DB group by & join

G's ACADEMY FUKUOKA

アジェンダ

- webの仕組み（復習）
- RDBの構造
- テーブルの追加
  - ユーザテーブル
  - いいねテーブル
- いいねボタンの追加とdb操作
- いいね数の表示
- 課題発表→チュータリング(演習)タイム

授業のルール

- 授業中は常にエディタを起動！
- 考えたことや感じたことはzoomチャットでガンガン発信！
- 質問はslackへ！ 他の人の質問にも目を通そう！（同じ質問があるかも）
- 演習時，できた人はスクショなどslackに貼ってアウトプット！
- まずは打ち間違いを疑おう！

  {} ' " ; など
- 書いたら保存しよう！（よく忘れる！）

  command + s

  ctrl + s

PHPの準備

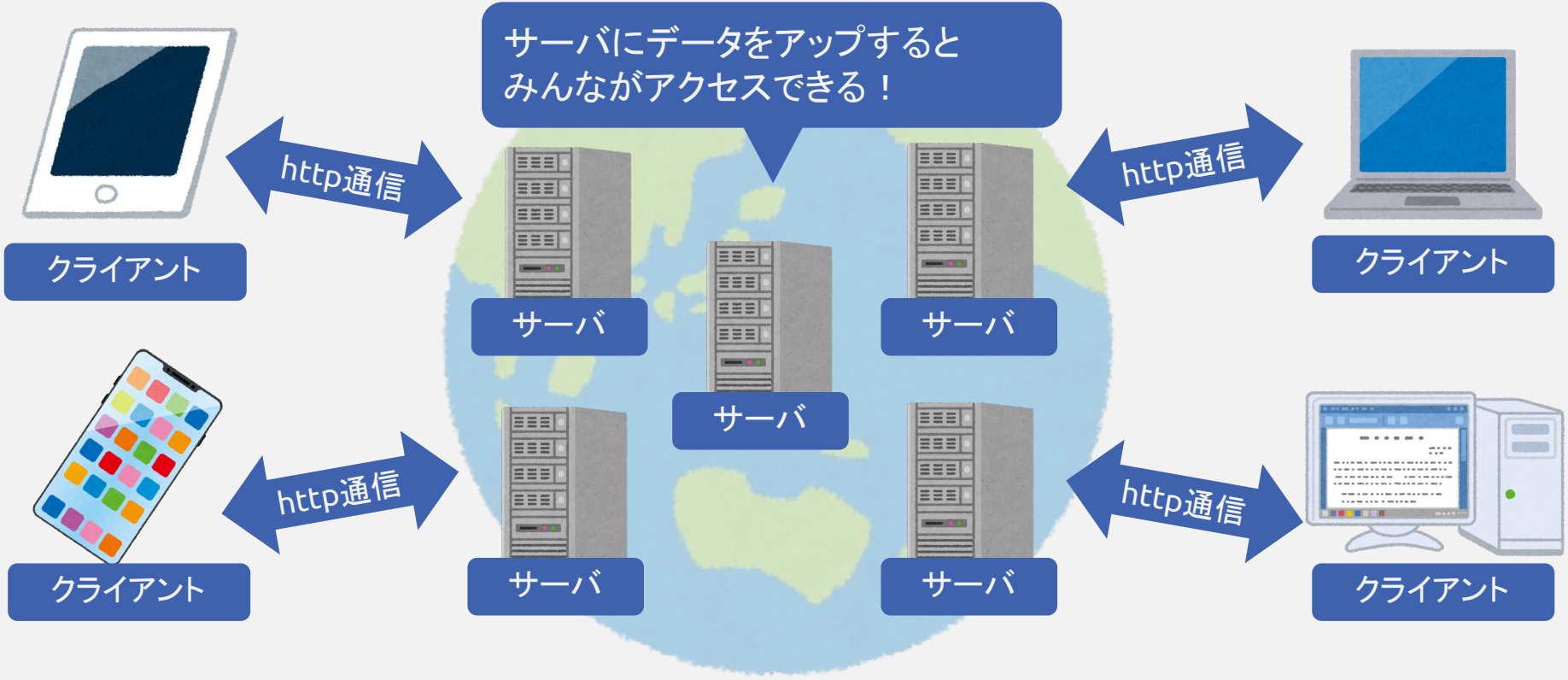
- XAMPPの起動確認
- http://localhost/のアクセス確認
- サンプルフォルダを「htdocs」フォルダに入れる

今日のゴール

- RDB構造を把握！
- RDBの考え方を知る！
- 複数テーブルを操作！

webの仕組み

雑なwebの仕組み



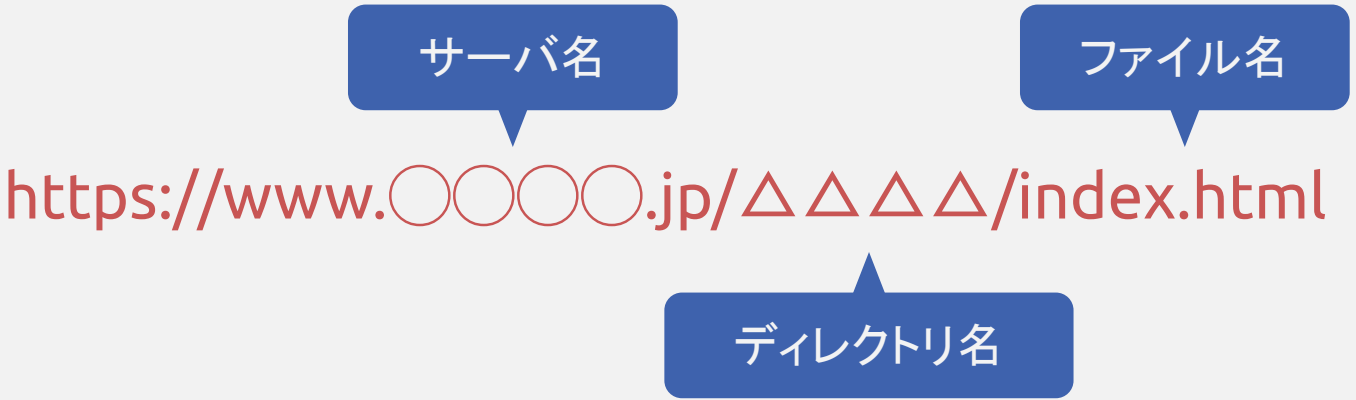
G's ACADEMY FUKUOKA

URL

■URLとは

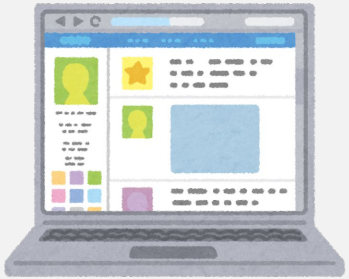
- web上にある情報（ファイル）の場所を指し示す住所.
- Uniform Resource Locatorの略（覚えなくてOK）.

■例



サーバサイド言語の仕組み

※ 言語によらず，ファイル（プログラム）はサーバ上に存在



送られてきたhtmlを実行

- こういう情報がほしい
- こういう処理をしたい
  例：index.phpにアクセス

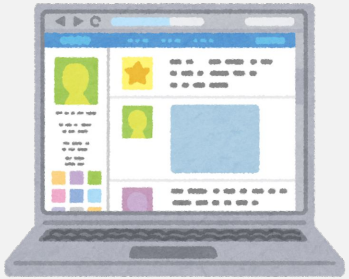
http通信

http通信

- 処理した結果のデータ
- 構成したhtml

データベース（DB）の動き方

サーバ上のプログラムがDBにアクセスして処理を実行！



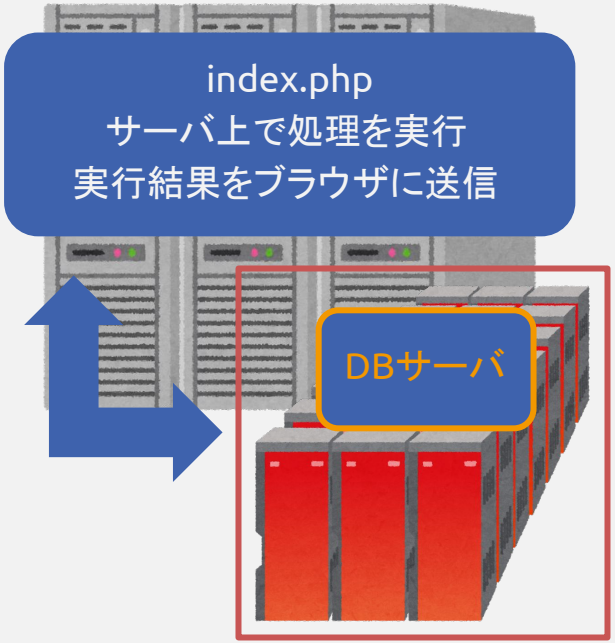
送られてきたhtmlを実行

- こういう情報がほしい
- こういう処理をしたい
  例：index.phpにアクセス

http通信

http通信

- 処理した結果のデータ
- 構成したhtml



RDBの構造

RDBとは

- 複数のテーブルを連携させる
  - 一つのテーブルから取得できる情報は限られている.
  - ECサイトで購入したユーザの情報と商品の情報を両方取得したい場合は. . ？？
- 両方のテーブルを関連付けるテーブルを「SQLで」作成する！
  - 購入履歴テーブルを作成！
  - 「いつ」「どのユーザが」「どの商品を買ったか」を記録する！

RDBとは

id	name	email	password	address	phone	photo
1	こだま	kodama@gs.com	123456	tokyo	11111111	https://gs/files/pics/57834265729.png
2	ごってぃー	goto@gs.com	password	fukuoka	22222222	https://gs/files/pics/57857392758.png
3	あび	abi@gs.com	123456789	fukuoka	33333333	https://gs/files/pics/84674265729.png
4	ゆうき	yuki@gs.com	12345678	nagasaki	44444444	https://gs/files/pics/57834265729.png
5	スティーブ	jobs@icloud.com	12345	u.s.	55555555	https://gs/files/pics/57834265729.png
6	マーク	mark@fb.com	111111	u.s.	66666666	https://gs/files/pics/57834265729.png
7	ビル	bill@ms.com	1234567	u.s.	77777777	https://gs/files/pics/57834265729.png
8	エリック	eric@gmail.com	sunshine	u.s.	88888888	https://gs/files/pics/57834265729.png
9	ァインハルト	reinhard@empire.com	qwerty	empire	99999999	https://gs/files/pics/57834265729.png
10	ジーク	sieg@empire.com	iloveyou	empire	12345678	https://gs/files/pics/57834265729.png

「だれが」

id	product_name	manufacturer	category	price	description	product_image
1	ryzen 1950X	amd	cpu	999	超はやい	https://gs/products/images/5362764375.png
2	ryzen 1920X	amd	cpu	799	はやい	
3	ryzen 1900X	amd	cpu	549	はやい	
4	ryzen 1800X	amd	cpu	499	必要十分	
5	ryzen 1870X	amd	cpu	399	コスパいい	
6	core i7 1068G7	intel	cpu	426	高い	
7	core i7 1065G7	intel	cpu	426	オーバークロックに向く	
8	core i7 1060G7	intel	cpu	340	ちょうどよい	
9	core i5 1035G1	intel	cpu	297	動画編集はきつい	

「何を購入したか」

user_id	product_id	created_at
5	1	2019/12/04
2	1	2019/12/04
4	1	2019/12/03
5	1	2019/12/02
3	1	2019/12/04
1	1	2019/12/04
4	2	2019/12/04
2	2	2019/12/03
5	2	2019/12/03
3	2	2019/12/02
3	3	2019/12/04
4	3	2019/12/04
2	3	2019/12/04
5	4	2019/12/02
1	5	2019/12/04

今回は「いいね機能」を実装

- 前回までに作成したtodoリストにいいね機能を追加！
  - ユーザテーブルといいねテーブルを作成
  - いいねボタン追加
  - いいねボタンクリック時の処理
  - いいね数を表示

テーブルの追加

【課題2】ユーザ管理機能の作成

- ユーザ管理テーブル（←必ず作成，DBはこれまでのものを使用）
  - テーブル名： users_table
- カラム名など

	#	名前	データ型	照合順序	属性	NULL	デフォルト値	コメント	その他	操作
☐	1	id	int(12)			いいえ	なし		AUTO_INCREMENT	変更 削除 その他
☐	2	username	varchar(128)	utf8mb4_unicode_ci		いいえ	なし			変更 削除 その他
☐	3	password	varchar(128)	utf8mb4_unicode_ci		いいえ	なし			変更 削除 その他
☐	4	is_admin	int(1)			いいえ	なし			変更 削除 その他
☐	5	is_deleted	int(1)			いいえ	なし			変更 削除 その他
☐	6	created_at	datetime			いいえ	なし			変更 削除 その他
☐	7	updated_at	datetime			いいえ	なし			変更 削除 その他

いいねテーブル

- いいねテーブル
  - テーブル名： like_table
- カラム名など

	#	名前	データ型	照合順序	属性	NULL	デフォルト値	コメント	その他	操作
☐	1	id 🔑	int(12)			いいえ	なし		AUTO_INCREMENT	変更 削除 その他
☐	2	user_id	int(12)			いいえ	なし			変更 削除 その他
☐	3	todo_id	int(12)			いいえ	なし			変更 削除 その他
☐	4	created_at	datetime			いいえ	なし			変更 削除 その他

いいねボタンの設置

やりたいこと

- いいねボタンをクリック
  - like_tableに「誰が」「何に」いいねをしたのかを追加
- 実装の方針
  - todo_read.phpにいいねボタンを設置
  - GETでtodoのidとユーザのidを送信する
  - 受け取り側のファイル（like_create.php）で受け取ったデータをdbに登録

いいねボタンの設置

todo_read.php

```
  // タグ生成部分
  $output .= "<td><a
href='like_create.php?user_id={$user_id}&todo_id={$record["id"]}'>like</a></td
>";
  $output .= "<td><a href='todo_edit.php?id={$record["id"]}'>edit</a></td>";
  $output .= "<td><a
href='todo_delete.php?id={$record["id"]}'>delete</a></td>";
  $output .= "</tr>";
```

いいねボタンの確認

select.php

こんな感じ！！

いいねボタンにカーソルを載せると
左下にこんな感じで表示される！！

いいねボタンクリック時の動作

いいね追加の処理

受け取り側の処理の流れ

- データの受取
- データベース接続
- テーブルへの登録
- 完了したら一覧画面へ移動

受け取り側の処理

like_create.php

```
// まずはこれ
var_dump($_GET);
exit();

// 関数ファイルの読み込み
include('functions.php');

// GETデータ取得
$user_id = $_GET['user_id'];
$todo_id = $_GET['todo_id'];

// DB接続
$pdo = connect_to_db();
```

受け取り側の処理

like_create.php

```
  $sql = 'INSERT INTO like_table(id, user_id, todo_id, created_at)VALUES(NULL,
:user_id, :todo_id, sysdate())';    // SQL作成

  $stmt = $pdo->prepare($sql);
  $stmt->bindValue(':user_id', $user_id, PDO::PARAM_INT);
  $stmt->bindValue(':todo_id', $todo_id, PDO::PARAM_INT);
  $status = $stmt->execute();           // SQL実行

  if ($status == false) {
    // エラー処理
  } else {
    header('Location:todo_read.php');
  }
```

受け取り側の動作確認

いいねボタンをクリックして，データが入っていればOK！

←T→	id	user_id	todo_id	created_at
☐ 編集 コピー 削除	1	1	10	2020-06-10 11:20:59

いいねしているかどうかで条件分岐

いいねしてなければいいね，していればいいね取り消し

-このままだと．．．

- 連打すれば無限にいいねできてしまう．．．
- いいねしている状況であれば，いいねを取り消す処理にしたい！

- 実装の方針

- いいねボタンをクリックしたtodoとuserでテーブルを検索
- データが1件以上存在すれば削除のSQLを作成して実行
- データが存在しなければ登録のSQL（前項で作成したもの）を実行

いいねしているかどうかのチェック

like_create.php

```
  // いいね状態のチェック（COUNTで件数を取得できる！）
  $sql = 'SELECT COUNT(*) FROM like_table WHERE user_id=:user_id AND
todo_id=:todo_id';
  $stmt = $pdo->prepare($sql);
  $stmt->bindValue(':user_id', $user_id, PDO::PARAM_INT);
  $stmt->bindValue(':todo_id', $todo_id, PDO::PARAM_INT);
  $status = $stmt->execute();
  if ($status == false) {
    // エラー処理
  } else {
    $like_count = $stmt->fetch();
    var_dump($like_count[0]);       // データの件数を確認しよう！
    exit();
  }
```

いいね状態によって条件分岐

like_create.php

```
// いいねしていれば削除，していなければ追加のSQLを作成
if ($like_count[0] != 0) {
  $sql =
     'DELETE FROM like_table WHERE user_id=:user_id AND todo_id=:todo_id';
} else {
  $sql = 'INSERT INTO like_table(id, user_id, todo_id, created_at)
          VALUES(NULL, :user_id, :todo_id, sysdate())';   // 1行で記述！
}

// INSERTのSQLは前項で使用したものと同じ！
// 以降（SQL実行部分と一覧画面への移動）は変更なし！
```

受け取り側の動作確認

←T→	id	user_id	todo_id	created_at
編集 コピー 削除	1	1	10	2020-06-10 11:34:45
編集 コピー 削除	2	1	16	2020-06-10 11:34:54

いいねしたあとでもう一度クリックし，
データが削除されていればOK！

←T→	id	user_id	todo_id	created_at
編集 コピー 削除	2	1	16	2020-06-10 11:34:54

一覧画面でいいね数の表示

一覧画面のいいねボタンにいいね件数を表示

- 一覧画面で何件いいねされているかがわからない．．．
  -いいねしたらいいねの件数が表示されるように！
- 実装の方針
  - todoごとのいいね数を集計
  - 取得したいいね数をtodoのテーブルに結合させる
  - いいね数をいいねボタンに表示する

各todoのいいねされている件数を集計

【解説】GROUP BY

- GROUP BYを使うと集計ができる！
- 例（phpmyadminで実行しよう！）
  - SELECT todo_id, COUNT(id) AS cnt FROM like_table GROUP BY todo_id
  - 「like_table」の「todo_id」ごとに「idの数」を「cnt」というカラム名で表示
- 注意点
  - COUNT()で件数を取得しているが，集計関数以外だとエラーになる
  - その他の集計関数
    - MIN(), MAX(), SUM(), …など

いいね件数集計の確認

- まずは記述したSQLをphpmyadminで実行しよう！

todo_id	cnt
10	1
15	1
16	1

「どのタスクに」「何件」
いいねがついているかが確認できる！

【Point】このSQL文は集計結果のテーブルを返す！！

テーブルの結合

テーブルを結合して一覧表示

todo_read.php

- 集計はできたので，いいねボタンに件数を表示したい！
- 画面の表示はtodo_tableからデータなので，集計結果を組み込めない．．．
  - テーブルを結合させることで，集計結果もまとめて表示できる！
- todo_read.phpのSQL部分を変更する
- SQLの考え方
  - 「todo_table」と「集計結果のテーブル」をつなげる
  - 「todo_tableのid」と「集計結果のテーブルのtodo_id」を対応させる
  - 「集計結果のテーブル」は前項で取得したアレ

テーブルを結合して一覧表示

todo_read.php

```
//データ表示SQL作成
// $sql = 'SELECT * FROM todo_table';     // ←select文を変更
$sql = 'SELECT * FROM todo_table
        LEFT OUTER JOIN (SELECT todo_id, COUNT(id) AS cnt
        FROM like_table GROUP BY todo_id) AS likes
        ON todo_table.id = likes.todo_id';
$stmt = $pdo->prepare($sql);                    // 変更なし
$status = $stmt->execute();                     // 変更なし
```

JOINを使って結合！

テーブルの結合結果

- 記述したSQLをphpmyadminで実行しよう！

id	todo	deadline	created_at	updated_at	todo_id	cnt
4	良い食材を買う	2020-06-19	2020-06-04 11:47:54	2020-06-05 12:01:56	*NULL*	*NULL*
7	PHP選手権準備	2020-06-07	2020-06-04 11:48:37	2020-06-04 11:48:37	*NULL*	*NULL*
10	紅茶を買う	2020-06-12	2020-06-04 11:49:20	2020-06-04 11:49:20	10	1
15	関数を動かす	2020-06-05	2020-06-05 10:54:39	2020-06-05 10:54:39	15	1
16	sessionを使う	2020-06-09	2020-06-09 10:48:12	2020-06-09 10:48:12	16	1

「php02_table」と「集計結果」が結合される！！

結合前

- todo_tableとGROUP BYしたテーブル

	id	todo	deadline	created_at	updated_at
編集 コピー 削除	4	良い食材を買う	2020-06-19	2020-06-04 11:47:54	2020-06-05 12:01:56
編集 コピー 削除	7	PHP選手権準備	2020-06-07	2020-06-04 11:48:37	2020-06-04 11:48:37
編集 コピー 削除	10	紅茶を買う	2020-06-12	2020-06-04 11:49:20	2020-06-04 11:49:20
編集 コピー 削除	15	関数を動かす	2020-06-05	2020-06-05 10:54:39	2020-06-05 10:54:39
編集 コピー 削除	16	sessionを使う	2020-06-09	2020-06-09 10:48:12	2020-06-09 10:48:12

todo_id	cnt
10	1
15	1
16	1

【解説】JOIN

- JOINを使うとテーブルの結合ができる！
- 例
  - SELECT * FROM todo_table LEFT OUTER JOIN result_table
    ON todo_table.id = result_table.todo_id
  - 「todo_table」と「result_table」を結合する
  - 「ON」の後でどのカラムを対応させるのかを決定する
  - 今回は「todo_tableのid」と「result_tableのtodo_id」が対応

【解説】OUTER JOINとINNER JOIN

- OUTER JOINは対応するものがない場合NULLで補完される
- 例（上で実行したものと同じ）
  - SELECT * FROM todo_table LEFT OUTER JOIN result_table
    ON todo_table.id = result_table.todo_id

id	todo	deadline	created_at	updated_at	todo_id	cnt
4	良い食材を買う	2020-06-19	2020-06-04 11:47:54	2020-06-05 12:01:56	*NULL*	*NULL*
7	PHP選手権準備	2020-06-07	2020-06-04 11:48:37	2020-06-04 11:48:37	*NULL*	*NULL*
10	紅茶を買う	2020-06-12	2020-06-04 11:49:20	2020-06-04 11:49:20	10	1
15	関数を動かす	2020-06-05	2020-06-05 10:54:39	2020-06-05 10:54:39	15	1
16	sessionを使う	2020-06-09	2020-06-09 10:48:12	2020-06-09 10:48:12	16	1

【解説】OUTER JOINとINNER JOIN

- INNER JOINは対応するものがない場合は削除される
- 例
  - SELECT * FROM todo_table INNER JOIN result_table
    ON todo_table.id = result_table.todo_id

id	todo	deadline	created_at	updated_at	todo_id	cnt
10	紅茶を買う	2020-06-12	2020-06-04 11:49:20	2020-06-04 11:49:20	10	1
15	関数を動かす	2020-06-05	2020-06-05 10:54:39	2020-06-05 10:54:39	15	1
16	sessionを使う	2020-06-09	2020-06-09 10:48:12	2020-06-09 10:48:12	16	1

あとは表示するだけ！！

todo_read.php

```
// いいねボタン
<a href='like_create.php?user_id={$user_id}&todo_id={$record["id"]}'>
  like{$record["cnt"]}
</a>
```

テーブルを結合したので,
カラム名「cnt」を指定していいね件数を取れる！！

いいねボタンの表示確認

- クリックすると1増える or 1減る

課題

DB連携アプリ（登録処理 / 表示処理 / 更新処理 / 削除処理 / 集計 / 結合）

- DBを使用したアプリを実装しよう！

  DB名：　　gsacfd06_受講番号（←授業で作成したものでOK）

  テーブル名：　　自由に！

- 卒制のプロトタイプ的な作品とか，SNS的なものとか！

提出は次回授業前木曜「23:59:59」まで！！

P2Pタイム

まずはチーム内で解決を目指す！
訊かれた人は苦し紛れでも応える！！