CS390-WAP

Tanay Gondil

AI Podcast Generator

Backend Development Assignment

1 Overview

In this assignment, you will build an **AI-powered podcast generator** that automatically creates audio podcasts from trending news. This command-line backend application fetches real data from the internet, processes it with AI to generate engaging scripts, and converts those scripts into professional-sounding audio files you can actually listen to.

1.1 What You'll Build

A Node.js application that:

1. Fetches trending news articles from the internet (NewsAPI)
2. Uses AI to write an engaging, conversational podcast script (OpenAI GPT)
3. Converts the script to natural-sounding human speech (ElevenLabs TTS)
4. Saves the complete podcast as an MP3 audio file
5. Runs entirely from the command line

1.2 Learning Objectives

By completing this assignment, you will apply concepts from:

- How to make HTTP requests to external APIs
- Different authentication methods (API keys, Bearer tokens, custom headers)
- Asynchronous programming with async/await
- Environment variables and API security
- Error handling in backend applications
- File system operations (saving text and binary files)
- Working with different response types (JSON and binary data)

1.3 What You Get

- Complete starter code with helper functions
- Testing utility to verify each API integration
- This comprehensive handout with everything you need

IMPORTANT SECURITY WARNING:
DO NOT COMMIT YOUR API KEYS TO VERSION CONTROL. Use the `.env` file for your secrets, and never push it to GitHub or share it publicly. Your API keys are like passwords!

2 Prerequisites

2.1 Required Software

- Node.js version 14 or higher
- A text editor (VS Code, Sublime, Atom, etc.)
- Terminal/Command Line access
- Internet connection
- Audio player (to listen to your podcast!)

2.2 Required Knowledge

This assignment builds on concepts covered in class:

- **Class 1:** Command-line interface (CLI) basics
- **Class 2:** JavaScript fundamentals (variables, functions, arrays, objects)
- **Class 3:** Async/await and API consumption
- **Classes 9-10:** Backend development and API creation

3 Setup Instructions

3.1 Step 1: Get API Keys (FREE)

You need to sign up for three free services:

NewsAPI (5 minutes):

1. Visit: `https://newsapi.org`
2. Click "Get API Key" and sign up
3. Copy your API key
4. Free tier: 100 requests per day

OpenAI (10 minutes):

1. Visit: `https://platform.openai.com/signup`
2. Create an account
3. Go to: `https://platform.openai.com/api-keys`
4. Click "Create new secret key"
5. Copy the key immediately (you won't see it again!)
6. New accounts get $5 free credit

ElevenLabs (5 minutes):

1. Visit: `https://elevenlabs.io`
2. Sign up for a free account
3. Go to your profile (top right)
4. Copy your API key
5. Free tier: 10,000 characters per month (plenty for this!)

3.2 Step 2: Install Dependencies

```
cd api-podcast-pso
npm install
```

This installs:

- `axios` - For making HTTP requests
- `dotenv` - For loading environment variables

3.3 Step 3: Configure Environment Variables

```
cp .env.example .env
```

Edit `.env` and add your API keys:

```
NEWSAPI_KEY=your_newsapi_key_here
OPENAI_API_KEY=sk-your_openai_key_here
ELEVENLABS_API_KEY=your_elevenlabs_key_here
```

Important: Never commit the `.env` file to git! It contains your secret keys.

3.4 Step 4: Test Your Setup

```
npm test
```

This will verify that your API keys work.

4 Assignment Tasks

You will implement **four main functions** in `student-implementation.js`:

4.1 Task 1: Fetch News Articles (25 points)

Function: `async function fetchNews()`

What it does: Makes an HTTP GET request to NewsAPI to fetch trending news articles.

Key Concepts:

- HTTP GET requests
- Query parameters
- API key authentication
- Parsing JSON responses

Requirements:

1. Use the correct NewsAPI endpoint
2. Include required parameters (apiKey, country, category, pageSize)
3. Handle the response and extract the articles array
4. Return the array of article objects
5. Add proper error handling

Example output: Array of 5 news article objects

4.2 Task 2: Generate Podcast Script (25 points)

Function: `async function generateScript(articles)`

What it does: Sends the news articles to OpenAI's GPT model to generate an engaging podcast script.

Key Concepts:

- HTTP POST requests
- Bearer token authentication
- Request body formatting
- Working with AI APIs

Requirements:

1. Format the articles into readable text (helper provided)
2. Create a prompt for the AI (helper provided)
3. Make a POST request to OpenAI's chat completions endpoint
4. Include proper headers (Authorization, Content-Type)
5. Extract the generated text from the response
6. Save the script to a file
7. Return the script text

Example output: A 2-3 minute podcast script

4.3 Task 3: Convert to Audio (25 points)

Function: `async function generateAudio(script)`

What it does: Converts the podcast script to natural-sounding speech using ElevenLabs.

Key Concepts:

- Binary data handling
- Custom header authentication
- Audio file formats
- Working with ArrayBuffer responses

Requirements:

1. Construct the ElevenLabs endpoint with voice ID
2. Set up custom headers (xi-api-key)
3. Create request body with text and voice settings
4. Make POST request with responseType: 'arraybuffer'
5. Save the audio buffer to an MP3 file
6. Return the file path

Example output: An MP3 file you can play!

4.4 Task 4: Main Orchestration (25 points)

Function: `async function generatePodcast()`

What it does: Coordinates the entire process - fetch news, generate script, create audio, handle errors.

Key Concepts:

- Function composition
- Error handling with try/catch
- Validation
- User feedback (logging)

Requirements:

1. Validate environment variables are set
2. Call fetchNews() and check the result
3. Call generateScript() with the articles
4. Call generateAudio() with the script
5. Handle errors gracefully
6. Provide clear progress updates
7. Return a summary object

5 Grading Rubric

5.1 Core Functionality (75 points)

Component	Points	Requirements
NewsAPI Integration	25	Correct endpoint, parameters, error handling
OpenAI Integration	25	Correct endpoint, authentication, extracts response
ElevenLabs Integration	25	Custom headers, binary handling, saves audio
Main Function	25	Validates, orchestrates, handles errors

5.2 Code Quality (15 points)

Aspect	Points
Error Handling	4
Logging/Progress Updates	4
Code Organization	4
Comments	3

5.3 Testing (10 points)

Test	Points
Individual API tests pass	5
Full pipeline runs successfully	5

Total: 100 points

6 Testing Your Code

6.1 Test Individual APIs

```
# Test individual APIs
node test-apis.js news
node test-apis.js openai
node test-apis.js elevenlabs

# Or test all at once
npm test
```

6.2 Run the Complete Program

```
npm start
```

6.3 Expected Output

```
============================================================
AI PODCAST GENERATOR
============================================================
[OK] Environment variables validated

============================================================
STEP 1: Fetching trending news from NewsAPI
============================================================
[OK] Fetched 5 news articles

============================================================
STEP 2: Generating podcast script with OpenAI
============================================================
[OK] Script generated (456 characters)
[OK] Saved to: output/script.txt

============================================================
STEP 3: Converting to audio with ElevenLabs
============================================================
[OK] Audio generated: podcast_2024-10-21.mp3
[OK] Saved to: output/podcast_2024-10-21.mp3

============================================================
```

```
PODCAST GENERATION COMPLETE!
============================================================
[OK] News articles: 5
[OK] Script: 456 characters
[OK] Audio: output/podcast_2024-10-21.mp3

Listen to your AI-generated podcast!
```

7 Key Backend Concepts Covered

7.1 HTTP Methods

- **GET**: Retrieve data (used for NewsAPI)
- **POST**: Send data (used for OpenAI)

7.2 Authentication

- **Query Parameter**: `?apiKey=YOUR_KEY` (NewsAPI)
- **Bearer Token**: `Authorization: Bearer YOUR_KEY` (OpenAI)
- **Custom Header**: `xi-api-key: YOUR_KEY` (ElevenLabs)

7.3 Asynchronous Programming

```javascript
// Wrong - doesn't wait!
const result = fetchNews();

// Right - waits for completion
const result = await fetchNews();
```

7.4 Environment Variables

Why use them:

- Keep secrets out of code
- Different values for dev/production
- Prevent accidental sharing of keys

7.5 Error Handling

```javascript
try {
    const data = await apiCall();
    // Use the data
} catch (error) {
    console.error('Something went wrong:', error.message);
    throw error; // Re-throw or handle
}
```

8 Common Issues & Solutions

8.1 Issue: "Missing required environment variables"

- **Cause:** API keys not set in `.env` file
- **Solution:** Copy `.env.example` to `.env` and add your keys

8.2 Issue: "401 Unauthorized"

- **Cause:** Wrong API key or wrong authentication method
- **Solution:** Double-check your API key is correct and using the right header

8.3 Issue: "Cannot read property 'X' of undefined"

- **Cause:** Response structure is different than expected
- **Solution:** Log the full response: `console.log(response.data)`

8.4 Issue: Network Error

- **Cause:** Internet connection or wrong URL
- **Solution:** Check internet and verify API endpoints are correct

9 Submission Requirements

9.1 What to Submit

1. `student-implementation.js` (your completed code)
2. `.env.example` (template only - DO NOT submit .env)
3. A sample output file from `output/summary.txt`

9.2 What NOT to Submit

- `.env` file (contains your secret keys!)
- `node_modules/` folder (too large)
- Any temporary or test files

9.3 Before Submitting

1. Run `npm test` - all tests should pass
2. Run `npm start` - program should complete successfully
3. Remove any debug `console.log` statements
4. Add comments explaining your logic
5. Verify your code is clean and readable

10 API Documentation Links

- NewsAPI: `https://newsapi.org/docs`
- OpenAI: `https://platform.openai.com/docs`
- ElevenLabs: `https://elevenlabs.io/docs`
- Axios: `https://axios-http.com/docs/intro`
- Node.js: `https://nodejs.org/docs`

11 Extension Ideas (Optional)

Once you complete the core assignment, try these:

1. Add different news categories (sports, business, etc.)
2. Generate summaries in different formats (bullet points, paragraphs)
3. Add a date/timestamp to saved files
4. Fetch from multiple news sources
5. Create a scheduling system to run daily
6. Add email delivery of summaries
7. Support multiple languages

12 Getting Help

12.1 Debugging Checklist

1. Read the error message carefully
2. Check which step is failing (use the test utility)
3. Verify API keys are correct in `.env`
4. Use `console.log` to inspect data
5. Review the relevant task section in this handout
6. Check the official API documentation links

12.2 When to Ask for Help

- After trying to debug for 30+ minutes
- When you've checked all resources
- When you're not sure what the error means

12.3 How to Ask for Help

Provide:

1. What you're trying to do
2. What you expected to happen
3. What actually happened (error message)
4. What you've already tried

Good luck! This is a great project to add to your portfolio.

You're building real-world backend skills that employers value!