DocProcessor

सामग्री

प्रलेखन को क्यूए स्वचालन के लिए एक सत्यापन योग्य फीचर मैप में बदलें।

सारांश

DocProcessor एक स्वतंत्र, पूर्णतः अलग किया गया Go मॉड्यूल है जो प्रोजेक्ट प्रलेखन लोड करता है, संरचित फीचर मैप बनाता है और सत्यापन कवरेज को ट्रैक करता है। इसे LLM एजेंटों के साथ बुद्धिमान फीचर निष्कर्षण के लिए डिज़ाइन किया गया है, लेकिन इसमें ऑफ़लाइन उपयोग के लिए ह्यूरिस्टिक निष्कर्षण भी शामिल है।

संक्षिप्त विवरण

प्रोजेक्ट-अज्ञेयवादी Go मॉड्यूल प्रलेखन प्रसंस्करण और फीचर-मैप निष्कर्षण के लिए। यह प्रलेखन को संरचित फीचर मैप में पार्स करता है और ट्रैक करता है कि कौन-से फीचर सत्यापित हैं—बुद्धिमान निष्कर्षण के लिए LLM एजेंटों का उपयोग करके या ऑफ़लाइन ह्यूरिस्टिक्स के माध्यम से—जो क्यूए स्वचालन को एक अचूक, हमेशा-वास्तविकता से मेल खाने वाली गारंटी प्रदान करता है।

विस्तृत विवरण

हर सॉफ़्टवेयर टीम एक ही धीमे झूठ के साथ जीती है: प्रलेखन फीचर का वादा करता है, टेस्ट कुछ और कवर करते हैं, और कोई भी यह कहने का आत्मविश्वास नहीं रखता कि दोनों एक ही उत्पाद का वर्णन कर रहे हैं। DocProcessor इसी अंतर को दृश्यमान और मापने योग्य बनाने के लिए मौजूद है। किसी प्रोजेक्ट के प्रलेखन को देखते हुए, यह एक संरचित फीचर मैप बनाता है—उत्पाद द्वारा किए जाने वाले हर दावे का एक गणनीय, मशीन-पठनीय मॉडल—और इसके विरुद्ध सत्यापन कवरेज को ट्रैक करता है, ताकि "क्या यह प्रलेखित फीचर वास्तव में साबित हुआ है?" जैसा सवाल गलियारे की बहस न रहकर एक ऐसा प्रश्न बन जाए जिसका उत्तर हो। यह जानबूझकर दोहरे मोड में काम करता है: जब उपलब्ध हो तो यह LLM एजेंटों का उपयोग बुद्धिमान, सिमेंटिक फीचर निष्कर्षण के लिए करता है, और पूरी तरह ऑफ़लाइन उपयोग के लिए ह्यूरिस्टिक पार्सर पर निर्भर करता है, ताकि यह कभी भी किसी मॉडल पर कठोर निर्भर न रहे और एक एयर-गैप्ड सीआई जॉब या डेवलपर के हवाई जहाज़ मोड वाले लैपटॉप पर समान रूप से चले।

वास्तुशिल्पीय रूप से यह एक स्वतंत्र, प्रोजेक्ट-अज्ञेयवादी, पूर्णतः अलग किया गया Go मॉड्यूल (CONST-051(B)) है: यह कोई प्रोजेक्ट-विशिष्ट मान नहीं भेजता और उपभोक्ताओं द्वारा समान-कोडबेस सबमॉड्यूल के रूप में शामिल किया जाता है, ताकि कोई

भी प्रोजेक्ट इसे अपनाए बिना किसी और के पूर्वधारणाओं को विरासत में न ले। यह स्वयं पर भी वही मानक लागू करता है जो दूसरों पर थोपता है—इसके अपने दावे एंटी-ब्लफ़ प्रतिज्ञा (CONST-035) और पूर्ण-स्वचालन कवरेज नियमों (CONST-048) से बंधे हैं, जिसका अर्थ है कि इसके README में विज्ञापित हर क्षमता का एक स्वचालित टेस्ट या चैलेंज स्क्रिप्ट द्वारा अभ्यास किया जाता है जो केवल शून्य पर समाप्त होने के बजाय वास्तविक, अंतिम-उपयोगकर्ता के लिए उपयोगी व्यवहार की पुष्टि करता है; उपयोगकर्ता-सामना वाले स्ट्रिंग्स CONST-046 i18n अनुवादक सीम से गुजरते हैं। इसका उद्देश्य एक बंद लूप बनाना है: DocProcessor क्यूए चक्र का इनपुट पक्ष है जिसे HelixQA पूरा करता है—यह प्रलेखन से फीचर मैप निकालता है, HelixQA प्रत्येक मैप किए गए फीचर को कैप्चर किए गए रनटाइम प्रमाण के साथ साबित करता है, और प्रलेखन, टेस्ट और शिप किया गया व्यवहार रिलीज़ दर रिलीज़ चुपचाप अलग होने के बजाय एक-दूसरे के साथ मेल खाने के लिए बाध्य होते हैं।

हमने इसे क्यों बनाया

प्रलेखन और टेस्ट अलग हो जाते हैं: प्रलेखन ऐसे फीचर का वादा करता है जिसका कोई टेस्ट प्रमाण नहीं देता, और क्यूए आसानी से यह नहीं बता सकता कि "पूर्ण" का क्या अर्थ है। DocProcessor प्रलेखन को एक मशीन-पठनीय फीचर मैप में बदलता है ताकि सत्यापन कवरेज को वास्तव में वादे के अनुसार मापा जा सके।

सामग्री

यह क्यों एक गेम-चेंजर है

यह सॉफ़्टवेयर डिलीवरी के सबसे अस्पष्ट सवाल—"क्या हमने जो भेजा, वह वही है जो हमने भेजने का दावा किया था?"—को एक स्वचालित और निरंतर जाँच योग्य प्रक्रिया में बदल देता है। और यह बिना किसी कड़े **AI** निर्भरता के संभव होता है: जब मॉडल उपलब्ध हो तो **LLM** निष्कर्षण, और जब न हो तो अनुमान-आधारित विधियाँ, ताकि यह गारंटी हर परिवेश में बनी रहे—चाहे वह ऑफ़लाइन रनर हो या पूरी तरह स्वचालित पाइपलाइन।

क्या है नवीन

- दस्तावेज़ीकरण से फीचर-मैप निष्कर्षण, सत्यापन-कवरेज ट्रैकिंग के साथ।
- दोहरा निष्कर्षण: **LLM**-एजेंट संचालित या अनुमान/ऑफ़लाइन।
- प्रोजेक्ट-अज्ञेयवादी, शून्य-कॉन्फ़िगरेशन डिकपलिंग (**CONST-051(B)**)।
- ब्लफ़-रोधी स्व-सत्यापन: README के दावों की पुष्टि टेस्ट/चैलेंज के ज़रिए (**CONST-035/048**)।

चुनौतियाँ और समाधान

- **मॉडल-ऑप्शनल ऑपरेशन:** अनुमान-आधारित निष्कर्षक फ़ॉलबैक से हल, ताकि मॉड्यूल ऑफ़लाइन भी काम करे।
- **दस्तावेज़ और वास्तविकता को समन्वित रखना:** संरचित फीचर-मैप और सत्यापन-कवरेज ट्रैकिंग से हल, जो QA लूप से जुड़ा है।
- **पुन:प्रयोगिता:** सख्त डिकपलिंग और समान कोडबेस सबमॉड्यूल उपभोग से हल।

- **अपने दावों की विश्वसनीयता:** हर विज्ञापित क्षमता के लिए ब्लफ़-रोधी टेस्ट/चैलेंज से हल।

तकनीकी ढाँचा (क्यों + कैसे)

- **Go (1.25+)** — मॉड्यूल का मुख्य भाग; Apache-2.0 लाइसेंस प्राप्त।
- **LLM एजेंट** — बुद्धिमान सिमेंटिक फीचर निष्कर्षण (वैकल्पिक)।
- **अनुमान-आधारित पार्सर** — ऑफ़लाइन फीचर निष्कर्षण का फ़ॉलबैक।
- **i18n अनुवादक (`pkg/i18n`)** — **CONST-046** स्थानीयकृत स्ट्रिंग्स।
- **चैलेंज हार्नेस** — मॉड्यूल के स्वयं के दावों का ब्लफ़-रोधी सत्यापन।