Alpha Test of Ddog (Diction Dog), the Diction Analyzer Version 0.1

Software Developer:
Bradford Connatser, Tutor, Craven Literacy Center
bconnatser1245@gmail.com, 541-430-4548

Issue Date: July 27, 2026

Introduction

Many thanks for agreeing to alpha test a new software designed to help tutors generate texts at the appropriate reading level for a particular student. What is an alpha test? It is the first phase of software testing and is usually limited to people within the corporation that builds the software. The goal of the alpha test is to help the developer (me!) find bugs, functional problems, functional shortcomings, usability issues, and specific targets for improvement. Because the software is under development, aesthetics (the "skin" of the software) will not be one of your areas of focus (although I would love to hear ideas about that from designers and illustrators)—unless an aesthetic element interferes with the operation of the software.

Time Commitment and Skipping Ahead

Have you ever met anyone who has time? If you are interested in saving time, please skip the Student Background, The Problem, and The Proposed Solution sections and start your test on page 7, Introduction to the Software. The next three sections provide a context for this software, why I am developing it and the need that it satisfies (or *will* satisfy), all of which are not required reading for the alpha test. I estimate that the entire test will take about an hour, but you may want to experiment with the software features, which will add time to your chore and many thanks from me.

Student Background

I am developing this software to enable me to write texts (short stories, in this case, and a graphic novel) for a student (let's call him *Jdog*) who has a learning disability (autism with ADHD). This student has full control of his onboard language capabilities—speaking and hearing English. At 19 years old, he cannot read because the correlation between spoken words (the words he uses) and written words (the squiggly lines that represent those words) is very weak. For example, because his family lost their vehicle, I sometimes take Jdog to and from the literacy center (I go by his house on the way to the center, as it happens). On the way to the center one day, we passed White's Tire Service, with a big sign on the road bearing the name "WHITE'S." I exclaimed, "Jdog, you can read this one!" He couldn't. His surname is *White,* and he couldn't read the sign. He has trouble with "wh" words, like *white.* Still, not recognizing your own name in print is a stark indication of this young man's reading difficulties.

A few months ago, I got a note from Jdog's mother, and I keep the note to keep me motivated. She was worried that the text that I had selected for in-class reading (a graphic novel) was too difficult for him, and she wanted him to experience *success* while reading. My first reaction to the letter was to defend the difficulty of the text: I needed to witness the failure modes, how Jdog struggles with a word that he does not recognize in print, what methods he employs to discover its pronunciation and thus recover its identity from the speech stream. However, his mother was right. What a boost to Jdog's confidence to complete a full sentence without stumbling over words. I searched the Internet, visited Books a Million, and spoke to several experts in the challenges of teaching autistic people to read. I was so disappointed when I failed to identify texts that matched the criteria that I set for Jdog.

The Problem

The criteria for any practice text for Jdog are:

- The *diction* must be simple, even at the grade-school level. However. . . .
- The *subject matter* must be for adults, because Jdog is an adult with a (very) adult vocabulary.

Simple diction with grownup syntax and adult themes seems to be an uncommon feature of available reading material and nonexistent for autistic readers. If the diction is juvenile—say, for a first-grader—then the subject tends to be commensurably juvenile. On the other hand, if the subject is for adult minds, then the diction is more advanced. Neither scenario works for Jdog. He eschews children's books (he probably feels infantilized, which is a no-starter for him), yet he needs to be able to read the words in those children's books. Where are the proper texts for Jdog? Where are the read-along texts for an adult with good spoken-language skills but limited recognition of words in print?

I spent some time with Jdog explaining the difference between knowing a word and recognizing it in print. For you, the alpha tester, I would like to explain the difference between knowing a word (cognitive vocabulary) and recognizing its representation in print (the written word) through an illustration. As shown in Figure 1, language users have vocabularies of words that they know when they are heard (**Cognitive** vocabulary), words that they know and often use (**Executive** vocabulary), and—unfortunately but true for us all—words that they don't know but use anyway (the "**Oops**" in the figure).

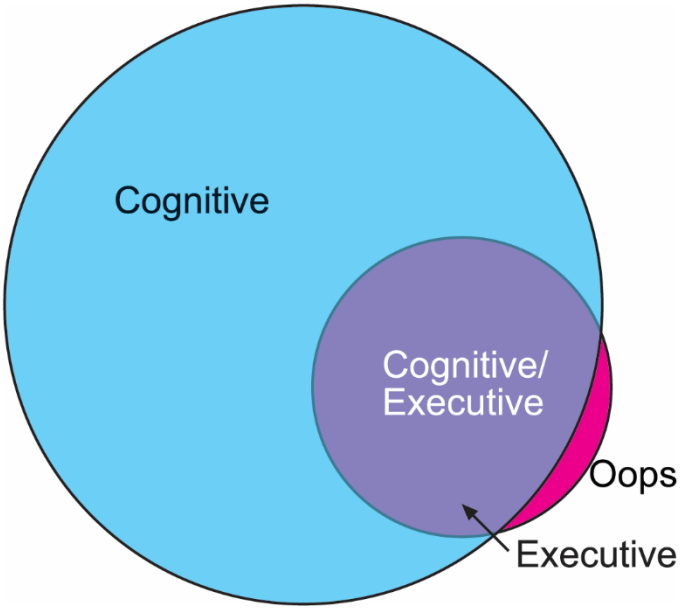
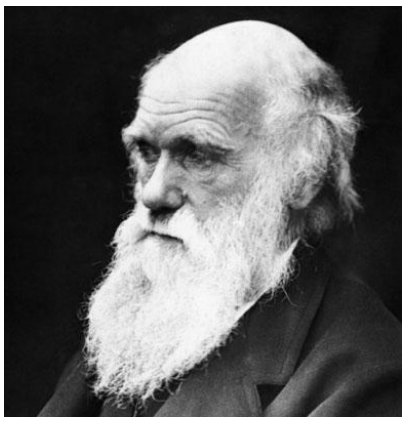


Figure 1: Vocabulary of Language Users



Now we must add a layer of linguistic complication called "writing," which enables people to transform words from the auditory channel (a spoken word, such as /craft • ee/) to a visual channel (a printed word, such as *crafty*). This is an unnatural transformation that requires years of learning and practice to master. As Darwin said, "man has an instinctive tendency to speak, as we see in the babble of our young children; whilst no child has an instinctive tendency to brew, bake, or **write**." The *organic* linguistic machinery in Jdog's brain runs well. He speaks like a teenager and is at times effusive and articulate. For example, Jdog likes snacks, but he made it clear that a soda by itself or a bag of chips by itself does not constitute a snack. A snack must include something to eat *and* drink. I asked him to categorize a solitary soda or bag of chips. If they are not snacks, what are they? "Treats," he said, beautifully rising to the sudden linguistic challenge. He understands his native language most excellently. I think that it would be congruous with Darwin's insight to say that a person may be good at one form of expression (auditory) and not another (visual), especially when one is organic to humans but the other is synthetic (an artifact of human society). Further, in some cases, that difference in skillset may be stark, as in the case of Jdog. Figure 2 illustrates the problem. Although Jdog's organic language skills are adequate, his ability to recognize *in print* the words that he knows by ear is just a morsel of his cognitive vocabulary.

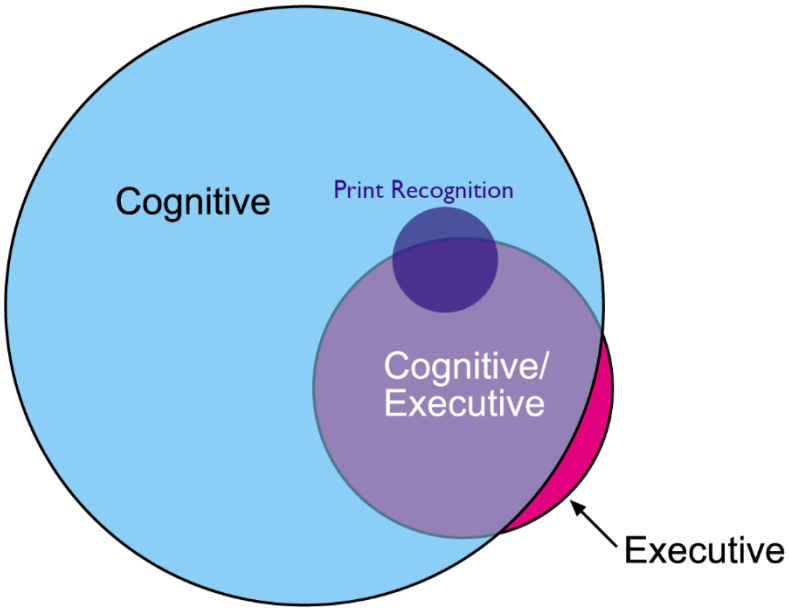


Figure 2: Aural/Oral Vocabulary Versus Print Recognition

The human writing system, which varies remarkably from language to language, requires the coding and decoding of language sounds (*phonemes,* the smallest unit of spoken language). For example, consider *coding* a word from a sound (/craft • ee/) to a printed word (*crafty*) when writing, or *decoding* a word from a printed word to a sound when reading. However, our alphabet letters do not perfectly represent all the English phonemes (of which there are 44, not 26 as the alphabet sort-of promises). In fact, the correlation between spelling and pronunciation is pathetic (pathetik?) for the English alphabet (alfabet?), because of, in part, a) the flood words borrowed and injected from other languages and b) the

understandably imperfect process of setting spelling standards, all resulting in silent letters, balls and clusters of letters that make non-native speakers say, “You’ve got to be kidding me!”

With all these complications recognized, I have to get Jdog reading—reading first, and *then* writing (that is a problem for some other kind of software!).

The Proposed Solution

In the case of Jdog, I propose writing “reader-ready” texts for him to use during reading practice, merging the content of those texts with his world, employing his vocabulary, and presenting relevant subject matters so that he can be tested for comprehension and recall, which can be tracked by the proposed software.

Specifically, I’ve been working on a graphic novel, which features bursts of text on a graphics-rich pages instead of line after line of prose. The superhero in the graphic novel, Tantalus (big green shirt with a purple T on the front), is a mix of me and my student: I simply typed into ChatGPT: “middle-aged and balding African American with freckles, a rotund belly, and nerd glasses.” He is autistic, a bit (very) clumsy, and, of course, demonstrably out-of-shape.

To help me create objectively readable texts for Jdog (simple vocabulary, adult theme), I developed Ddog—Diction Dog, the Diction Analyzer. Imagine a tutor who wants to develop student texts (such as practice texts or diagnostic texts). She knows that her student is having trouble reading “difficult” material, but how does she know whether reading material is objectively difficult? With this software, a tutor can:

1. Write a student text (let’s say, a short story), knowing that the student has difficulty with certain spelling patterns and polysyllabic words, doing her best to select words that are appropriate for the student.
2. Submit all or part of the text for analysis to Diction Dog.
3. Use the results to fine-tune the text and anticipate trouble spots (which the tutor may want to intentionally introduce to the student to challenge him). The software will report the results of the analysis, offer suggestions for helping the student improve his reading skills, and offer synonyms for words that exceed thresholds.
4. Test single words quickly to determine their fitness.
5. Save settings for particular students and track their progress.
6. Load settings from saved settings to ensure consistency in analyzing texts.

The user can submit a word or group of words (such as complete sentences or a random sampling of words) for analysis to determine the likelihood that the studen knows those words by sight, depending on *the kinds of reading difficulties* that the student has. With the aid of this software in its alpha stage (this is Version 0.1), I plan to write nine calibration texts (*cats* is the theme), one of which I have completed and used to tune the software already (see Appendix A). I will use these texts to finetune the software to ensure easy reading for Jdog.

The experimental question for this software is: Does the software objectively predict ease of word-identification, such as predicting troubling words based on known spelling challenges? Based on the first calibration test, the answer is tentatively *yes.* I was sweating during the calibration tests as Jdog read the first cat story out loud, but he stumbled where the software predicted that he would, such as *Annabelle*

and *curiosity.* Why? *Annabelle* is a word that infrequently appears in print (this can be objectively estimated based on public book collections), has three or more syllables, contains a silent letter, and contains some tricky spelling patterns (double letters "n" and "l"). *Curiosity* is a word that appears much more frequently than *Annabelle* but has spelling troubles of its own, such as "ur" blending to produce one sound and "io" forming two separate vowel sounds. Moreover, it has five syllables. Jdog consistently stumbles over polysyllabic words, and therefore syllable count seemed a natural starting point for me to develop objective metrics that can be employed to measure characteristics of a word to predict its difficulty for Jdog (and perhaps others with his seemingly rare reading disability). This is why the filenames of the software contain "countSyllables," which, in the beginning, was all that I was trying to do.

For months, I have been observing Jdog as he reads and collecting notes about his reading failures, thereby gleaning insights that can be turned into rules for the software. Let's consider the word *white* again. Why does he have trouble with it? *White* has three spelling complications: First, it contains the consonant digraph "wh"—two letters joined to form one sound, and this happens a lot in English. Second—and with a bit of an overlap—it contains two silent letters: the first being the "h" in "wh" and the second being the "e" in the terminal "te." Third, it is rather long in letters for its snappy single syllable. Compare the word *white* with the word *win,* as shown in the tables below. Jdog knows both words, but the representation of the word *white* in print trips his linguistic feet. *White* has 67% more letters, and two of the letters are silent. On the other hand, "win" has no silent letters and no blended consonants or vowels at all. Therefore, the software will eventually include rules about *many* ways that diction can cause reading paralysis, and those rules will penalize words for containing silent letters, containing consonant digraphs and blends, or exceeding a threshold for letter count per syllable (5/1 versus 3/1), to name a few.

Word: white		
	By Sound	By Sight
Know it.	X	
Don't know it.		X

Word: win		
	By Sound	By Sight
Know it.	X	X
Don't know it.		

One clarification that I feel compelled to make is that although Jdog tests at the kindergarten level for reading skills, he does not read at all. His vision moves linearly through prose one letter, one syllable, one small word at a time. By the time that he reaches the end of a sentence—even if he knows all of the words—he is not processing the words to create meaning, and therefore the sentence does not collapse into a single unit of meaning as it would for veteran readers. When he stumbles, he cannot see the forest (the word) for the trees (its letters).

How does a typical reader read, then, if not linearly? Figure 3 shows typical eye movements during the reading process, where the reader focuses on a spot in a sentence (central focus), picks up the text in the central focus during this *fixation* (to the right and left of the central focus), and then moves to a subsequent spot (a motion called a *saccade*) or a previous spot in the case of confusion (a motion called a *retrograde saccade*). It is an odd process that evolves from learning to spell, read, and understand written language (*odd* because I've always "felt" that reading is a linear process, minus the mistakes that we must visually repair). Jdog reads with his nose pointed at the word of interest as if it were his foe.

Technically, he's not reading, but he is learning to read and understands the importance of integrating into a literate society. He must practice.

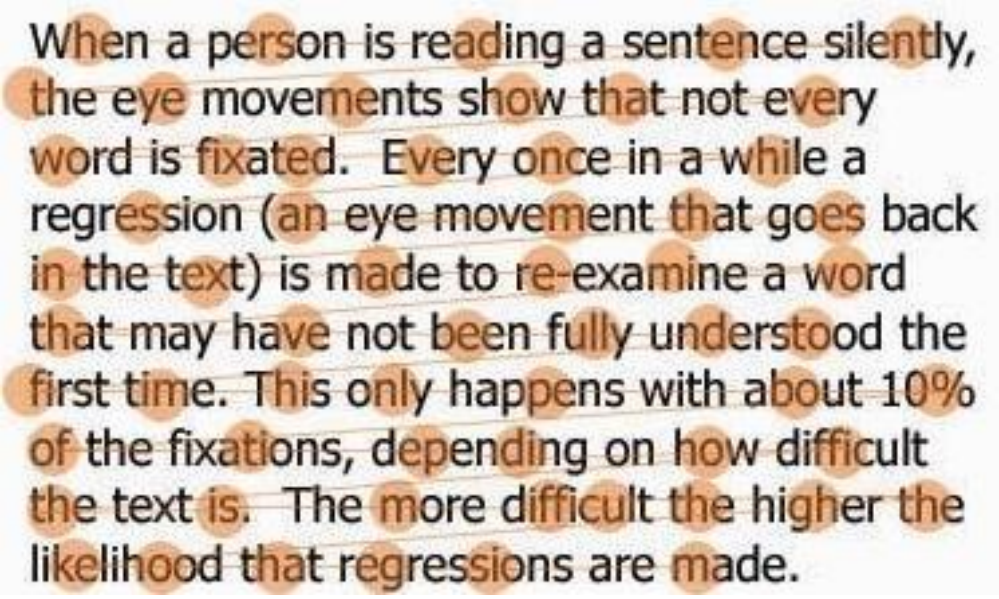


Figure 3: Eye Movements and Fixations of a Typical Reader During Reading

Introduction to the Software

Ddog is a *diction* analyzer. The user enters a word or group of words, and the software analyzes each word in isolation to the others and provides metrics of difficulty. The user can use the values of these metrics to refine prose for ease of reading—to make it easier or more challenging to read. It works at the word level, not the sentence level. It does not estimate the difficulty of a sentence or passage of several sentences (which is the goal of readability formulas), but Ddog does provide average values that can be interpreted as the estimated "diction difficulty" of the provided text. Here are the rules and their violation penalties:

Existing Metrics

I flag words that Jdog does not recognize in print and study them for common points of difficulty, such as difficult spelling patterns (the "wh" in *white,* for example). I turned several difficulty metrics into rules. If a word violates a rule, it is penalized. Therefore, all metrics are numerical, as shown in Figure 4, and are adjustable via sliders. For now, the software penalizes a word for violating the following rules:

- Threshold for How Common the Word Is: The "Over Frequency Threshold" adds a difficulty penalty between 5 and 40 points. The software downloads hundreds of thousands of words from a database. The words are in order of how often (frequency) they appear in print. If the threshold is set at 20 (see Figure 4), then the software will download the top 20% of the words in the database. If the word is among those top-20% words, then it gets a pass, but if it is not, then the word is penalized according to the setting "Over Frequency Threshold," shown in Figure 4. The software also checks for the word's rank among the top X% to indicate relative frequency of the word in print within the clutch of words downloaded from the database.

- Number of Syllables: Each syllable for Jdog is a mountain that he must climb. Any syllable beyond the first syllable incurs a penalty of 1 to 4 points (for each syllable above one).
- Difficult Spelling Patterns: I gathered these patterns from words unrecognizable to Jdog and created a rule: If a word "Contains a difficult spelling pattern: ght|igh|eigh|gh|wh," then it incurs a penalty of 1 to 4 points (per word).
- Diphthongs and Triphthongs: These could be categorized as "Difficult Spelling Patterns," but they pose a unique challenge to Jdog, so I carved them out. Each diph- or triphthong incurs a penalty of 1 to 4 points (per instance, because there may be more than one per word).
- Contractions: Jdog falters when trying to read contractions. Therefore, each instance of a contraction incurs a penalty of 1 to 10 points (per word).

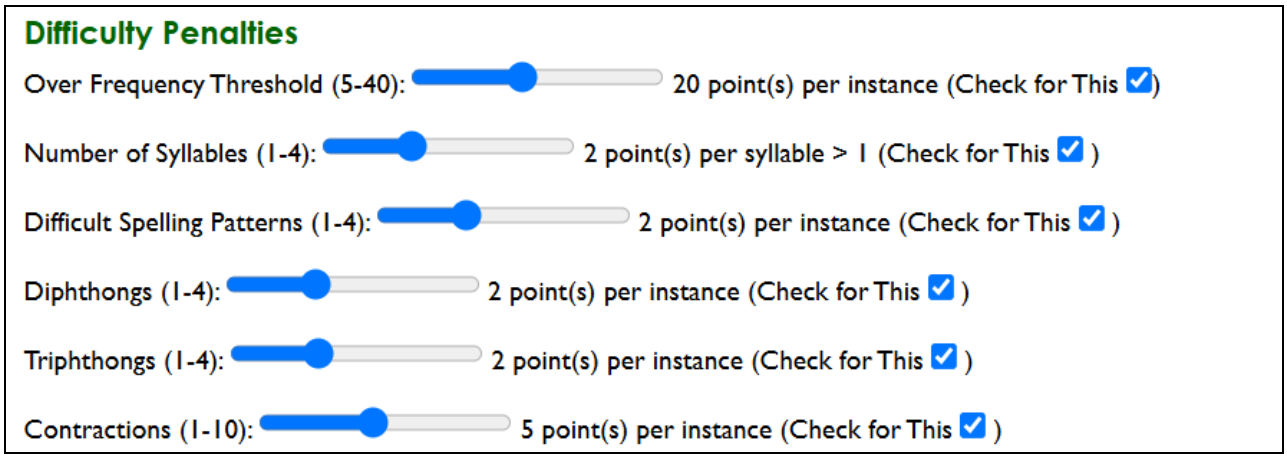


Figure 4: Difficulty Penalties

Thresholds are used for alerting the user to words and averages that exceed a threshold for the metrics shown in Figure 5.

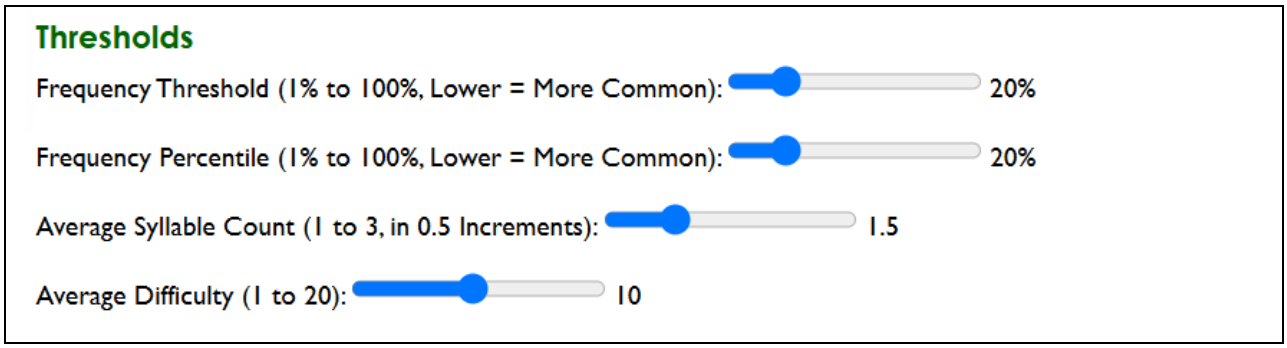


Figure 5: Thresholds for Signaling Problems with a Word or Group of Words (Averages)

Planned Metrics

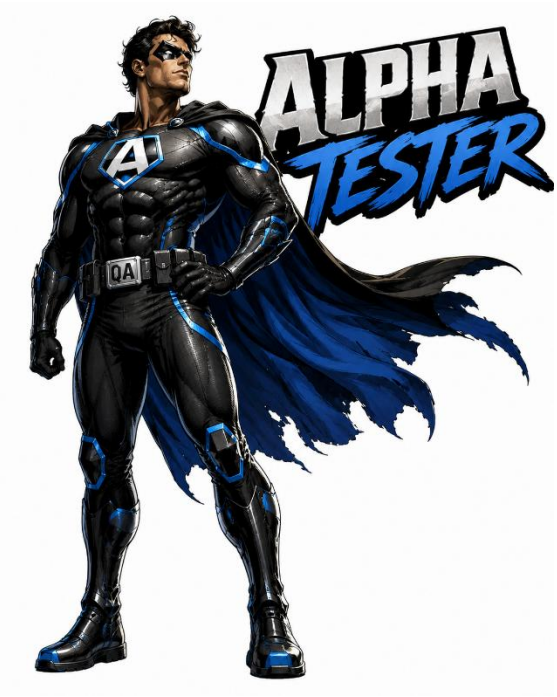
As I learn more about the features of words that make them difficult for Jdog to parse, I will add rules to the existing mix (they, too, will be customizable). I invite you, appreciated alpha tester, to suggest other metrics to include in the software. Proposed new metrics include:

- Word Length: The length of a word—perhaps letters per syllable—seems to correlate with Jdog's reading problem. This is an easy rule to add, but I may somehow integrate it with syllable count.

- Silent Letters: Silent letters complicate learning an unrecognizable word, especially if the reader is using phonics to recover a word by "sounding it out." This rule will be difficult to implement unless there is an existing list of not just words that have silent letters but which letters in the words are silent.
- Consonant Blends and Digraphs: I am searching for a good list of these troublesome gatherings of letters. I like my chances.
- Foreign Influence: Latin and Greek spellings confuse Jdog when he does not recognize a word in print. For example, the "ph" in *alpha tester* or "pt" in *receipt* (which overlaps the silent-letter rule—in this case, the "p" in "pt"). The English language is replete with words derived from Latin, words that have poor spelling-to-pronunciation correlation. Many of these can be eliminated during careful composition of the student texts, but there are so many foreign words and spelling patterns in English that objectively flagging them may prove useful.

Tasks of the Alpha Tester

Thank you, alpha tester, for spending your time testing software that will (hopefully) benefit the Craven Literacy Council and its tutors. Alpha and beta testers are the heroes of software development. Your goal is to test the software's usefulness, but if you happen upon a bug, I would appreciate your reporting that. Alpha testing is also about finding missed opportunities and both broad and narrow areas for improvement. Alpha testing also road-tests the test instructions (the document you are now reading) to enable the software developer to refine the instructions for subsequent beta testing. The following tasks, when completed, will constitute your alpha test.



What You Need

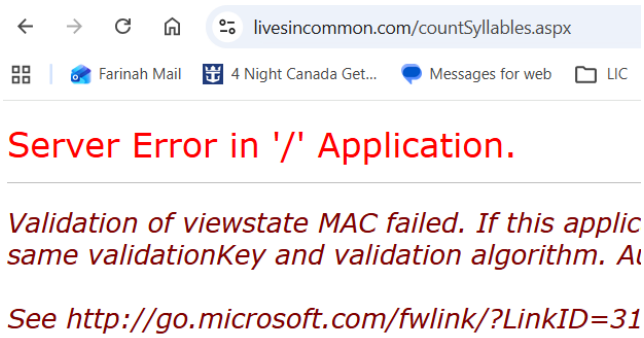
For your test, you will need a computer with a monitor (I haven't designed Diction Dog for a phone yet), a connection to the Internet, and a browser (I use Google Chrome, but the software should work on other popular browsers).

Load the Web Application Called "Diction Dog! Version 0.1."

You can load the web application by pointing your browser to:

https://livesincommon.com/countsyllables.aspx

NOTE: If you get the server error shown below, then open the link in a new browser tab. This web app is not complete, and sometimes the validation key expires during pauses in use.



Get Familiar with the GUI

After the page loads, please review the graphical user interface (GUI), shown in Figure 6. The features of the GUI are:

- Submitting Words for Analysis: At the top of the software is a text box to capture the words to be analyzed.
- Customizing Controls: The customizing controls of the GUI include sliders, which may be adjusted to set a value for a threshold or penalty, and checkboxes, which enable the user to turn penalty checks on and off.
- Buttons: At the bottom of the screen are buttons that enable the user to **Submit** words for analysis, **Reset** the sliders and checkboxes to their default values, **Load Custom Settings** for thresholds and penalties, and **Save Settings**.

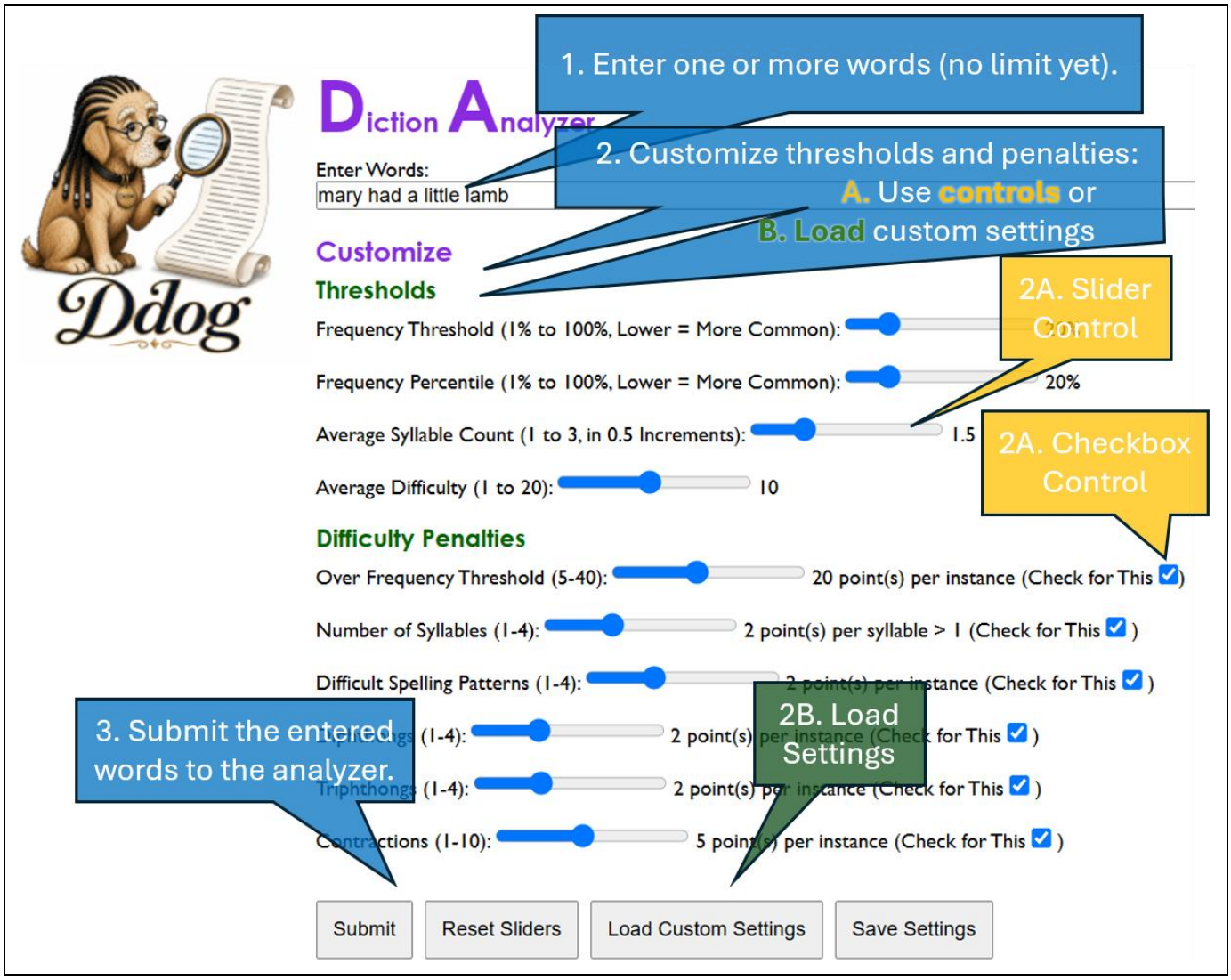


Figure 6. The Graphical User Interface for Diction Dog

The GUI controls are described below:

- Enter Words (text input box): This control accepts a word or multiple words. Because the software focuses on words in isolation to each other, all punctuation gets stripped before the software analyzes the input. A future version may include a bigger text box for more robust input with smart text controls.
- Customize: Thresholds (Used for Alerting Problems):
  - Frequency Threshold: A word's frequency is an indication of how often the word appears in print, based on the analysis of modern texts. The threshold is a percentile. For example, if you set the threshold to 20%, you are instructing the software to pull the 20% top-easiest words in the frequency list. Adjust the threshold up or down, depending on the student's starting point. The lower the number, the easier it will be for the student to identify the words. For example, if you want to limit your database pull to the simplest words in the database, then adjust the slider down. This threshold can be adjusted from 1% to 100%. Any

word that does not appear within the threshold will be assigned a failure symbol (❌) in the results.

- Frequency Percentile: Within the Frequency Threshold (say, 20%), words are ranked according to how often they appear in print. The Frequency Percentile sets the diction difficulty within the Frequency Threshold. Let's say that we want to pull only the top 1% of words from the database (the easiest 1%) for a beginning reader or new English speaker but would like for the complexity of the words in the sample text (input) to vary widely within that 1%. The Frequency Percentile could be set at 100% to allow a full spectrum of words within the Frequency Threshold to be analyzed without penalty (optionally, the check for whether a word is inside or outside the threshold can be turned off, which is discussed later).

❖ Customize: Difficulty Penalties:

Note that all difficulty penalties can be selected or deselected for inclusion in the diction analysis by checking/unchecking the checkbox next to "Check for This."

- Over Frequency Threshold: If a word or average exceeds the user-adjusted Frequency Threshold, a failure symbol (❌) will be displayed next to the word (in the case of a single word in the Results table), or, in the case of an AVERAGE, the THRESHOLD will be displayed in red characters (such as 20%). Words that exceed the threshold (that is, words that do not appear in the top X% of the words in the database) are heavily penalized, from 5 to 40 points (on a 100-point scale for total word difficulty).
- Number of Syllables: Polysyllabic words (such as *polysyllabic*) can pose a kind of reading difficulty that overlaps with word length. I don't know the degree of alignment between syllable count and word length. However, I might be able to apply Menzerath's Law to come up with a way to estimate the effect of word length separate from the effect of the number of syllables. For now, the software estimates the number of syllables (and overcounts with some words that have interposed silent letters, like the "e" in *notebook,* a word that the software currently says has three syllables). Polysyllabic words are penalized per syllable beyond the first syllable (the first syllable is free!) between 1 to 4 points per syllable.
- Difficult Spelling Patterns: This category of difficulty is evolving as I study the reading behavior of my student. For now, these problematic patterns are: "ght", "igh", "eigh", "gh", and "wh" (apologies for the British punctuation). A word with one of these spelling patterns is penalized from 1 to 4 points per instance.
- Diphthongs: From AI: "A diphthong is a complex vowel sound or "gliding vowel" where the mouth shifts from one vowel position to another entirely within a single syllable, such as the sounds in *boy, out,* and *day."* For now, I include only "au", "ou", and "oi" (Jdog doesn't seem to have a problem with "oy" or "ay"). A word with a diphthong is penalized from 1 to 4 points per instance.
- Triphthongs: From AI: "A triphthong is a phonetic term for a smooth, rapid glide through three vowel sounds within a single syllable. It occurs when a closing diphthong transitions into a neutral schwa (ə) sound. For instance, the word 'fire' transforms rapidly from /aɪ/ to /ə/." For now, I include only "eau", "our", "air", "ear". A word with a triphthong is penalized from 1 to 4 points per instance.
- Contractions: This software deletes characters that are non-letters but spares two: the space and the apostrophe. The spaces define the word boundaries, and an apostrophe is a sign

that the word may be a contraction, but it also may be, for example, a possessive ("I don't know about Frank's car" would be flagged as having two contractions). This will be corrected in future versions. For now, any word that includes an apostrophe is penalized from 1 to 4 points per instance.

Buttons

- **Submit**: Submits the user-provided input for the software to analyze.
- **Reset Sliders**: Resets all controls to their default settings.
- **Load Custom Settings**: Users may recall custom settings.
- **Save Settings**: Users may save custom settings.

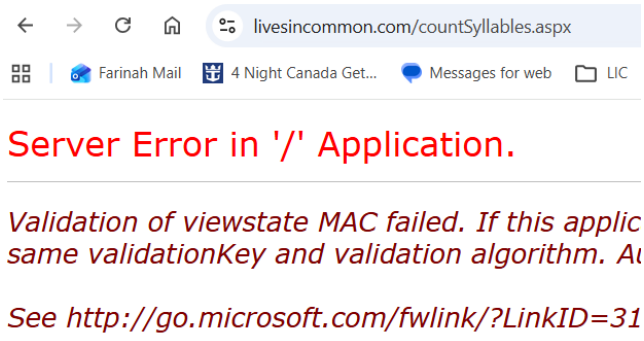
Tasks

Thank you for completing the following ten tasks (A through J). Please record your findings and comments in the form shown in Appendix B. If you need assistance during testing, please contact Bradford at bconnatser1245@gmail.com or 541-430-4548. Your ten tasks are described below (items A through J):

A. Submit several words for analysis with default settings:
   1. Browse to the following web address: https://livesincommon.com/countsyllables.aspx

---

NOTE AGAIN! If you get the server error shown below at any point during testing, then open the link in a new browser tab.



---

   2. As shown in Figure 6 (page 6), in the textbox under "Enter Words" near the top of the screen, type in: When she was a kitten, she crawled between the cushions on the sofa.
   3. Scroll down to the four buttons below the slider controls and click on the "Reset Sliders" button to set all controls to their default settings.
   4. Click on the "Submit" button to analyze the words.

B. Compare your results with my results:
   1. Compare your results with the results shown in Figure 7.
   2. Record any discrepancies in the "Alpha Tester's Notes" in Appendix B.

Results

Provided Input: When she was a kitten she crawled between the cushions on the sofa

Word	Frequency Threshold	Frequency Percentile	Syllables	Difficulty (1-100)	Cautions
when	✔ @ 20%	1%	1	7	Contains a difficult spelling pattern: ght|igh|eigh|gh|wh.
she	✔ @ 20%	1%	1	1	
was	✔ @ 20%	1%	1	1	
a	✔ @ 20%	1%	1	1	
kitten	✔ @ 20%	22%	2	7	Has **two** syllables.
she	✔ @ 20%	1%	1	1	
crawled	✔ @ 20%	55%	2	7	Has **two** syllables.
between	✔ @ 20%	1%	2	7	Has **two** syllables.
the	✔ @ 20%	1%	1	1	
cushions	✔ @ 20%	28%	2	7	Has **two** syllables.
on	✔ @ 20%	1%	1	1	
the	✔ @ 20%	1%	1	1	
sofa	✔ @ 20%	14%	2	7	Has **two** syllables.
AVERAGES (13 Words)	**0.0%**	**9.8%**	**1.4**	**3.8**	
THRESHOLDS	**20%**	**20%**	**1.5**	**10**	

Figure 7: Results of Diction Analysis with Frequency Threshold Set at 20%

3. Note for this and all other tests whether the meanings of the results are apparent. Did any word or average exceed a threshold? Does the color-coding help?
4. Click on any word in the "Word" column. Compare the resulting information popup to Figure 8. The features of this popup will be activated in the beta release (but are shown in the figure as design elements). In this popup, users can learn how a word is properly pronounced and syllabified, see the difficulty score and cautions for the word, access articles and exercises, invoke the word's definition, and view a list of scored synonyms for potential substitution.
5. Close the popup by clicking on the "Close" button at the bottom.

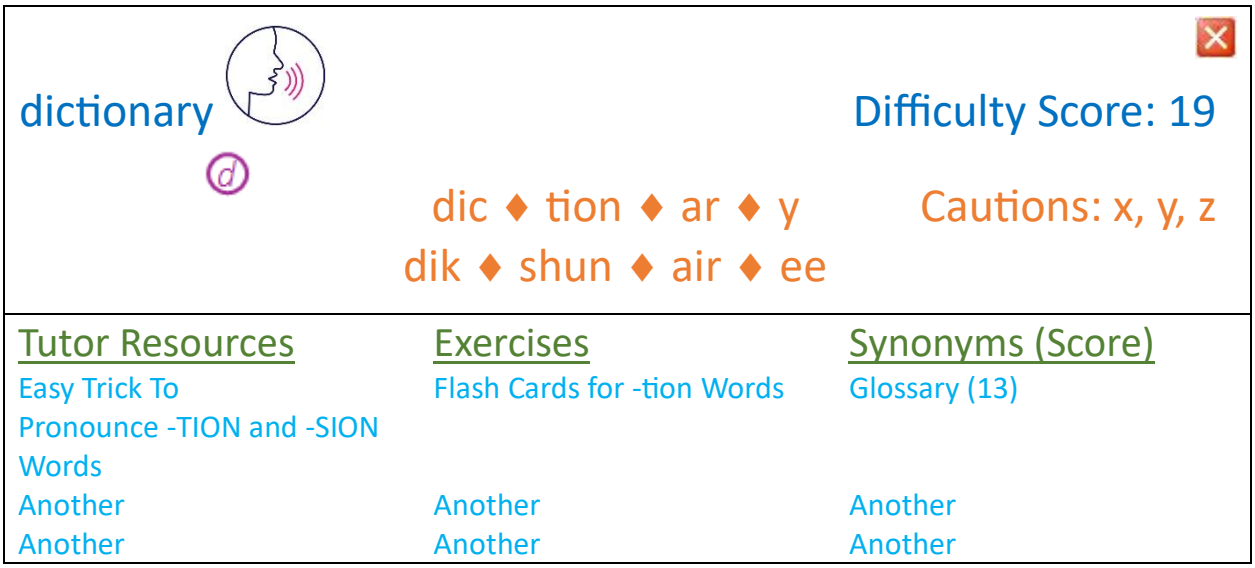


Figure 8: Information Popup for Clicked Word

C. Alter the Frequency Threshold:
   1. Change the Frequency Threshold slider to 2%, which will instruct the software to pull the top 2% of the most common words from the database.
   2. Click on the "Submit" button to analyze the words again, using the new setting.
   3. Review the Results. Note in Figure 9 that several words no longer meet the threshold (*kitten, crawled, cushions,* and *sofa*). Note that two thresholds in the AVERAGES row have now been exceeded and are highlighted in RED: Frequency Threshold and Difficulty.

Results

Provided Input: When she was a kitten she crawled between the cushions on the sofa

Word	Frequency Threshold	Frequency Percentile	Syllables	Difficulty (1-100)	Cautions
when	✔ @ 2%	2%	1	7	Contains a difficult spelling pattern: ght|igh|eigh|gh|wh.
she	✔ @ 2%	4%	1	1	
was	✔ @ 2%	1%	1	1	
a	✔ @ 2%	1%	1	1	
kitten	✘ @ 2%	0%	2	89.5	Is not in the top 2% of most frequent words.
she	✔ @ 2%	4%	1	1	
crawled	✘ @ 2%	0%	2	89.5	Is not in the top 2% of most frequent words.
between	✔ @ 2%	5%	2	5.5	Has **two** syllables.
the	✔ @ 2%	1%	1	1	
cushions	✘ @ 2%	0%	2	89.5	Is not in the top 2% of most frequent words.
on	✔ @ 2%	1%	1	1	
the	✔ @ 2%	1%	1	1	
sofa	✘ @ 2%	0%	2	89.5	Is not in the top 2% of most frequent words.
AVERAGES (13 Words)	**30.8%**	**1.5%**	**1.4**	**28.8**	
THRESHOLDS	**2%**	**20%**	**1.5**	**10**	

Figure 9: Results of Diction Analysis with Frequency Threshold Set at 2%

D. Alter the input and resubmit for analysis:
   1. Delete the current content of the “Enter Words:” textbox.
   2. Type in “the cat can't keep up.”
   3. Click on the “Submit” button to analyze the words again, using the 2% threshold and new input.
   4. Note in Figure 10 that all results are within the thresholds, but the analyzer detected a contraction (see “Is a contraction” in red under “Cautions”), which by itself earns a difficulty score of 16.

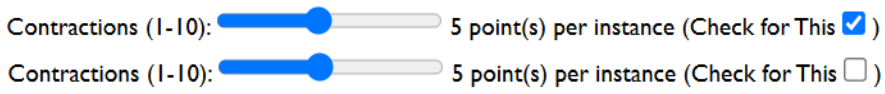
Results

Provided Input: the cat can't keep up

Word	Frequency Threshold	Frequency Percentile	Syllables	Difficulty (1-100)	Cautions
the	✔ @ 2%	1%	1	1	
cat	✔ @ 2%	27%	1	1	
can't	✔ @ 2%	2%	1	16	Is a contraction.
keep	✔ @ 2%	11%	1	1	
up	✔ @ 2%	2%	1	1	
AVERAGES (5 Words)	**0.0%**	**8.4%**	**1.0**	**4.0**	
THRESHOLDS	**2%**	**20%**	**1.5**	**10**	

Figure 10: Analysis of the New Words with Contraction-Checking

E. Turn off the Contractions checker:
   1. Deselect the "Check for This" checkbox next to the Contractions slider.



   2. Click on the "Submit" button to submit the text for analysis without checking for contractions.
   3. Note in Figure 11 that the results have no alerts (no font-coloring or Cautions).

Results

Provided Input: the cat can't keep up

Word	Frequency Threshold	Frequency Percentile	Syllables	Difficulty (1-100)	Cautions
the	✔ @ 2%	1%	1	1	
cat	✔ @ 2%	27%	1	1	
can't	✔ @ 2%	2%	1	1	
keep	✔ @ 2%	11%	1	1	
up	✔ @ 2%	2%	1	1	
AVERAGES (5 Words)	**0.0%**	**8.4%**	**1.0**	**1.0**	
THRESHOLDS	**2%**	**20%**	**1.5**	**10**	

Figure 10: Analysis of the New Words without Contraction Checking

F. Validate the functions of sliders and checkboxes:
   1. Close the browser.
   2. Open the browser and browse to the following web address: https://livesincommon.com/countsyllables.aspx

3. Validate that the following test text appears in the “Enter Words:” text box (if it does not, please copy it from below and paste it into the input textbox):

   Hello Dears isn't perspicacity asdfkasfdjhllkuy to all of our lovely gardeners and fresh fruit espresso and weight fighters notwithstanding the pressure fighters notebook aadgfsadfasdfkjsahdfy hdfjg kkhhhdf auyytgjdjhehg

4. Click on the “Submit” button.
5. Inspect the results.
6. Experiment with the software by changing the slider values and checkboxes and submitting the test text for analysis with these different settings.
7. Note how changing the settings changes the alerting on the page and the calculation of different penalties and threshold crossings.

G. Reset the page form: Click on the “Reset” button below the slider controls to reset the values of the sliders and checkboxes to their default settings. Verify that the settings for the thresholds and difficulty penalties match those shown in Figure 11.

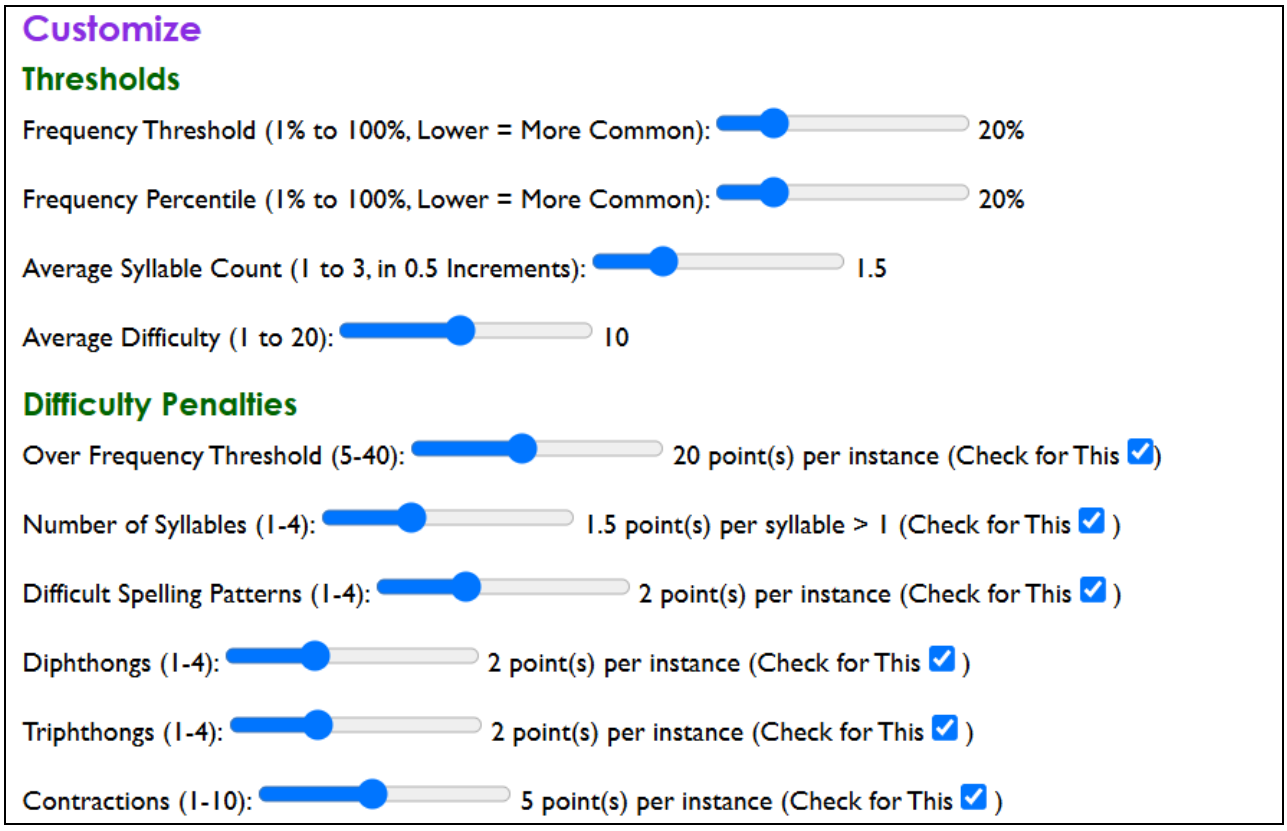


Figure 11: Default Settings

H. Save custom settings (will require user login in the future):
   Note that during alpha testing, all testers share the same account, so anything that you save to the database will be available to other testers. A future version will enable the user to delete accounts, students, and progresses.

1. Adjust the sliders and checkboxes to random values.
2. Click on the “Save Settings” button below the slider controls to save the current settings.

The popup window shown in Figure 12 will appear. The database is prepopulated with test students and their various progresses. You can select an existing student or create a new one.

---

3. Enter a new student name in the textbox after “Create New Student,” such as “Linda Popper.” However, please make up a unique name because other alpha testers will be saving settings to the database as well.
4. Enter a progress name in the textbox after “Progress Name,” such as “1-Test 1 for Alpha Test.”
5. Click on the “Save Settings” button to save your settings to the database. You should see a SUCCESS message.
6. Close this popup window.

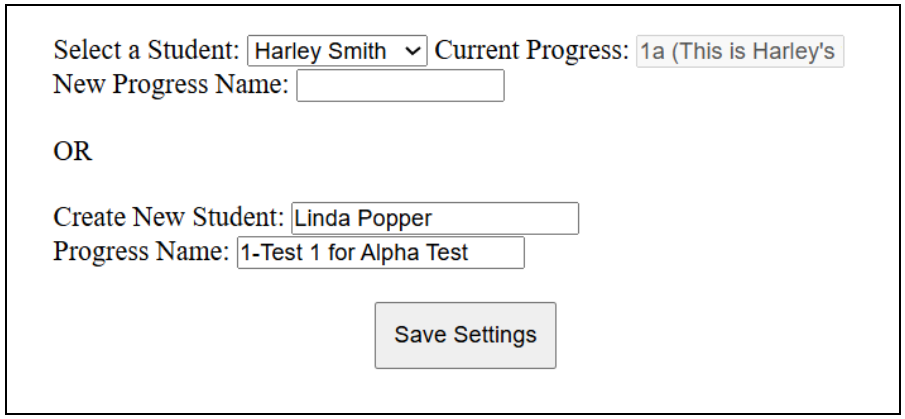


Figure 12: Input Form to Save Settings

I. Reset the page form: Click on the “Reset” button below the slider controls to reset the values of the sliders and checkboxes to their default settings.
J. Load custom settings (will require user login in the future):
    1. Click on the “Load Custom Settings” button below the slider controls to load your saved settings.

The popup window shown in Figure 13 will appear. Each small button shown after a student name is a link to saved settings.

---

2. Mouseover (rollover) the small buttons and observe the expanded text under “Progress Description.”
3. Mouseover the button next to the new student that you saved to the database and note the progress description: “1-Test 1 for Alpha Test.”
4. Click that button to populate the Diction Dog sliders and checkboxes with the saved settings.
5. Verify that the controls on the Diction Dog GUI reflect the settings that you saved.
6. Close this popup window.

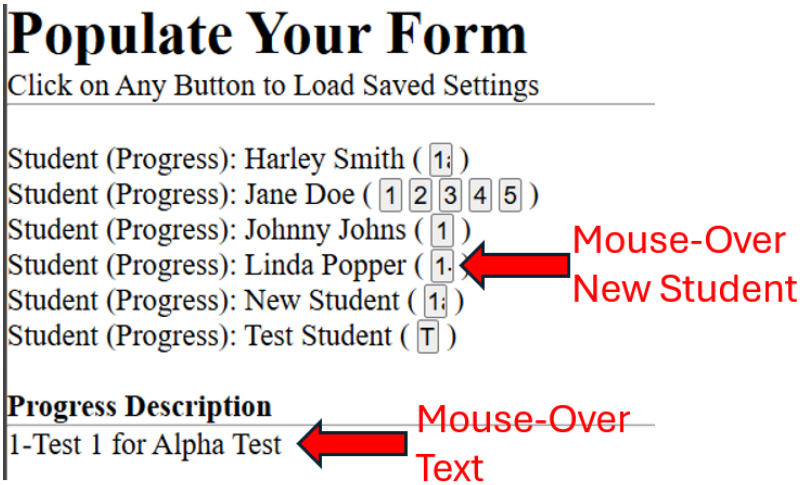


Figure 13: Load-Custom-Settings Form

Your Alpha Test Is Complete! Thank you!

Appendix A: A Story about Annabelle

Annabelle

Annabelle was a small black cat with a large curiosity.

When she was a kitten, she crawled between the cushions on the sofa.

She liked plastic bags.

In fact, she liked them a little too much.

When she was about a year old, she stuck her head into a plastic bag.

Perhaps she was looking for snacks.

However, the bag stuck on her head.

She tried to shake the bag off.

She shook her head a few times.

Then, she ran all around the house.

And ran.

And ran.

She ran down the stairs. Blind!

Annabelle smashed against the front door!

Then, she was on the ground.

Her parents heard the noise and ran to her.

She seemed confused as her parents removed the bag.

She blinked. That was a good sign.

Her parents rushed her to the vet.

The doctors checked her out.

Carefully.

Thank goodness!

Just a bump on the head!

Appendix B: Notes of the Alpha Tester

Please record the results of each test (as simple as saying “check,” but I invite you to elaborate at will).

Test	Note
A	
B	
C	
D	
E	
F	
G	
H	
I	
J	
Other Notes: Click or tap here to enter text.	