Web 掲示板システム仕様書

小川 泰生

2025 年 1 月 7 日

目次

1 はじめに

本レポートは、Web 掲示板システムについて、利用者向けガイド、管理者向けガイド、および開発者向けガイドの 3 つの視点から構成されています。以下に、それぞれのセクションの内容について説明します。

1.1 レポートの構成概要

1. 利用者向けガイド:
   - アプリケーションの目的や機能一覧を説明し、ユーザーがどのようにアプリケーションを操作するかを示す。タスクの追加・削除・完了状態の変更など、主要な機能についての操作方法を具体的に

説明する。

2. 管理者向けガイド:
   - サーバーセットアップやシステム要件について説明し、管理者がアプリケーションを正常に運用するために必要な手順を記載する。
3. 開発者向けガイド:
   - プロジェクトのディレクトリ構造、ファイル構成、使用した技術について説明する。
   - フロントエンドとバックエンド間の API 通信の仕様（エンドポイント、データ形式、リクエストとレスポンスの例など）を記載する。
   - 将来的な拡張案や改善点についても提案する。

2 利用者向けガイド

2.1 アプリケーションの目的

Web 掲示板システムは、ユーザーがメッセージを投稿、表示、ソート、削除できるよう設計されています。ユーザーが簡単に操作できるインターフェイスを提供し、コミュニケーションを円滑にします。

2.2 画面のレイアウトとボタンの仕様

掲示板システムの画面は以下の要素で構成されています。

- **ヘッダー**: システムのタイトルを表示します。
- **入力フィールド**: 名前とメッセージを入力するためのテキストフィールドです。
- **送信ボタン**: 入力されたメッセージを掲示板に追加します。
- **リセットボタン**: すべての投稿を削除します。
- **ソートボタン**: メッセージを名前順でソートします。
- **投稿表示ボタン**: すべての投稿を表示します。
- **文字数カウント**: メッセージの文字数をリアルタイムで表示します。

3 管理者向けガイド

3.1 サーバーのセットアップ

このアプリケーションは、ターミナルを使用して `app8.js` を起動することでサーバーが動作します。以下の手順に従って、サーバーをセットアップします。

1. `app8.js` を使用したサーバーの起動

   ```
   node app8.js
   ```

   このコマンドを実行すると、サーバーが起動し、アプリケーションが動作します。

2. サーバーの動作確認ブラウザで以下の URL にアクセスして、掲示板が正常に動作することを確認します。

```
http://localhost:3000
```

4 開発者向けガイド

4.1 内部的な作りと変数の意味

このアプリケーションの主な内部構造と変数の意味を以下に示します。

- **mainContainer**: メッセージリストを格納する要素。
- **nameInput**: 名前を入力するテキストフィールド。
- **messageInput**: メッセージを入力するテキストフィールド。
- **charCount**: メッセージの文字数をカウントする要素。
- **addMessage**: メッセージをリストに追加する関数。

4.2 通信内容

フロントエンドとバックエンドの通信は、以下のエンドポイントを使用して行われます。

- **GET /messages**: すべてのメッセージを取得。
- **POST /messages**: 新しいメッセージを追加。
- **DELETE /messages/:id**: 指定された ID のメッセージを削除。

4.2.1 リクエスト例

```
POST /messages
{
  "name": "ユーザー名",
  "message": "メッセージ内容"
}
```

4.2.2 レスポンス例

```
{
  "id": 1,
  "name": "ユーザー名",
  "message": "メッセージ内容",
  "timestamp": "2025-01-07T12:00:00Z"
}
```

4.3 データ形式の詳細

メッセージデータは JSON 形式でやり取りされます。以下に、主要なデータフィールドとその説明を示します。

- **id**: メッセージの一意の識別子（整数）。
- **name**: メッセージを投稿したユーザーの名前（文字列）。
- **message**: 投稿されたメッセージ内容（文字列）。
- **timestamp**: メッセージが投稿された日時（ISO 8601 形式の文字列）。

4.4 通信フロー

メッセージの送信と取得の基本的なフローは以下の通りです。

1. ユーザーがメッセージを入力し、送信ボタンをクリックすると、フロントエンドからバックエンドに POST リクエストが送信されます。
2. バックエンドはリクエストを受け取り、新しいメッセージをデータベースに保存し、保存されたメッセージのデータをレスポンスとして返します。
3. フロントエンドはレスポンスを受け取り、新しいメッセージを画面に表示します。

5 GitHub リポジトリ

このアプリケーションのソースコードは GitHub でホストされています。以下の URL からアクセスできます：`https://github.com/Ogitai/SPA.git`