BBS　掲示板　仕様書

24G1001 相京　航汰

2024 年 1 月 7 日

1　利用者向け

画面のレイアウト

- **掲示板ページ (bbs.html)**

- ページのタイトル: **掲示板**
- ページ構成:
  - **投稿フォーム:**
    - 名前を入力するテキストボックス
    - メッセージを入力するテキストボックス
    - 投稿ボタン
  - **投稿一覧エリア:**
    - 投稿内容が表示される.
    - 各投稿には以下の要素が表示される:
      - 名前
      - メッセージ
      - いいねボタン
      - 編集ボタン
      - 削除ボタン
  - **投稿チェックボタン:** 投稿の更新状況を確認するボタン

ボタンの動作

1. **送信ボタン**
   - 名前とメッセージを入力して押すと，その内容が新しい投稿として追加される.
   - 投稿後は入力欄がクリアされ，投稿一覧が更新される.
2. **投稿チェックボタン**
   - サーバーに現在の投稿数を問い合わせ，更新があれば新しい投稿を一覧に追加する.
3. **いいねボタン**
   - ボタンを押すと，その投稿に「いいね」が 1 つ追加される.
   - クリック後，ボタンに表示される「いいね数」が更新される.
4. **編集ボタン**
   - ボタンを押すとポップアップが表示され，メッセージの編集が可能になる.
   - 編集後，サーバーに変更が送信され，投稿が更新される.

5. **削除ボタン**
   - ボタンを押すとその投稿が削除される.
   - 削除後，画面から投稿が消える.

2　管理者向け

サーバーの起動手順

1. **前提条件**
   - Node.js および npm がインストールされていること.
   - 必要なモジュールがインストールされていること.
2. **サーバーの起動**

   ```
   node app8.js
   ```

   - サーバーはデフォルトでポート `8080` で動作する.
3. **サーバーの停止**
   - サーバーを停止するには、起動中のターミナルで `Ctrl + C` を押す.
4. **ブラウザでの確認**

   ```
   http://localhost:8080/public/bbs.html
   ```

ファイル構成

- **app8.js:** サーバーサイドのコード.
- **bbs.html:** クライアントの HTML.
- **bbs.js:** クライアントの JavaScript.
- **bbs.css:** クライアントのスタイルシート.

3　開発者向け

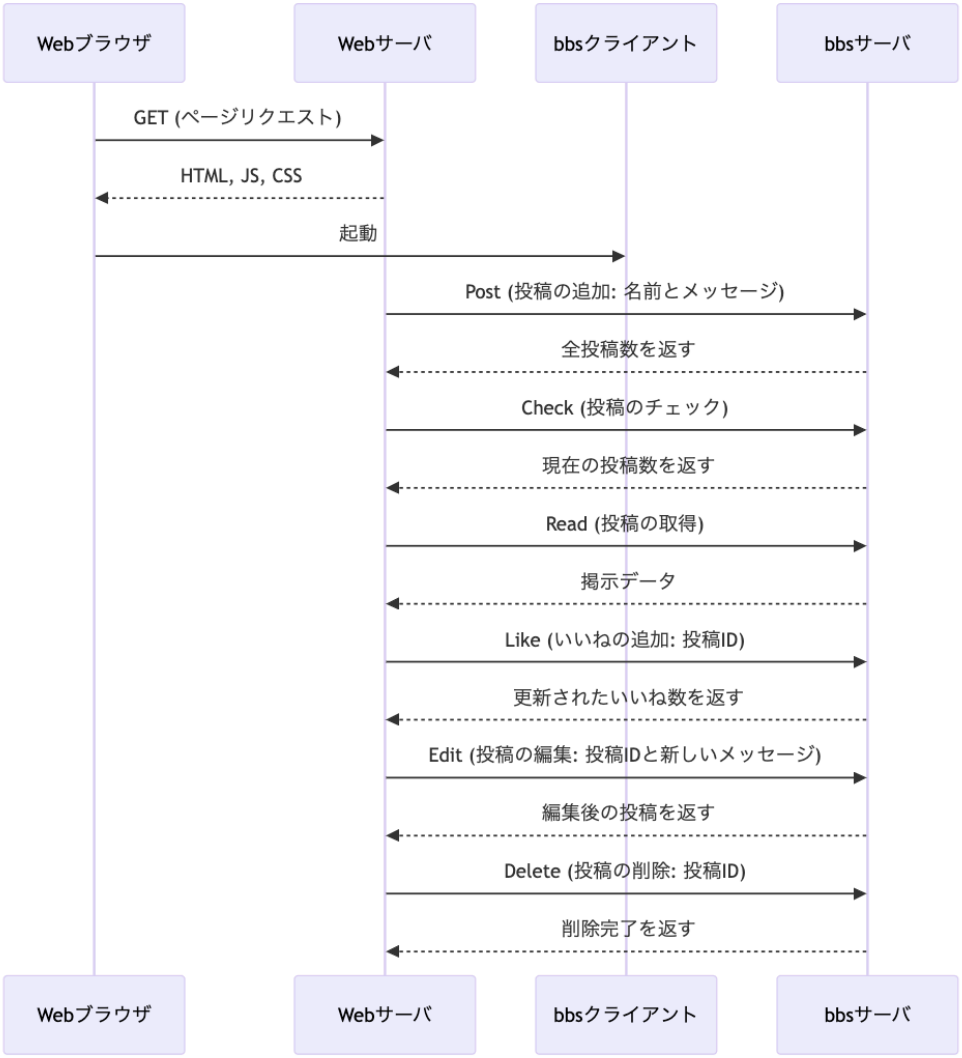

図 1　掲示板システムの全体の流れ

プログラム概要

サーバーサイド (app8.js)

- **フレームワーク:** - **主なエンドポイント:**

HTTP メソッド	URL	説明
POST	/post	新しい投稿を追加する
POST	/check	投稿数を確認する
POST	/read	新しい投稿を取得する
POST	/like	投稿にいいねを追加する
POST	/edit	投稿を編集する
POST	/delete	投稿を削除する

- **データ形式:**

- リクエストは `application/x-www-form-urlencoded` 形式.
- レスポンスは JSON 形式.

application/x-www-form-urlencoded 形式とは

application/x-www-form-urlencoded は，Web フォームから送信されるデータのエンコーディング形式の一つである．この形式は，HTTP リクエストのボディ部分でデータを送信する際に使用され，特に GET または POST メソッドでよく利用される．例えば，ウェブフォームを使ってユーザーが入力した情報をサーバーに送信する際に，この形式でデータが送られる．採用理由は，簡単で広くサポートされているからであり，この形式は，HTTP リクエストの標準的なデータ形式として広く認識されており，ほとんどのサーバーおよびクライアントライブラリでサポートされている．Web フォーム（例えば，HTML フォーム）から送信されるデータは，この形式でエンコードされるのが一般的である．

主な関数

- **投稿の追加 (/post)**
  - リクエストボディ例:

    ```
    name=ユーザー名&message=投稿内容
    ```

  - レスポンス例:

    ```
    { "number": 3 }
    ```

- **投稿のチェック (/check)**
  - リクエストボディ: 空
  - レスポンス例:

    ```
    { "number": 3 }
    ```

- **投稿の取得 (/read)**
  - リクエストボディ例:

    ```
    start=0
    ```

  - レスポンス例:

    ```
    {
      "messages": [
        { "id": 1, "name": "Alice", "message": "Hello!", "likes": 2 },
        { "id": 2, "name": "Bob", "message": "Hi there!", "likes": 1 }
      ]
    }
    ```

- **いいねの追加 (/like)**
  - リクエストボディ例:

    ```
    id=1
    ```

  - レスポンス例:

```
{ "likes": 3 }
```

- **投稿の編集 (/edit)**
  - リクエストボディ例:

```
id=1&message=新しいメッセージ
```

  - レスポンス例:

```
{
  "updatedPost": { "id": 1, "name": "Alice", "message": "新しいメッセージ", "likes": 2 }
}
```

- **投稿の削除 (/delete)**
  - リクエストボディ例:

```
id=1
```

  - レスポンス例:

```
{ "message": "Post deleted" }
```

クライアントサイド

- **bbs.js:**
  - 各ボタンに対応するイベントリスナーを定義.
  - サーバーと通信し，投稿データを更新・取得.
- **bbs.css:**
  - 投稿エリアのレイアウトをスタイル定義.

4 ソースコード

https://github.com/Kooxxx/webpro_06.git