

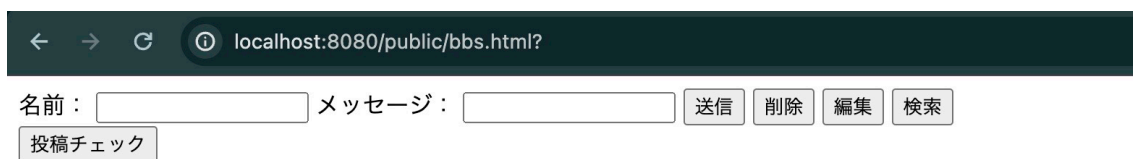
アプリケーション仕様書

2025 年 1 月 7 日

1. 利用者向け仕様

1.1 機能概要

このアプリは掲示板機能を提供し，利用者は以下の操作を行うことが可能．まずはじめに「<http://localhost:8080/public/bbs.html?>」この URL のサイトを開く．実際の画面は図 1 のようになり，表示されるボタンの種類と説明を表 1 に示す．



The screenshot shows a web browser window with the address bar displaying 'localhost:8080/public/bbs.html?'. Below the address bar, there are two input fields: '名前:' (Name) and 'メッセージ:' (Message). To the right of these fields are five buttons: '送信' (Send), '削除' (Delete), '編集' (Edit), '検索' (Search), and '投稿チェック' (Post Check).

図 1 実際に Web ページを開いた時の画面

表 1: 掲示板の操作ボタンとその機能の説明

ボタン	説明
送信	入力した名前とメッセージを掲示板に投稿する
削除	投稿番号を指定して、該当する投稿を削除する
編集	投稿番号と新しいメッセージを指定し、投稿を編集する
検索	キーワードを入力し、名前またはメッセージに一致する投稿を検索する
投稿チェック	全ての投稿を最新の状態で表示する

1.2 操作手順

- 名前とメッセージを入力し，「送信」ボタンを押すと新しい投稿が追加される．
- 「削除」ボタンを押すと，削除したい投稿の番号を入力するダイアログが表示され，指定した番号の投稿が削除される．

3. 「編集」ボタンを押すと、編集したい投稿の番号と新しいメッセージを入力するダイアログが表示され、指定した投稿が更新される。
4. 「検索」ボタンを押すと、キーワードを入力するダイアログが表示され、結果がアラートで通知される。
5. 「投稿チェック」ボタンを押すと、すべての投稿が更新表示される。

2. 管理者向け仕様

2.1 サーバーの起動手順

1. 任意の場所にディレクトリを作成し、その中に必要なファイル (app8.js, bbs.js, bbs.html, bbs.css) をダウンロードし、配置する
2. ターミナルを開き `node app8.js` を実行する
3. 手順 2 により、サーバーが `http://localhost:8080` で起動し、リクエストの受け付けを開始する
4. Web ページを開き、`http://localhost:8080/public/bbs.html?` のサイトにアクセスする

2.2 各ソースコードの管理

- `app8.js`: サーバー側のロジックを担っており、各種 API のエンドポイント (投稿, 削除, 編集, 検索) を定義している。
- `bbs.js`: クライアント側でユーザーの操作を処理しており、`fetch` API を用いてサーバーと通信をしている。
- `bbs.html`: ユーザーインターフェースを定義している。
- `bbs.css`: 表示レイアウトとスタイルを管理している。

3. 開発者向け仕様

3.1 構成ファイル

- `app8.js`: サーバーサイド
- `bbs.js`: クライアントサイドの操作スクリプト
- `bbs.html`: ユーザーインターフェース
- `bbs.css`: レイアウトとスタイルシート

3.2 サーバールートと説明

メソッド	ルート	説明
POST	/post	新しい投稿を追加する。パラメータ: は <code>name</code> と <code>message</code>
POST	/check	投稿数を返す。

POST	/read	すべての投稿を取得する。パラメータは <code>start</code>
POST	/delete	指定した投稿を削除する。パラメータは <code>index</code>
POST	/edit	指定した投稿を編集する。パラメータは <code>index</code> , と <code>message</code>
POST	/search	キーワード検索を行う。パラメータは <code>keyword</code>
POST	/like	指定した投稿に「いいね」を追加する。パラメータは <code>index</code>

3.3 クライアントイベントと対応するサーバー処理

イベント	対応するサーバーエンドポイント
投稿送信	/post
投稿削除	/delete
投稿編集	/edit
投稿検索	/search
投稿チェック	/read

4. データ形式と例

4.1 通信フォーマット

- **Content-Type:** application/x-www-form-urlencoded

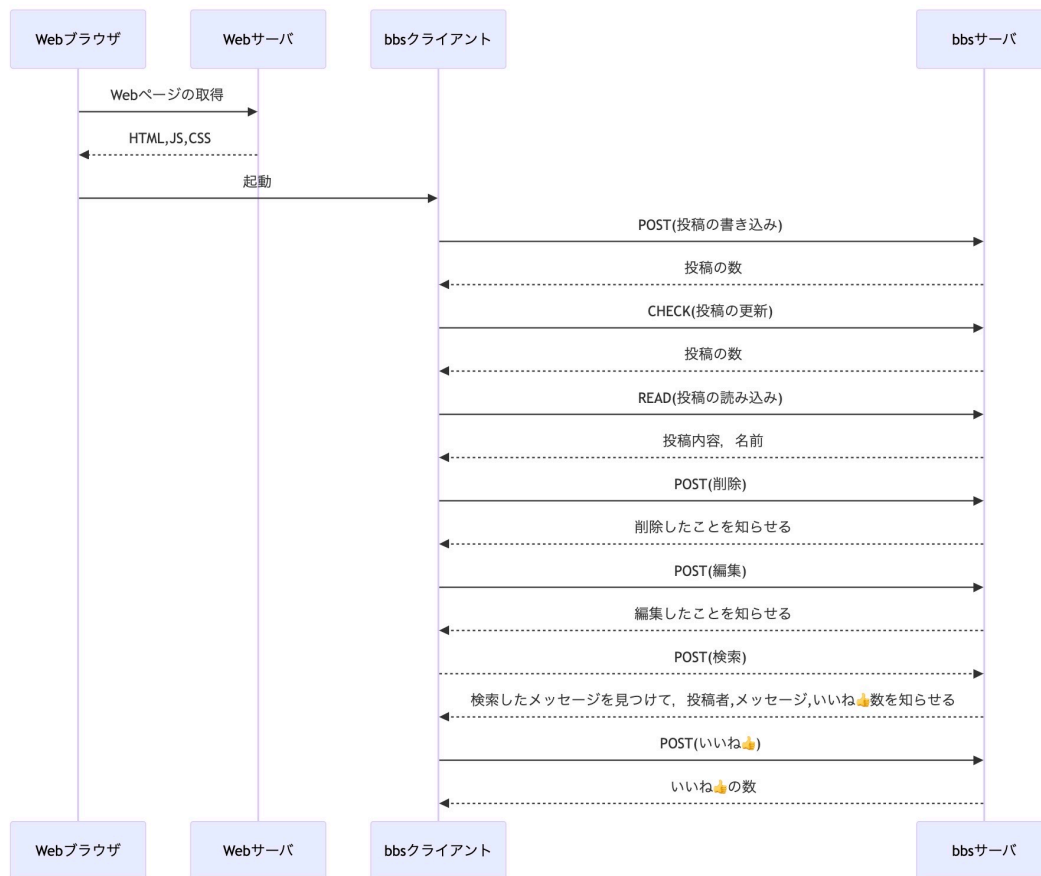


図2 Web ページの開始, 起動, bbs クライアントとサーバのやり取りの全体図

POST /post

リクエスト例

name=伊藤&message=こんにちは

レスポンス例

```
{ "number": 1 }
```

POST /read

リクエスト例

start=0

レスポンス例

```
{ "messages": [{ "name": "伊藤", "message": "こんにちは", "likes": 0 }] }
```

POST /delete

リクエスト例

index=0

レスポンス例

```
{ "status": "success", "message": "Deleted successfully" }
```

POST /like

リクエスト例

index=0

レスポンス例

```
{ "status": "success", "likes": 1 }
```