GitHub Developer Assessment

Professional Technical Evaluation for **vpoluyaktov**

User Overview

FIELD	VALUE
Username	vpoluyaktov
Name	Vladimir Poluyaktov
Company	Petabyte Technology Inc.
Location	Seattle
Email	vpoluyaktov@gmail.com
GitHub Since	2012-07-17T17:05:59Z

FIELD	VALUE
Public Repos	20
Private Repos	N/A
Forked Repos	13
Public Gists	0
Followers	1
Following	1
Plan	pro
Last Update	2025-07-28T19:48:28Z

Project Summary

Repository Overview

Repository Statistics

CATEGORY	COUNT	DETAILS
Total Repositories	21	Original: 8, Forked: 13
Most Active Forks	0	With >10 commits
Total Stars	25	Across all repositories

Original Repositories

REPOSITORY	STARS	DESCRIPTION	LANGUAGES	CREATED	LAST UPDATED
mp4_splitter	0	A command-line tool to split M4B audiobooks into individual chapter files	Go, Shell	2024-04-04T23:02:11Z	2025-08-01T01:27:08Z
cw-gen	0	A command-line tool to convert text into Morse	Go	2024-08-03T00:57:10Z	2025-08-01T01:26:08Z

REPOSITORY	STARS	DESCRIPTION	LANGUAGES	CREATED	LAST UPDATED
		code audio files			
github_evaluation_tool	0	GitHub developer assessment tool for OpenWeb UI	Python	2025-07-28T19:45:58Z	2025-08-01T01:24:48Z
abb_ia	15	Download a collection of .mp3 files from Internet Archive and create an audiobook in .m4b format	Go, Shell	2023-03-15T17:16:57Z	2025-07-09T15:44:18Z
pp_sdk	0	SDK for Product Partner	N/A	2024-09-08T16:39:00Z	2025-05-21T21:41:49Z
IA_audiobook_creator	9	Download a collection of .mp3 files from Internet	Python, Shell	2020-06-19T19:44:13Z	2024-01-13T00:29:18Z

REPOSITORY	STARS	DESCRIPTION	LANGUAGES	CREATED	LAST UPDATED
		Archive and create an audiobook in .m4b format			
EBook_audiobook_creator	0	Create an audiobook from an ebook	Python	2021-09-30T15:19:34Z	2022-07-06T17:58:33Z
morse-keyboard-pi	0	Morse keyboard and iambic keyer for Raspberry Pi	Python	2020-02-19T21:01:45Z	2022-07-06T17:54:52Z

Forked Repositories

REPOSITORY	SOURCE	USER COMMITS	STARS	LANGUAGES	LAST UPDATED
tview	fork	2	0	Go	2023-12-14T00:56:54Z

REPOSITORY	SOURCE	USER COMMITS	STARS	LANGUAGES	LAST UPDATED
freeLib	fork	0	0	C++, C, PLpgSQL, ...	2023-02-09T19:43:45Z
charts-1	fork	0	0	N/A	2022-02-14T11:09:54Z
bigquery-schema-generator	fork	0	0	N/A	2021-10-27T22:50:35Z
charts	fork	0	0	N/A	2021-06-03T00:22:39Z
cwwav	fork	0	0	N/A	2020-02-16T22:57:40Z
adonis-swagger	fork	0	0	N/A	2018-09-04T18:42:06Z
shinysdr	fork	0	0	N/A	2018-03-27T23:26:49Z
dump1090	fork	0	0	N/A	2017-09-22T00:31:25Z

REPOSITORY	SOURCE	USER COMMITS	STARS	LANGUAGES	LAST UPDATED
rx_tools	fork	0	0	N/A	2017-09-08T20:28:14Z
docker-registry-frontend	fork	0	0	N/A	2017-08-24T00:41:53Z
docker-homebridge	fork	0	0	N/A	2017-08-01T00:50:48Z
ZoneMinder	fork	0	1	N/A	2017-02-13T19:18:02Z

Technical Assessment

Code Quality Analysis

mp4_splitter

CATEGORY	ASSESSMENT
Architecture	Modular CLI tool, clear separation (utils, logger, ffmpeg, models, splitter). Uses idiomatic Go structure.
Error Handling	Uses Go error wrapping, custom error types, and propagates errors with context.
Testing	Includes unit tests for utils, splitter, and error handling. Good coverage for edge cases.
Documentation	README and inline comments are present. Public functions are documented.
Performance	Efficient file operations, avoids unnecessary allocations. Uses streaming where possible.

cw-gen

CATEGORY	ASSESSMENT
Architecture	Single-file Go CLI, but well-organized. Uses CGo for MP3 encoding. Modular waveform generation.
Error Handling	Checks for errors on all I/O and external calls. Returns detailed error messages.
Testing	Includes tests for main logic.
Documentation	Inline comments and usage help.

CATEGORY	ASSESSMENT
Performance	Uses efficient math and buffer operations for audio generation.

abb_ia

CATEGORY	ASSESSMENT
Architecture	Multi-package Go app with MVC-like split (controller, dto, ui, utils).
Error Handling	Consistent error propagation, context-rich error messages.
Testing	Comprehensive test coverage for utils, sorting, and controller logic.
Documentation	Good README, inline comments, and docstrings.
Performance	Optimized for batch file operations and streaming.

IA_audiobook_creator

CATEGORY	ASSESSMENT
Architecture	Monolithic Python script, but logically split into functions.
Error Handling	Uses try/except throughout, handles edge cases (file not found, bad metadata).

CATEGORY	ASSESSMENT
Testing	Manual and some automated tests.
Documentation	Extensive docstrings and usage comments.
Performance	Uses subprocess for ffmpeg, handles large files in batches.

EBook_audiobook_creator

CATEGORY	ASSESSMENT
Architecture	Modular Python app with clear separation (utils, TTS engines, parsers).
Error Handling	Uses try/except, logs errors, and provides user feedback.
Testing	Includes test scripts for processors and TTS.
Documentation	Docstrings and inline comments.
Performance	Handles audio processing efficiently, leverages ffmpeg and TTS libraries.

morse-keyboard-pi

CATEGORY	ASSESSMENT
Architecture	Modular Python project, clear separation between UI, utils, and sound generation.
Error Handling	Handles device and file errors, provides user notifications.
Testing	Some unit and integration tests for sound and lookup modules.
Documentation	Inline comments, some docstrings.
Performance	Real-time audio and UI handling; uses numpy and pyaudio efficiently.

tview (fork)

CATEGORY	ASSESSMENT
Architecture	Minor changes to a mature Go TUI library. Maintains upstream structure.
Error Handling	Follows upstream conventions.
Testing	Upstream tests plus minor additions.
Documentation	Follows upstream documentation standards.
Performance	No regressions introduced.

Representative Code

Go: Smart String Truncation (mp4_splitter)

```go
// SmartTruncate truncates a string to the specified maximum length while trying to
// preserve word boundaries. It will:
// 1. Try to truncate at the last space before maxLen
// 2. If no space is found, truncate at a word boundary if possible
// 3. If no word boundary is found, truncate at maxLen
// 4. Add ellipsis (...) if the string was truncated.
func SmartTruncate(s string, maxLen int) string {
    if len(s) <= maxLen {
        return s
    }
    s = strings.TrimSuffix(s, "...")
    ellipsisLen := 3
    if maxLen <= ellipsisLen {
        return s[:maxLen]
    }
    maxLen -= ellipsisLen
    lastSpaceIdx := strings.LastIndex(s[:maxLen+1], " ")
    if lastSpaceIdx > maxLen/2 {
        return s[:lastSpaceIdx] + "..."
    }
    // ... (see full code above)
}
```

Python: Audio Processing Pipeline (EBook_audiobook_creator)

```python
for chapter in parser.book_chapters:
    print('Narrating chapter {0}'.format(chapter['chapter_title']))
    filename = 'chapter_' + str(chapter_no)
    text = tts_engine.fix_pronunciation(chapter['chapter_text'])
    parser.save_text_to_file(text, 'tmp/' + filename + '.txt')
    if NARRATE_CHAPTERS:
        tts_engine.saveTextToMp3(text, 'tmp/' + filename + '.mp3')
    mp3_file_names.append(filename + '.mp3')
    chapter_names.append(chapter['chapter_title'])
    chapter_no += 1
```

Go: GitHub API Client (abb_ia)

```go
func (c *GithubClient) GetLatestVersion() (string, error) {
    url := fmt.Sprintf("https://api.github.com/repos/%s/%s/releases/latest", c.repoO
    resp, err := http.Get(url)
    if err != nil {
        return "", err
    }
    defer resp.Body.Close()
    if resp.StatusCode != http.StatusOK {
        return "", fmt.Errorf("failed to fetch latest release: %s", resp.Status)
    }
    var release Release
    err = json.NewDecoder(resp.Body).Decode(&release)
    if err != nil {
```

```
        return "", err
    }
    return release.TagName, nil
}
```

Technical Highlights

- **Notable Implementations**:
  - Smart string/file name sanitization and truncation logic in Go (mp4_splitter).
  - End-to-end audiobook creation pipelines, including TTS, audio normalization, and chapter metadata (Python).
  - Modular CLI tools with robust flag parsing and error handling.
  - Forked and improved open-source Go TUI library (tview) for custom UI needs.
- **Design Decisions**:
  - Preference for modular, testable code (Go: package separation, Python: utility modules).
  - Consistent use of error propagation and logging.
  - User-centric CLI design with clear feedback and help messages.
- **Areas for Improvement**:
  - Some monolithic scripts (notably in Python) could benefit from further modularization.
  - More extensive use of type hints and static analysis in Python code.
  - Increased use of CI/CD and automated linting across all projects.

- Broader adoption of docstring and API documentation standards.

Experience Level Assessment

1. Technical Proficiency

Architecture & Design

- Demonstrates strong understanding of modular CLI design in Go and Python.
- Implements cross-language solutions (Go, Python, Shell).
- Uses advanced Go features (custom error types, interfaces, CGo for audio encoding).
- Designs pipelines for complex tasks (audiobook creation, TTS, audio processing).

Implementation

- Implements robust file and string utilities, audio processing, and API clients.
- Efficient use of standard libraries and third-party packages.
- Handles edge cases and provides user-friendly error messages.

Example: Go Error Handling

```
if err != nil {
    return 0, fmt.Errorf("failed to get file size: %w", err)
}
```

2. Software Engineering Practices

Code Quality

- Consistent use of idiomatic Go and Python.
- Includes unit and integration tests for core logic.
- Uses logging and error reporting effectively.

Project Management

- Frequent, descriptive commit messages.
- Uses feature branches and pull requests.
- Maintains and updates documentation and CI configs.

Example: Python Modularization

```
class AudioProcessor:
    def noise_filter(self, sound_np_array, output_sample_rate, noise_samples_dir, vo
```

```
        # ...
```

Level Mapping

Software Developer Level Matrix

LEVEL	GOOGLE	AMAZON	MICROSOFT	TYPICAL EXPERIENCE
Entry	L3 – Software Engineer	SDE I	Software Engineer (Level $^{59}/_{60}$)	0–2 years
Mid	L4 – Software Engineer	SDE II	Software Engineer (Level $^{61}/_{62}$)	2–5 years
Senior	L5 – Senior Software Engineer	SDE III / Senior SDE	Senior Software Engineer (Level 63)	5–8 years
Staff	L6 – Staff Software Engineer	Principal SDE	Principal Software Engineer (Level 65)	8–12 years
Senior Staff / Principal	L7 – Senior Staff Software Engineer / L8	Sr. Principal SDE or Sr. Engineer	Partner / Distinguished	12–16+ years

LEVEL	GOOGLE	AMAZON	MICROSOFT	TYPICAL EXPERIENCE
	– Principal Engineer		Engineer (Level $^{67}/_{68}$)	
Distinguished / Fellow	L9/10 – Distinguished Engineer / Google Fellow	Sr. Principal / Distinguished Engineer	Technical Fellow / CVP (Level $^{69}/_{70}$)	Rare, Top 1%

Assessment Criteria by Level

AREA	OBSERVED EXAMPLES	ASSESSED LEVEL	MICROSOFT EQUIVALENT	GOOGLE EQUIVALENT	AMAZON EQUIVALENT
Code Quality	Modular Go/Python, idiomatic code, robust error handling, unit tests	Senior	Level 63	L5	SDE III / Senior SDE
System Design	CLI tools, modular pipelines, custom TUI, audio processing pipelines	Senior	Level 63	L5	SDE III / Senior SDE

AREA	OBSERVED EXAMPLES	ASSESSED LEVEL	MICROSOFT EQUIVALENT	GOOGLE EQUIVALENT	AMAZON EQUIVALENT
Testing	Unit/integration tests, edge case coverage, test utilities	Senior	Level 63	L5	SDE III / Senior SDE
Error Handling	Consistent error propagation, user feedback, custom error types	Senior	Level 63	L5	SDE III / Senior SDE
Documentation	Good README, inline comments, usage help, but some scripts could be improved	Senior	Level 63	L5	SDE III / Senior SDE

Overall Level Recommendation

Senior Software Engineer
- Microsoft: Level 63
- Google: L5
- Amazon: SDE III / Senior SDE

Justification:
Vladimir demonstrates strong architectural skills, robust implementation of complex CLI and audio processing pipelines, and a mature approach to error handling and testing. The code is idiomatic, modular, and well-documented. There is clear evidence of technical leadership in open-source contributions (e.g., forking and extending tview), and the ability to deliver end-to-end solutions in both Go and Python. While some Python scripts could be further modularized, the overall engineering practices, code quality, and system design are consistent with a Senior Software Engineer at top-tier tech companies.

Summary & Recommendation

Vladimir Poluyaktov is a highly capable software engineer with a strong track record of building robust, modular CLI tools and audio processing pipelines in both Go and Python. His repositories demonstrate:

- **Strong code quality**: Idiomatic use of Go and Python, modular design, and comprehensive error handling.
- **Solid architecture**: Well-structured CLI applications, modular pipelines, and custom tooling for audio and text processing.
- **Problem-solving ability**: Tackles complex tasks such as TTS integration, audio normalization, and file management with practical, maintainable solutions.
- **Technical depth**: Uses advanced language features, integrates with native libraries (CGo), and extends open-source projects.
- **Best practices**: Maintains tests, documentation, and CI/CD configs; uses descriptive commits and PRs.

Recommendation:

Vladimir is recommended for Senior Software Engineer roles (Microsoft Level 63, Google L5, Amazon SDE III/Senior SDE). He is well-suited for positions requiring ownership of complex backend, CLI, or tooling projects