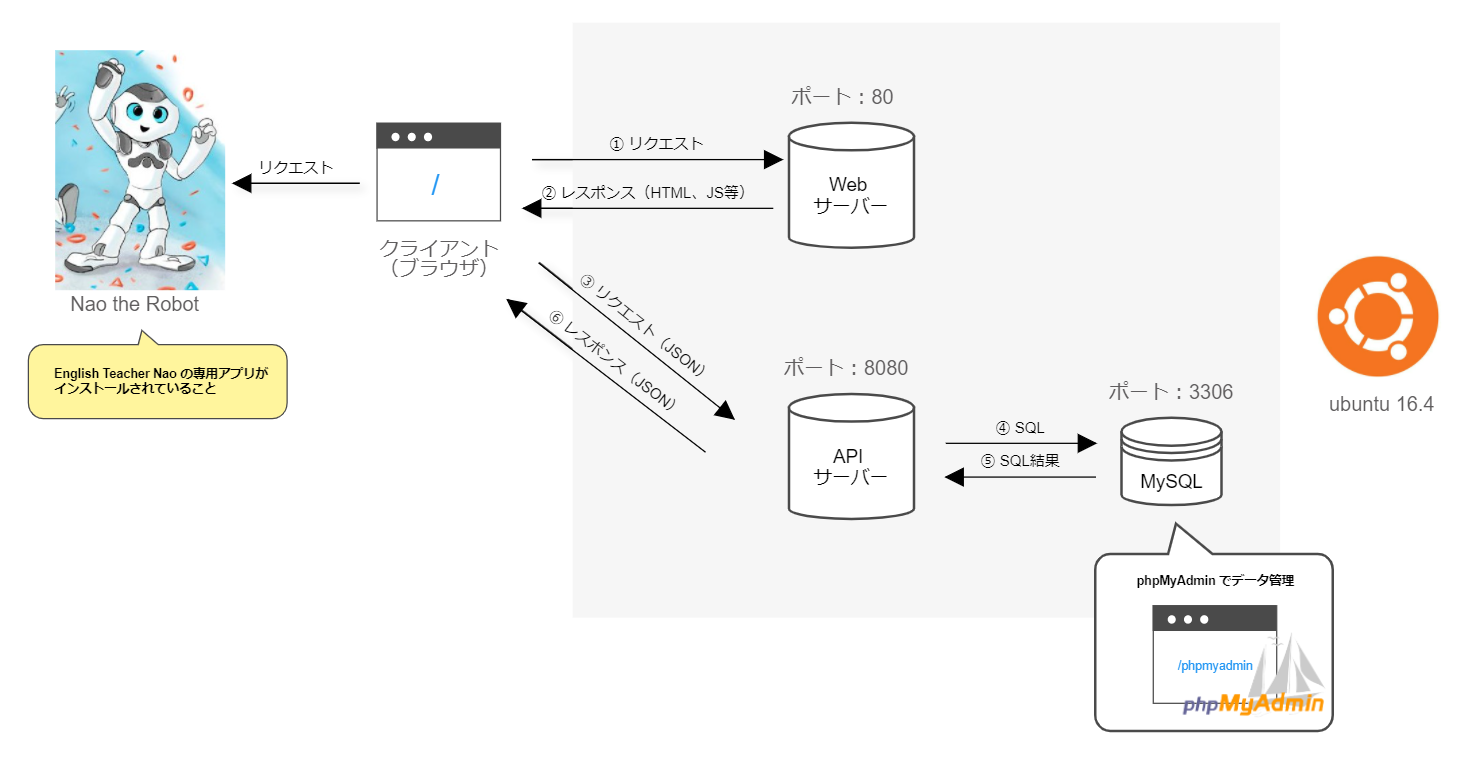
開発者ガイド（開発編）

## 構成イメージ

アプリケーションの全体的なアーキテクチャを示します。



画像[拡大](https://drive.google.com/file/d/1Hk2TqpIkgR3bWsn_MGxckEEafcsp2nvc/view?usp=sharing)

English Teacher NAO における、ボタンの数・色、コンテンツの数・タイトル、単語などの情報は、データベース（MySQL）で管理しています。データの変更に応じ、APIサーバーを介してブラウザに変更を反映する仕組みをとっています。

単語以外のデータをデータベースで管理する理由は、開発者でなくても UI やコンテンツを容易に変更できるようにするためです。phpMyAdmin などのクライアントツールを利用して仕様を変更が行えるよう実装しています。

## Webサーバー

　VPS（仮想専用サーバー）またはクラウド上の仮想サーバー（Amazon EC2 など）を用意し、Apache などのWebサーバー環境を構築してください。ドメインを取得し、**通信は http で行ってください。https での通信はロボットとの接続の問題で使用しないでください。**

## データベース（MySQL）

English Teacher NAO のデータベースは MySQL を使用しています。アプリを新しい環境で構築する場合は、Webサーバーを構築した環境に MySQL をインストールしてください。データベースの構成は以下に示します。

##### データベース名

　english\_teacher

##### テーブル定義

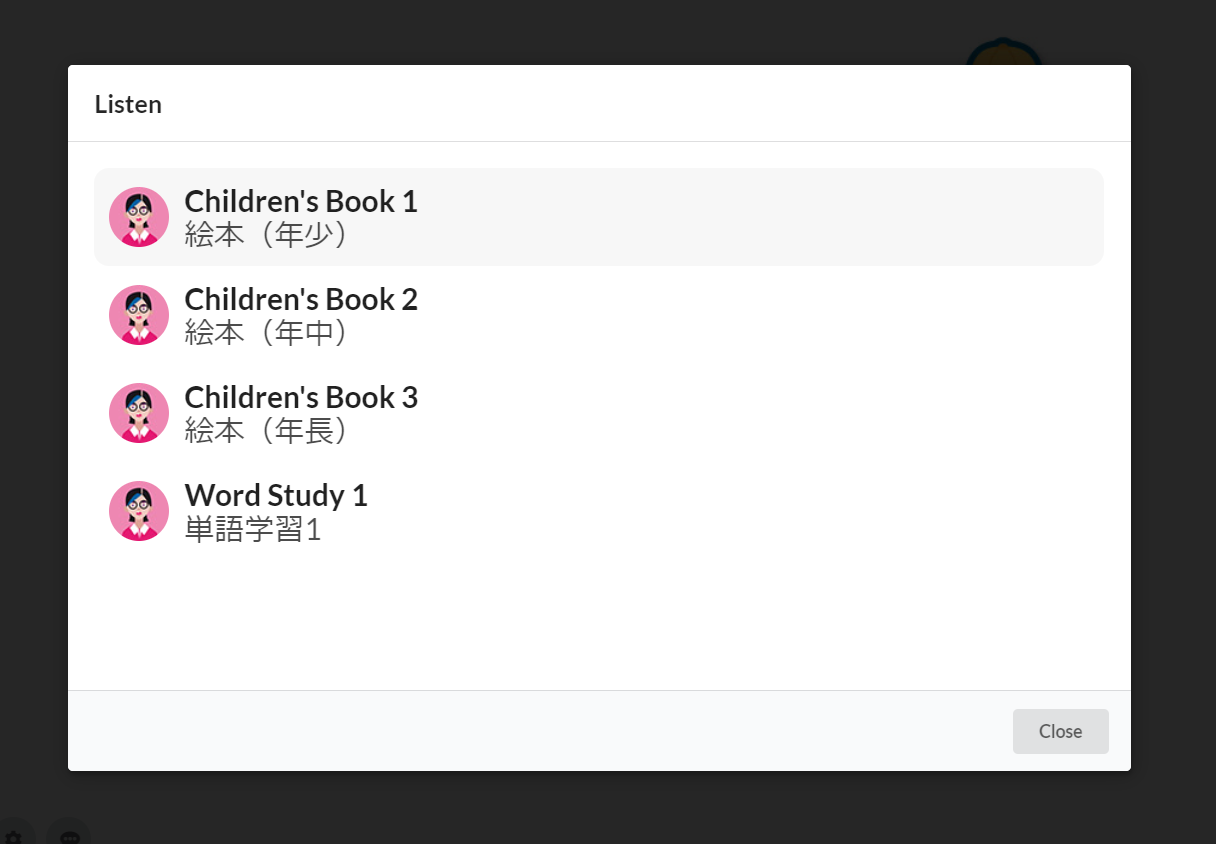
| テーブル名: category | | |
| --- | --- | --- |
| カラム名 | データ型 | コメント |
| id | int(10) | 主キー |
| name\_en | varchar(30) | ボタンに表示させたい英語文字（Listen や Talk など） |
| name\_ja | varchar(30) | ボタンに表示させたい日本語文字（聞く や 話すなど） |
| image | varchar(500) | ※ 使用していません |
| color\_id | int(10) | color テーブルの主キー（id）に紐く |
| 備考:  category テーブルは、ホーム画面の各メニュー（Listen, Talk）について、表示する文字や色を決める情報がおさめられます。図1 を参照してください。 | | |

**図 1　メニュー表示**



| テーブル名: subcategory | | |
| --- | --- | --- |
| カラム名 | データ型 | コメント |
| id | int(10) | 主キー |
| name\_en | varchar(50) | リストに表示させたい英語文字（Word Study 1 など） |
| name\_ja | varchar(50) | リストに表示させたい日本語文字（単語学習1 など） |
| image | varchar(500) | アイコン画像 |
| category\_id | int(10) | category テーブルの主キー（id）に紐く |
| 備考:  subcategory テーブルは、各メニュー（Listen, Talk, ...）の中のコンテンツについて、表示する文字やアイコンを決める情報がおさめられます。図2 を参照してください。 | | |

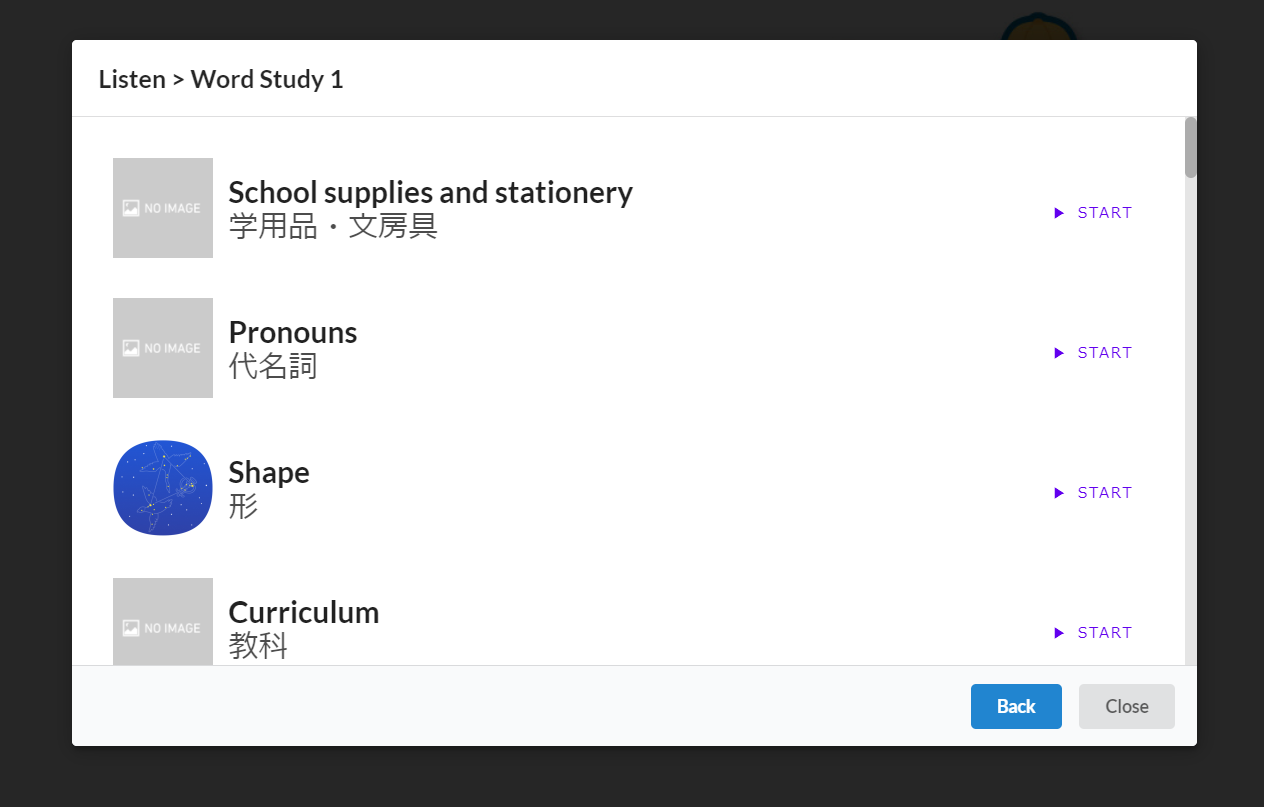
**図2　Listen のコンテンツ**



##### 

| テーブル名: apps | | |
| --- | --- | --- |
| カラム名 | データ型 | コメント |
| id | int(10) | 主キー |
| name\_en | varchar(100) | リストに表示させたいアプリの名前表示（英語） |
| name\_ja | varchar(100) | リストに表示させたいアプリの名前表示（日本語） |
| image | varchar(500) | アイコン画像 |
| uuid | varchar(500) | コレグラフアプリの uuid |
| display\_order | int(11) | リスト表示における表示順番 |
| number\_color\_id | int(10) | 単語学習における数字の色 |
| type\_id | int(10) | アプリの種類（app\_type テーブルに紐づく） |
| subcategory\_id | int(10) | subcategory テーブルの id に紐づく |
| category\_id | int(10) | category テーブルの id に紐づく |
| 備考:  apps テーブルは、各コンテンツに含まれるアプリについて、表示する文字やアイコンを決める情報がおさめられます。図3 を参照してください。 | | |

**図3　Listen の Word Study 1 のアプリ一覧**



##### 

| テーブル名: lesson | | |
| --- | --- | --- |
| カラム名 | データ型 | コメント |
| id | int(11) | 主キー |
| word | varchar(1000) | 単語文字 |
| app\_id | varchar(11) | apps テーブルの id に紐づきます |
| 備考: lesson テーブルは、単語学習における単語の情報がおさめられます。図4を参照してください。 | | |

**図4　lesson テーブルのデータ挿入例**



##### 

| テーブル名: color | | |
| --- | --- | --- |
| カラム名 | データ型 | コメント |
| id | int(11) | 主キー |
| color | varchar(50) | CSS の color プロパティの値（blue, #0bd など） |
| remark | varchar(100) | 備考を記入 |
| 備考: color テーブルは色の情報がおさめられます。category や apps テーブルから参照されます。 | | |

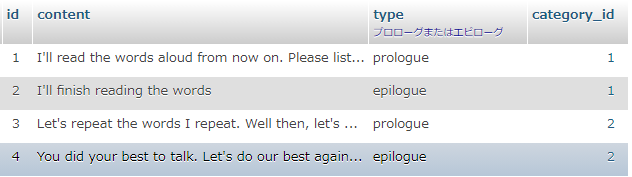
##### 

| テーブル名: app\_type | | |
| --- | --- | --- |
| カラム名 | データ型 | コメント |
| id | int(11) | 主キー |
| type | varchar(50) | アプリの種類（**※ 値は lesson または app**） |
| 備考: app\_type テーブルはアプリの種類情報がおさめられます。apps テーブルに参照されます。 | | |

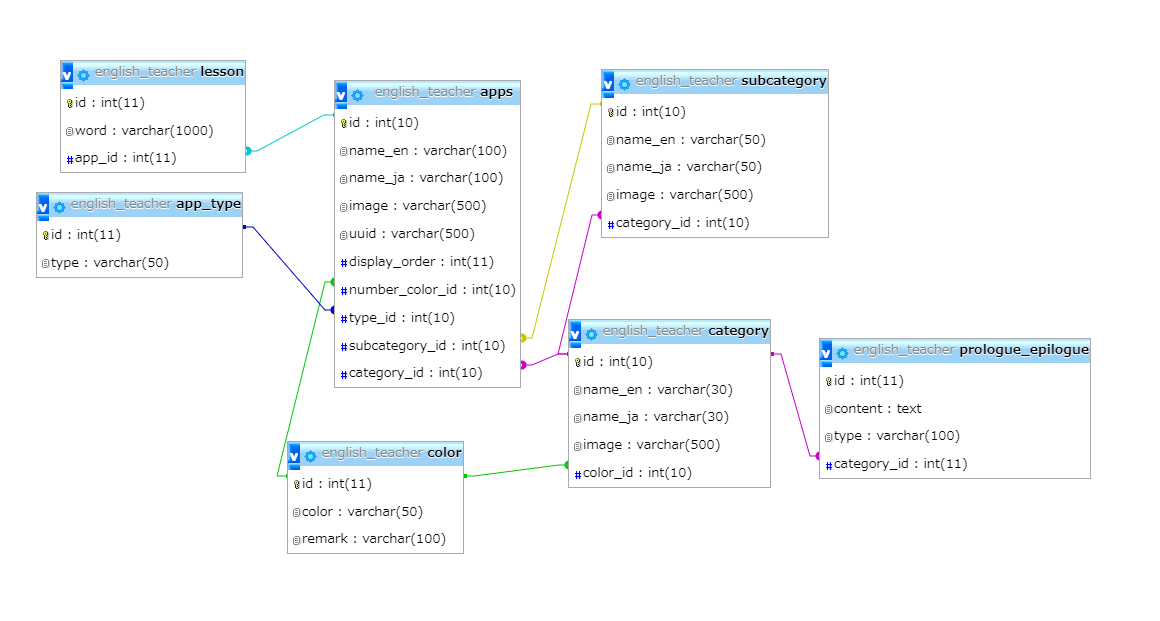
##### 

| テーブル名: prologue\_epilogue | | |
| --- | --- | --- |
| カラム名 | データ型 | コメント |
| id | int(11) | 主キー |
| content | text | 単語学習における開始または終了時のスピーチ |
| type | varchar(100) | 単語学習における開始時のスピーチか終了時のスピーチか (**※ 値はprologue または epilogue**) |
| category\_id | int(11) | category テーブルの id に紐づきます。 |
| 備考: prologue\_epilogue テーブルは、単語学習における開始または終了時のスピーチ情報がおさめられます。図5 を参照してください。 | | |

**図5　prologue\_epilogue データの挿入例**



##### ER図



[画像拡大](https://drive.google.com/file/d/1DdL3gJv1J9ok0tZUzmHGqFnUoPxV1Dji/view?usp=sharing)

## phpMyAdmin

　管理ツールの「phpMyAdmin」をインストールします。phpMyAdmin は、MySQLサーバーの操作をコマンドラインではなく、ブラウザを使用したWeb-UI で GUI として操作可能なアプリケーションです。インストールやDBの接続方法は、ネット上にたくさんの記事が載っているので割愛します。

## APIサーバー

　APIサーバーは、データベース接続やデータを取得しクライアントに渡す機能を担います。本ドキュメントの冒頭で載せた構成イメージを参照してください。厳密にはAPIサーバーの定義に当てはまりませんが、便宜上そう呼んでいます。

これからサーバーの構築手順を説明します。**以下の処理は、Webサーバーを構築した環境で行ってください**。

### 前提条件

* Git
* Vim
* Node.js v10.24.x

開発のソースコードをダウンロードします。ダウンロードする場所をホームディレクトリ（~）しましたが、任意で構いません。

| $ cd ~ $ git clone https://github.com/sd21y/english\_teacher\_api.git |
| --- |

server.js ファイルを開き、データベース接続のユーザー名とパスワードを環境に合わせて変更します。

| $ cd english\_teacher\_api $ vim server.js |
| --- |

| // server.js  ~略~  const connection = mysql.createConnection({  host: 'localhost',  user: '<mysql\_username>',  password: '<mysql\_password>',  database: 'english\_teacher' });  ~略~ |
| --- |

パッケージをインストールします。

| $ npm install |
| --- |

PM2 でアプリケーションを起動します。

| $ npx pm2 start server.js |
| --- |

## 

## フロントエンド環境

　ブラウザ上で動作するHTMLとJavaScript、CSSで構成されるプログラムやデータ一式を準備し、Webサーバー上に配置します。

### 前提条件

* Git
* Node.js v12.17.x

### ビルド

開発のソースコードをダウンロードします。ダウンロードする場所をホームディレクトリ（~）しましたが、任意で構いません。また、ロカール環境で実行しても問題ありません。

| $ cd ~ $ git clone https://github.com/sd21y/english\_teacher\_2.git |
| --- |

フォルダの中に移動します。

| $ cd ~/english\_teacher\_2 |
| --- |

続いて、パッケージをインストールします。

| $ npm install |
| --- |

src/js/app.js を開き、URL と公開ディレクトリ（Ubuntu の場合、/var/www/html ）からの PATH を設定します。/var/www/html 直下に公開ファイルを配置する場合は、PATH を空文字列（= "" ）にしてください。

| // src/js/app.js  ~略~  export const URL = "<URL>" // 例) http://robotics.sd21.co.jp export const PATH = "<PATH>" // 例) /nao/apps/english\_teacher  ~略~ |
| --- |

URLを変更できたら、ビルド（実行な可能なファイルの作成）を行います。ビルドが完了すると、ルートディレクトリに dist ディレクトリが作成されます。

| $ npm build |
| --- |

### デプロイ

###### ◆ 画像ファイルのコピー

　ページで利用する画像素材や単語学習の画像データをWebサーバーに配置します。画像データは以下のように入手します。

・GitHub からリポジトリをクローン

| $ git clone https://github.com/sd21y/english\_teacher\_data.git |
| --- |

ダウンロードしたディレクトリ（english\_teacher\_data）の image フォルダを、ビルド時にできた dist ディレクトリ直下に配置してください。

なお、image の中の lesson ディレクトリは単語学習用の画像データで中身が空になっています。単語に画像を合わせたい場合は、画像のファイル（jpg）の名前を単語と同じにして image/lesson の中に配置します。

###### ◆ ソースファイルのコピー

　ビルドしたファイルをWebサーバーに配置します。公開ディレクトリは、Ubuntu では、/var/www/html が用いられます。ビルドで作成した dist ディレクトリの中身をWebサーバーの公開ディレクトリに配置（コピー）します。**再デプロイするときは必ず、既にデプロイしたファイルをWebサーバーから削除してください。**

## コレグラフアプリのインストール

　Webアプリを通して実行する、ロボットの専用アプリ（コレグラフアプリ）をロボットにインストールします。

コレグラフアプリが入っているリポジトリをクローンしてください。sd21y の組織のメンバーのみクローンできます。

| $ git clone https://github.com/sd21y/english\_teacher\_data.git |
| --- |

クローンしたディレクトリの中の、english\_teacher\_apps がコレグラフのプロジェクトで、これをロボットにインストールします。

english\_teacher\_data

|\_ english\_teacher\_apps ← 使用するもの

|\_ ...

コレグラフを開き、コレグラフをロボットに接続させてください。続いて、ダウンロードしたアプリをコレグラフでプロジェクトとして開き、ロボットにインストールします。なお、コレグラフのバージョンは 2.8.6.23 を想定します。

◆ プロジェクトの開き方

ファイル → プロジェクトを開く → 対象の pml ファイルを選択 → オープン

◆ ロボットにアプリをインストールする方法

表示 → <ロボアプリ一覧> にチェックを入れる → 図６の赤枠で囲ったボタンをクリック

**図６　ロボットアプリケーションのインストール**

