

生成AIパスポート試験対策アプリ

このWebアプリケーションは、生成AIパスポート試験の学習を支援するものです。



試験問題の演習

本番さながらの模擬試験で、実践的な演習を繰り返すことができます。



過去の結果を確認

これまでの学習履歴や結果をグラフで確認し、弱点を把握できます。



学習コンテンツの閲覧

試験範囲をカバーする豊富な学習コンテンツにいつでもアクセス可能です。

機能概要



問題を解く

選択した問題数で試験同様の形式で挑戦できます。解答後には正解と解説が表示されます。



各問題一覧

章、セクション、サブセクション、学習項目ごとに問題を閲覧し、解答と解説を確認できます。



学習モード

各章、セクション、学習項目に対応する解説コンテンツを閲覧できます。



過去の結果

過去に実施した問題の結果履歴（正解数、正答率）を確認できます。

ファイル構成

本アプリケーションは以下のファイルで構成されています。

index.html	アプリケーションのメインとなるHTMLファイルです。各画面の構造を定義しています。
styles.css	アプリケーション全体のスタイル (CSS) を定義しています。
main.js	アプリケーションの初期化、画面遷移、共通処理を管理しています。
quiz.js	クイズ機能 (問題の表示、解答判定、タイマーなど) を管理しています。
questions.js	問題データが格納されています。このファイルを編集して問題を追加・更新します。
textbookContent.js	学習コンテンツのデータが格納されています。このファイルを編集して学習コンテンツを追加・更新します。
pastResults.js	過去の問題の結果の保存と表示に関する処理を管理しています。
questionList.js	各問題一覧画面の表示とナビゲーション (パンくずリスト含む) を管理しています。
utils.js	共通で利用されるユーティリティ関数 (配列のシャッフル、画面表示制御など) を提供します。

問題の追加・編集方法

問題はすべて `questions.js` ファイルで管理されています。新しい問題を追加したり、既存の問題を編集したりする場合は、このファイルを直接編集します。

questions.js ファイルの概要

`questions.js` ファイルには、`allQuestions` という名前のJavaScript配列が定義されています。この配列の各要素が1つの試験問題に対応します。

```
// questions.js
const allQuestions = [
  // 問題オブジェクト 1
  { ... },
  // 問題オブジェクト 2
  { ... },
  // ...
];
```

問題データの構造

各問題オブジェクトは以下のプロパティを持つ必要があります。

<code>id</code>	<code>string</code>	問題の一意な識別子です。必ずユニークな文字列または数字（例: "1", "2", "101" など）にしてください。既存の最大IDを確認し、それよりも大きな値を使用することをおすすめします。
<code>chapter</code>	<code>string</code>	問題が属する「章」のタイトルです。例: "第一章"。これは、学習モードで表示される章タイトルと一致させることで、問題と学習コンテンツの連携が向上します。
<code>section</code>	<code>string</code>	問題が属する「セクション（中項目）」のタイトルです。例: "AI（人工知能）の定義"。
<code>subsection</code>	<code>string</code>	問題が属する「サブセクション（小項目）」のタイトルです。例: "AIとは"。もしサブセクションがない場合は、 <code>section</code> と同じ値にしても問題ありません。
<code>learningItem</code>	<code>string</code>	問題が関連する「学習項目」のタイトルです。これは、各問題一覧画面で特定の学習項目に絞り込んで問題を表示する際に利用されます。 <code>textbookContent.js</code> の <code>learningItems</code> 内の <code>title</code> と一致させることを推奨します。
<code>Quiz</code>	<code>string</code>	問題文です。
<code>Answer</code>	<code>string</code>	正解の選択肢の番号です。文字列で指定してください（例: "1", "2", "3", "4"）。
<code>Choice</code>	<code>Array</code>	選択肢の配列です。各要素は必ず「数字 選択肢のテキスト」の形式で記述してください（例: "1 選択肢1のテキスト", "2 選択肢2のテキスト"）。これにより、アプリケーションが自動的に選択肢番号とテキストを分離して表示します。
<code>Description</code>	<code>string</code>	問題の解説文です。なぜその選択肢が正解なのか、不正解の選択肢がなぜ間違いなのかを説明します。

問題の追加方法

既存の章やセクションに問題を追加する

最も一般的な問題追加の方法です。

1. questions.js ファイルをテキストエディタで開きます。
2. const allQuestions = [で始まる配列の末尾に移動します。
3. 新しい問題オブジェクトを、既存の最後の問題オブジェクトの後にカンマ (,) で区切って追加します。
4. 重要: 最後の問題オブジェクトの後にカンマを忘れないでください。
5. 追加する問題の chapter, section, subsection, learningItem には、既存の適切な値を正確にコピー & ペーストしてください。これにより、アプリ内で問題が正しい階層に分類されます。
6. id は、既存のすべての問題IDと重複しない、ユニークな値を設定してください。現在の最後の問題のIDが"453"であれば、"454"など、連続した番号を振るのが最も安全です。

```
// ... 既存の問題 (questions.js 内) ...

{
  "id": "453",
  "chapter": "第五章",
  "section": "テキスト生成AIのプロンプト制作と実例",
  "subsection": "テキスト生成AIの不得意なこと",
  "learningItem": "芸術の批評",
  "Quiz": "テキスト生成AIが芸術批評において最も得意とすることは何か？",
  "Answer": "2",
  "Choice": [
    "1 人間のような深い感動を伴う詩的な批評を生成すること",
    "2 既存の芸術作品に関する客観的な情報や歴史的背景をまとめること",
    "3 リアルタイムで変化する芸術市場のトレンドを予測すること",
    "4 批評対象の芸術作品を独自に創造すること"
  ],
  "Description": "AIは、学習データに基づいた客観的な情報整理や要約は得意ですが、人間のような創造性や深い感情を伴う批評は苦手です。"
}, // <-- ここにカンマが必要です！

{
  "id": "454", // 新しい問題のユニークなID
  "chapter": "第五章",
  "section": "テキスト生成AIのプロンプト制作と実例",
  "subsection": "テキスト生成AIの不得意なこと",
  "learningItem": "芸術の批評", // 既存の学習項目名を正確に指定
  "Quiz": "テキスト生成AIが生成する芸術批評において、倫理的な懸念が生じる可能性が最も高いのはどのような場合か？",
  "Answer": "3",
  "Choice": [
    "1 事実に基づいた客観的な情報を提供する際",
    "2 複数の批評家の意見を要約する際",
    "3 特定の芸術家や作品に対してAIが偏見に基づいた評価を下す際",
    "4 一般的な芸術史の解説を行う際"
  ],
  "Description": "AIが学習データに含まれる偏見を反映し、特定の芸術家や作品に対して差別的な評価を下す可能性があるため、倫理的な懸念が生じます。"
}
]; // allQuestions配列の閉じ括弧
```

追加時の注意点

- id のユニーク性: 最も重要な点です。既存のIDと重複すると、アプリの動作に予期せぬ問題が発生する可能性があります。
- JSON形式の厳密性: JavaScriptのオブジェクトリテラルですが、実質的にはJSONに近いため、以下の点に注意してください。
 - すべてのキーと文字列値はダブルクォーテーション ("") で囲んでください。
 - 各プロパティの区切りはカンマ (,) です。最後のプロパティの後にカンマは不要ですが、次の要素を追加する場合は必要になります。
 - [] や {} の閉じ忘れがないか確認してください。
- Choice のフォーマット: 各選択肢は「数字 選択肢テキスト」の厳密なフォーマット（例: "1 これは選択肢のテキストです"）を維持してください。このフォーマットでないと、選択肢の表示や解答判定が正しく行われません。

学習コンテンツの追加・編集方法

学習コンテンツは textbookContent.js ファイルで管理されています。新しいコンテンツを追加したり、既存のコンテンツを編集したりする場合は、このファイルを直接編集します。

textbookContent.js ファイルの概要

textbookContent.js ファイルには、textbookContent という名前のJavaScriptオブジェクトが定義されており、章、セクション、学習項目の階層構造でコンテンツを管理しています。

```
// textbookContent.js
const textbookContent = {
  "章のキー": {
    "name": "章の表示名",
    "sections": {
      "セクションのキー": {
        "name": "セクションの表示名",
        "learningGoal": "学習目標",
        "learningItems": [
          {
            "title": "学習項目のタイトル",
            "paragraphs": ["段落1", "段落2"]
          },
          // ...
        ]
      }
    }
  }
};
```

学習コンテンツの構造

章とセクションの構造

- 章のキー、セクションのキー: これらはJavaScriptオブジェクトのプロパティ名（キー）として使用されます。ユニークな文字列である必要があります（例: "第一章", "AI（人工知能）の定義"）。
- name: 各章やセクションの表示名です。
- learningGoal: セクションごとの学習目標です。

学習项目的構造

- learningItems: そのセクションに含まれる学習项目的配列です。
- title: 学習项目的タイトルです。questions.js の learningItem と一致させることで、問題と学習コンテンツの連携がスムーズになります。
- paragraphs: その学習项目的内容を構成する段落の配列です。各要素が1つの段落となり、HTMLでタグとして表示されます。

既存の学習項目にコンテンツを追加する

- textbookContent.js をエディタで開きます。
- 該当する章、セクション、学習項目を見つけます。
- paragraphs 配列に新しい段落のテキストを文字列として追加します。

```
// ... 既存のコンテンツ ...

"AIとは": {
  "title": "AIとは",
  "paragraphs": [
    "AI（人工知能）とは、人間の知的な能力をコンピュータ上で再現しようとする技術や研究分野のことです。具体的には、学習、推論、問題解決、知覚、言語理解などの能力をコンピュータに持たせることを目指しています。これにより、コンピュータが自律的に状況を判断し、適切な行動を選択できるようになります。",
    "AIは Artificial Intelligence の略であり、その目的は人間の知能を機械で模倣し、様々な多問題を解決することにあります。例えば、音声認識、画像認識、自然言語処理、ゲームのプレイ、意思決定支援など、多岐にわたる分野で応用されています。",
    "新たにこの段落を追加します。AIの最新動向や具体的な応用例について記述できます。" // 新しい段落
  ],
}

// ... 既存のコンテンツ ...
```

開発者向け情報とスタイル編集

新しい章、セクション、学習項目を追加する

新しい分類を作成してコンテンツを追加する場合は、textbookContent.js の階層構造に従って新しいオブジェクトを追加します。

```
const textbookContent = {
  // ... 既存の章 ...

  "第六章": { // 新しい章のキー
    "name": "第六章 AIの倫理と社会", // 表示名
    "sections": {
      "AI倫理の基本": { // 新しいセクションのキー
        "name": "AI倫理の基本", // 表示名
        "learningGoal": "AI倫理の重要性と基本的な原則を理解する",
        "learningItems": [
          {
            "title": "AIの公平性", // 新しい学習項目のタイトル
            "paragraphs": [
              "AIシステムにおける公平性とは、差別的な結果や偏見を避けることを意味します。",
              "これは、学習データに存在する偏りや、アルゴリズム設計上の問題に起因することがあります。"
            ]
          }
        ]
      }
    }
  };
};
```

スタイル (CSS) の編集

アプリケーションの見た目を変更したい場合は、styles.css を編集してください。

- body や h1 などの全体的なスタイル。
- .screen や .menu-button などの各要素のスタイル。
- 問題結果表示時の .correct-answer や .user-answer などの色。
- レスポンシブデザインのための @media クエリ。

など、直感的に編集しやすいようにクラス名が付けられています。

開発者向け情報

- utils.js: shuffleArray (フィッシャー-イエーツシャッフル) や showScreen といった汎用的な関数が定義されています。
- quiz.js: 問題を解くロジックが実装されています。QUIZ_TIME_LIMIT や MAX_QUESTIONS といった定数もここで定義されています。
- questionList.js: 問題一覧表示のロジックとパンくずリストの管理が含まれます。
- pastResults.js: ローカルストレージを使用した結果の永続化と表示ロジックが含まれます。

これらのファイルを参照することで、アプリケーションの動作原理をより深く理解し、さらに高度なカスタマイズを行うことが可能です。