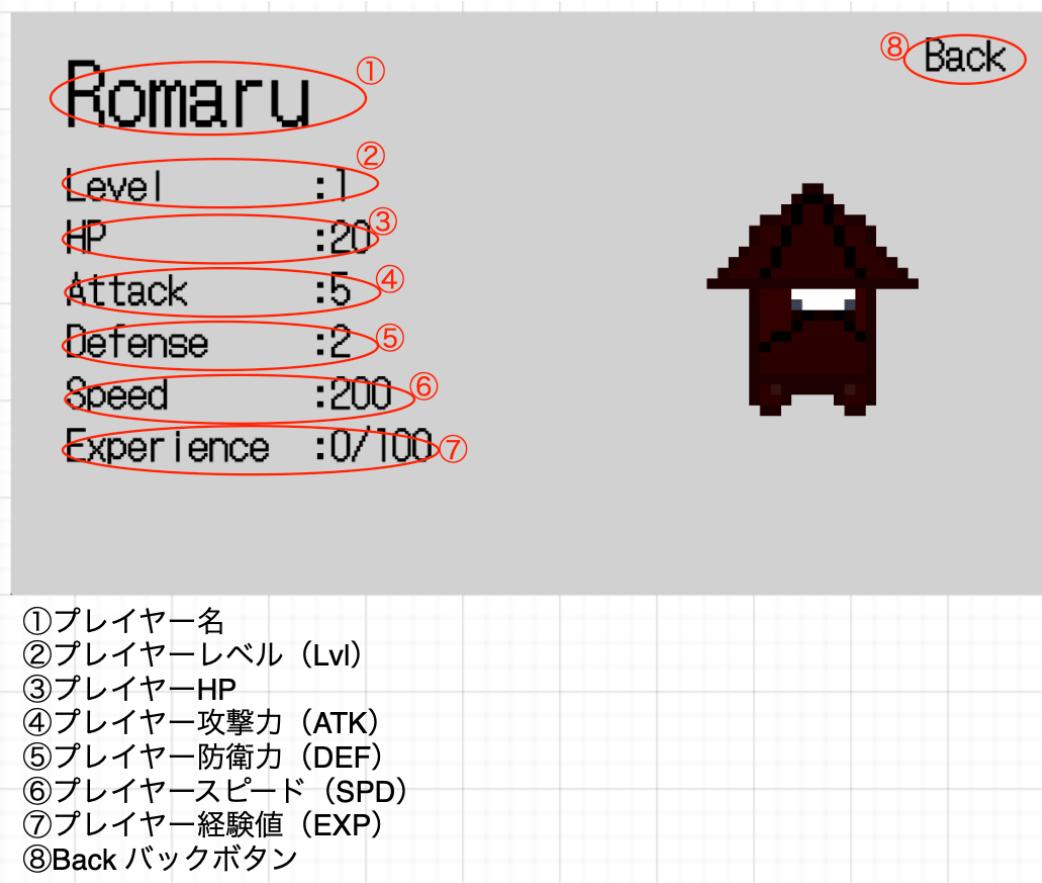
- ① HPバー
- ② プレイヤー名
- ③ プレイヤーレベル
- ④ Menu メニューボタン
- ⑤ Potion Box/Button 押せば、回復する
- ⑥ キャラ (これを動かす)
- ⑦ Cursor
- ⑧ Potion 回復するアイテム
- ⑨ Kill Count
- ⑩ Speedrun Time

メニュー画面



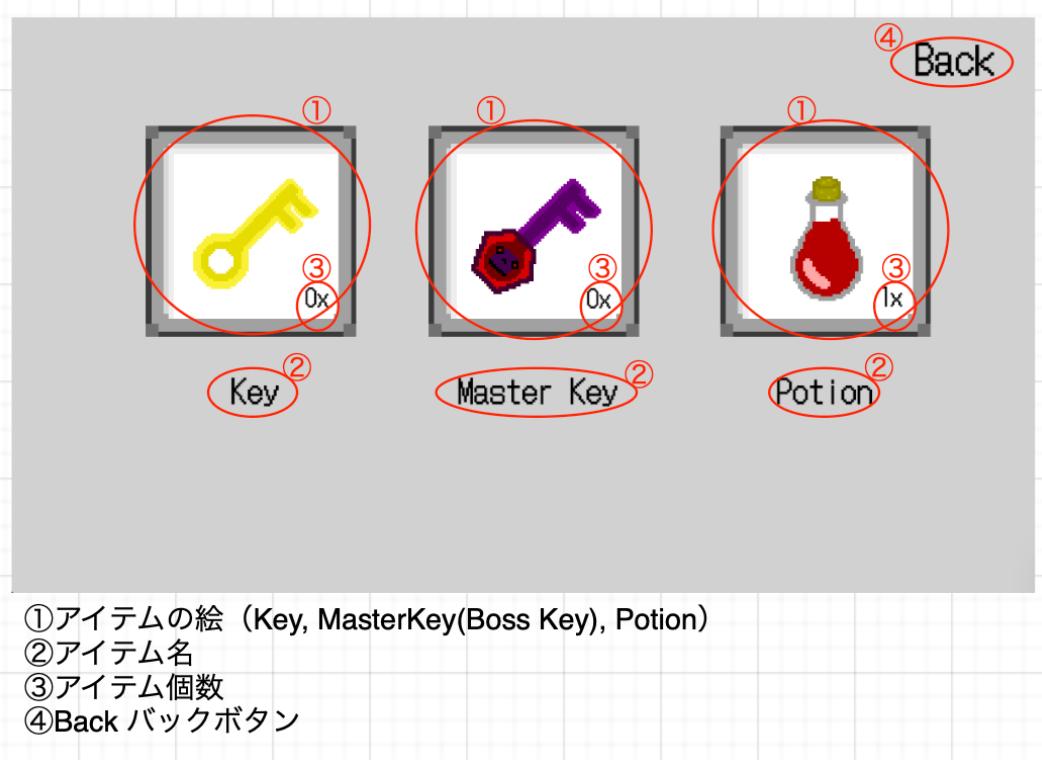
- ① Status プレイヤーのステータスを見るボタン
- ② Inventory インベントリを見るボタン
- ③ Almanac 敵の図鑑・敵の情報を見るボタン
- ④ Save セーブボタン
- ⑤ Option 設定ボタン
- ⑥ Main Menu 最初の画面（メインメニュー）へ戻るボタン
- ⑦ Back バックボタン

プレイヤーステータス画面



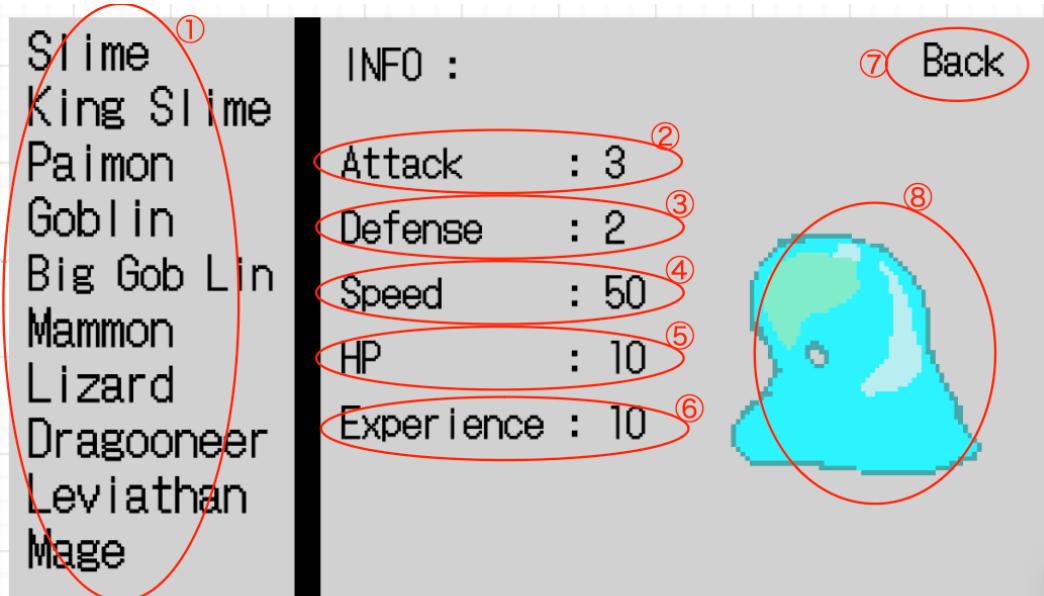
- ①プレイヤー名
- ②プレイヤーレベル (Lvl)
- ③プレイヤーHP
- ④プレイヤー攻撃力 (ATK)
- ⑤プレイヤー防衛力 (DEF)
- ⑥プレイヤースピード (SPD)
- ⑦プレイヤー経験値 (EXP)
- ⑧Back バックボタン

インベントリ画面



- ①アイテムの絵 (Key, MasterKey(Boss Key), Potion)
- ②アイテム名
- ③アイテム個数
- ④Back バックボタン

敵の図鑑・敵の情報・敵のステータス画面



- ①敵名（押せば、敵のステータスとイラスト変わる）
- ②敵の攻撃力（ATK）
- ③敵の防衛力（DEF）
- ④敵のスピード（SPD）
- ⑤敵のHP
- ⑥敵を倒して、貰った経験値（EXP）
- ⑦Back バックボタン
- ⑧敵のイラスト

データ定義（API 開発している場合は必須）

テーブル一覧

player.sqlite(プレイヤーのステータステーブル)

id INTEGER Primary key	lvl INTEGER	hp INTEGER	atk INTEGER	def INTEGER	spd INTEGER	maxexp INTEGER
1	1	20	5	2	200	100
2	2	25	7	2	200	150
3	3	25	9	2	200	200
4	4	30	10	3	200	250
5	5	35	12	5	200	300
6	6	35	14	10	200	350
7	7	40	15	10	200	400
8	8	45	17	15	200	450
9	9	45	19	15	200	500
10	10	50	20	20	200	550

...

enemy.sqlite(敵のステータステーブル)

id INTEGER Primary key	name TEXT	atk INTEGER	def INTEGER	spd INTEGER	exp INTEGER	hp INTEGER
1	Slime	1	2	50	10	10
2	King slime	30	5	25	100	200
3	Paimon	50	7	0	300	500
4	Goblin	15	10	50	10	15
5	Big Gob Lin	20	12	25	100	150
6	Mammon	60	15	25	300	300
7	Lizard	25	10	75	10	20
8	Dragooneer	35	15	50	100	200
9	Leviathan	50	25	75	300	300
10	Mage	40	12	25	10	15

...

savedata.sqlite

Save exist state(セーブデータがあるとき)

Player_name TEXT	Player_level INTEGER	Area_level INTEGER	Player_exp INTEGER	Key INTEGER	Mkey INTEGER	Potion INTEGER	Hours INTEGER	Minutes INTEGER	Seconds INTEGER
Romaru	1	1	0	0	0	0	0	0	0
Romaru	5	2	0	1	1	4	0	3	10
Romaru	5	3	100	1	1	6	0	5	15
Romaru	7	3	100	1	1	8	0	7	12

...

leaderboard.sqlite(ゲームクリアしたプレイヤーの情報のテーブル)

id INTEGER	Player_name INTEGER	Level INTEGER	Kill count INTEGER	Hours INTEGER	Minutes INTEGER	Seconds INTEGER
1	Roger	8	106	0	0	0
2	0	3	10	1	30	15
3	Louis	8	108	1	1	27
4	Rogerm	9	204	0	10	48

非機能要件

拡張性

- ゲームのコンテンツを増やすことです。新しい敵や武器やステージを DLC やアップデートとして追加する。それとも、同じの仕組みのゲームを作れます。例えば、RPG ゲームや MMO ゲームなどで同じのような API を使って、プレイヤーの情報を取れます。
- ゲームの公式ウェブサイトはゲームクリアプレイヤーのスコアやゲームのログイン数のデータを表示できます。この仕組みで、いろいろなものが作れます。例えば、ゲームの代わりに、テスト問題を答えた学生たちの成績のデータを表示します。
- 他には、オンラインストアの商品のパフォーマンスのデータを表示できるものです。例えば、この商品のページを開く回数とか、商品を買う数とか、一年の中で時期ごとに/地域ごとに商品を買うデータ表示できます。そこでデータ分析でも使えるものとなりました。

稼働環境

利用者側

システム側

- ゲームの API のコードを編集や追加をしたい時には VS Code などで編集できます。編集するファイルは Main.py という Python ファイルです。
- ゲームの機能を編集、バグを修正、新しレベルを追加などしたかったら、Gdevelop というゲームエンジンが必要です。この Gdevelop で、ゲーム仕組みを編集だけではなくて、キャラや背景やステージなどをデザインもできます。さらに、Audio FX でも編集できます。
- Gdevelop のデザイン機能は限りがあるため、Aseprite というアプリでピクセルをかけるアプリです。*このアプリは課金です。
- API を起動するためには、ターミナルで 「**pip3 install fastapi uvicorn**」で入力必要です。これは main.py, API を起動するためのライブラリです。
- API を起動するためには、ターミナルで「**python3 -m uvicorn main:app --reload**」を入力してください。

```
○ rogermarvin@macbook gameapi % uvicorn main:app --reload
INFO:     Will watch for changes in these directories: ['/Users/rogermarvin/gameapi']
INFO:     Uvicorn running on http://127.0.0.1:8000 (Press CTRL+C to quit)
INFO:     Started reloader process [50665] using StatReload
INFO:     Started server process [50667]
INFO:     Waiting for application startup.
INFO:     Application startup complete.
```

- 起動できると API が使えることを確認できたら、Gdevelop でゲームプレビューするか、実際にゲームアプリを開けます。
- ここで、ゲームの編集やアップデートをできます

利用者側(クライアント環境)

- このゲームはパソコンのゲームで、パソコンに起動するしかないです。ゲームの操作はキーボードとマウスです。
- キーボードのキーの操作はゲームの Menu の Option の場所にあります。
- ゲームをインストールためには Itch.io で、「Deadly Death Deadliest」を検索すれば、ゲームをダウンロードできます。
- Macbook や Windows を問わず、ゲームを起動できます。
- ゲームをクリアできたら、ゲームの公式サイトに自分の Achievement を見えます。* Top10 のプレイヤーしか表示しないです。
- ゲームをアップデート・パッチする時に、ゲームのサーバーがダウンする時あります。その時には我々からお知らせします。
- バグ、要求、悩みがあれば、ホームページで連絡できます。* 営業時間は 9 AM～9 PM です。返事は 1 日内で返事しますが、日による返事が遅いかもしれません。

保守

- ゲームの保守は 1 ヶ月ごとにやります。保守の内容はゲームのバグを修正するや UI のインターフェースを変化するなどです。
- ゲームのデータベースも管理します。例えば、ゲームのコンテンツを増やしたら、敵のデータやプレイヤーのデータを新しい追加するなどです。
- ゲームの利益は十分もらったら、続きのコンテンツを追加する予定です。
- ユーザーから悩み、バグレポート、要求があれば、技術チームが最短時間で課題を解決します。
- 大バグがあれば、保守のためにサーバーシャットダウン可能性があります。その時には 1 週間前にはホームページにお知らせします。シャットダウン時間は 2 日～3 日かかるとお思いますが、問題による長く/短くなる可能性があります。