**Scope of Work: Code Challenge – MyDictionary**

**Tech Stack:** Laravel (backend API), Vue.js (front-end SPA), TailwindCSS (styling)

**Authentication:** Laravel Sanctum (token-based)

**3rd-party API:** Free Dictionary API (https://dictionaryapi.dev/)

**Project Summary**

You are tasked with building a simple full-stack web application called MyDictionary, where users can:

* Register and log in securely
* Search for English words using the Free Dictionary API
* View definitions, pronunciation, part of speech, examples, and synonyms
* Save words to a personal favorites list
* Manage favorites (view, update notes, or remove saved words)

**Core Features**

1. **Authentication**

* Use Laravel Sanctum for email+password authentication
* Implement registration, login, logout
* Protect all user data behind authentication

1. **Word Search Integration**

* Create a backend endpoint that calls the Free Dictionary API

Example: GET /api/dictionary/search?q=patio

* Parse the response to return a clean and structured JSON format with:
  + Word
  + Phonetics / pronunciation
  + Definitions
  + Part of speech
  + Example usage
  + Synonyms

1. **Favorites Management**

* Create a model for “Favorite” words tied to each authenticated user
* API endpoints:
  + GET /api/favorites – List current user’s saved words
  + POST /api/favorites – Add a new favorite (must include word and definition data)
  + PUT /api/favorites/{id} – Update notes for a saved word
  + DELETE /api/favorites/{id} – Remove from favorites

1. **Vue.js Frontend (SPA)**

* Public Pages:
  + Login
  + Registration
* Protected Pages:
  + Search Page:
    - Input box for searching a word
    - Display results from API
    - “Save to Favorites” button
  + Favorites Page:
    - List of saved words
    - Ability to view full definition
    - Add/edit personal notes
    - Delete word from favorites

1. **Validation & UX**

* Client-side and server-side validation
* Basic form and API error handling
* Toast notifications for actions
* Loading indicators for API calls

1. **Scheduled Cleanup Task**

* Implement a Laravel scheduler that runs daily
* Automatically deletes any favorites older than 30 days
* Ensure proper logging or error handling for this background task