

Project - Milestone 2 Progress Report

Myles Lefkovitz

mlefkovitz@gatech.edu

Project Summary

The problem: How can we develop a gamified/intelligent tutoring system that effectively delivers language education that minimizes time spent learning while maximizing learning effectiveness and learning engagement?

Existing solutions: There are existing gamified ITS implementations for education. There are also existing language learning applications. However, there are no gamified ITS implementations for language learning.

Track: This is a **Development Track** problem.

The project: This project will develop a working prototype of a language learning application that implements best practices in language learning along with an intelligent tutoring system (to maximize learning effectiveness) and gamification (to maximize learning engagement).

Scheduled Activities

Scheduled Work:

Week 8

1. Research ITS and SRS platforms (4 hours)
2. Set up a local development environment as a fork of an open source SRS environment (4 hours)
3. Read through the documentation and add data to the tool (4 hours)

Week 9

1. Add initial learning material to the tool: add material across 3 categories (3 types of right/wrong pairs with corresponding mnemonics) (5 hours)
2. Test the software (2 hours)
3. Document the implementation plan for a selected ITS open source platform (4 hours)

Week 10

1. Implement the ITS within the existing flashcard system (6 hours)
2. Create students (0.5 hours)
3. Create training plans for students (1.5 hours)
4. Prepare Milestone 1 Progress Report with low-fidelity prototypes (drawn diagrams of the intended user interfaces and interactions) (2 hours)

Week 11

1. Implement the students as users in the system (1.5 hours)
2. Test that the student model of the ITS is working properly (1 hour)
3. Design and implement the ITS problem-solving suggestions feature (8 hours)

Week 12

1. Order learning material, develop ITS dependency feature, and test the feature (10 hours)

Week 13

1. Improve the user interface and all user interactions by fully implementing the ITS features developed in previous weeks and deactivating most features not relevant to the current tool while testing and resolving bugs (8 hours)
2. Identify potential beta testers and recruit testing participants (1 hour)
3. Publish beta test-ready application (2 hours)

Successfully Completed Work:

As discussed with my mentor, I ran into a number of implementation issues throughout the course of this project. I have worked through most of them, but am behind vs the scheduled work efforts and have had to reduce the scope of work.

Thus far, I have successfully implemented a learning (SRS) platform. I selected “Learn With Text”. Learn With Text is written in PHP with some components in JavaScript and a MySQL backend to store the data.

After researching various open source intelligent tutoring systems (ITS), I was unable to find a suitable one to implement so opted to develop a basic one on my own. I have only been able to implement the most basic features thus far. The ITS implementation is described in detail below.

Work not completed:

Given the difficulties I’ve run into, I’ve been unable to complete a beta-ready application as planned. I have some components working at this stage, but they are not ready for testing (and I don’t have a way to make this application available to users who aren’t within arms reach of my physical machine).

Lessons Learned:

It’s apparently very challenging to run some applications from source on Windows, especially if these applications run on multiple platforms with very specific dependencies and if most of the setup instructions are designed for Linux/Mac.

In addition to the challenge of setting up a single tool on Windows, it’s a major challenge to integrate tools written in different languages with different frameworks.

ITS Implementation

Data Model Changes

Original Data Model:

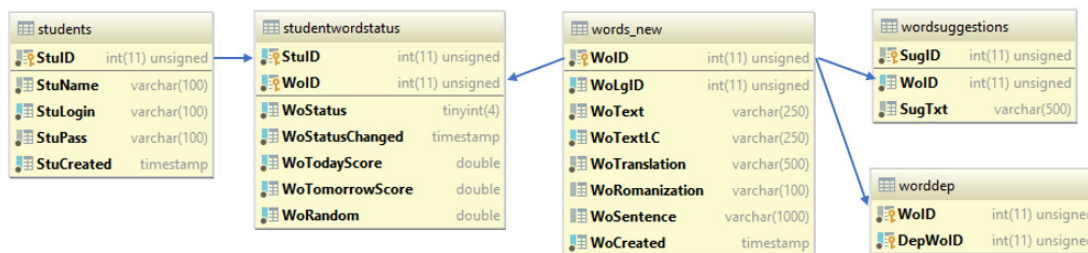
Originally, the language learning application did not have a system for configuring users. The entire domain model could be simplified to a single table (there are other tables, but they are somewhat extraneous, though they're used), shown below.

words	
WoID	int(11) unsigned
WoLglID	int(11) unsigned
WoText	varchar(250)
WoTextLC	varchar(250)
WoStatus	tinyint(4)
WoTranslation	varchar(500)
WoRomanization	varchar(100)
WoSentence	varchar(1000)
WoCreated	timestamp
WoStatusChanged	timestamp
WoTodayScore	double
WoTomorrowScore	double
WoRandom	double

Each record in the *words* table stores a single word (a unit of learning) along with relevant word-level information, like its translation, an example sentence, and when the word was added to the database. It also stores learning information like what the 'status' of the word is (when using the software students learn words and score them – a status of 0 represents an unlearned word and a score of 99 represents a mastered word) and the most recent date of score change.

Updated Data Model:

A functional intelligent tutoring system needs a **student model**, a **domain model**, and a **teaching model**. These pieces are all represented in the updated data model shown in the diagram below:



The Student Model:

The *student* table stores information about each student.

The *studentwordstatus* table stores the learning-related information for each student and word combination (including 'status' described above).

The Domain Model:

The *words_new* table stores only word-related information.

The Teaching Model:

The *wordsuggestions* table stores suggestions for each word. These suggestions are presented when mistakes are made within the user interface.

The *worddep* table stores word dependencies. These dependencies are used to prohibit challenging words from being displayed until component words have been learned.

User Interface and Functionality Changes:

A user selection system was added to the user interface. A simple drop-down box is used to select the current user – this is not secure, but for early development it should be sufficient.

The original lesson system was adapted to use the new data model.

The basic ITS features described above were implemented. When a student doesn't know a word a random suggestion from the *wordsuggestions* table is given, if available. Words in the *worddep* table are locked for learning until all of the independent words have been learned.

Appendices:

The information below appears unchanged from the progress report at Milestone 1.

Appendix 1: Training Plans

Below I detail the (simple) training plans I've created for 3 students. The language content added to the system is in Japanese as described above.

Student 1:

This student gets the most basic hiragana characters あ、い、う、え、お (a, i, u, e, o) correct, but is incorrect on the next set of hiragana characters: か、き、く、け、こ (ka, ki, ku, ke, ko).

The idea is that the ITS should suggest mnemonics for each of the incorrectly answered characters (I'll have to make them up).

Student 2:

This student gets all of the katakana characters correct except for one: メ (me)

The student then tries to learn words spelled in katakana that do not contain the character. The system should not interfere.

The student then tries to learn a word spelled in katakana アメリカ (America) that does contain the character. If the ITS's dependency-based teaching system is implemented correctly, the intelligent teacher should prevent or warn the student against learning more complex words that build upon fundamental learnings that the student hasn't mastered.

Student 3:

This student gets all of the hiragana characters correct. This student then tries to learn kanji characters. The student gets some kanji characters incorrect. The ITS should suggest mnemonics for the incorrectly answered kanji characters.

The student then tries to learn words using the kanji characters. The ITS should warn/prevent the student from learning these words before the kanji characters have been mastered.

Appendix 2: User Flow Diagrams

Below are 5 low-fidelity prototype design pages indicating the design of the application.

Interaction with the application begins with the “Home Page” below:

Home Page

The Home Page interface includes a top navigation bar with 'Logo' and 'Home' on the left, and 'Student 1' and 'Sign Out' on the right. Below the navigation bar, there is a 'Language:' label followed by a dropdown menu currently set to 'Japanese'. Underneath the language selector, there are three blue links: 'My Statistics', 'Learn New Words', and 'Practice Learned Words'.

From this page, the user is able to select their language (for learning, the menu language is always English). Log out (or in, if not logged in). View their user statistics (via “My Statistics”), learn new words (via “Learn New Words”), or practice learned words (via “Practice Learned Words”).

The “Learn New Words Page” below is accessed by clicking “Learn New Words” on the home page above.






Learn New Words Page

The Learn New Words Page features a top navigation bar with 'Logo' and 'Learn' on the left, and 'Student 1' and 'Sign Out' on the right. The main content area displays a list of learning categories with their respective progress: 'Hiragana' (5 of 46 learned), 'Katakana' (0 of 46 learned), 'Kanji' (0 of 2,000 learned), 'Words' (0 of 5,000 learned), and 'Sentences' (0 of 5,000 learned). The category names are in blue, while the progress text is in grey.

On this page the user can select the category of words they want to learn (available categories are shown in blue and unavailable categories are shown in grey) or return to the home page by clicking the logo in the top left.

Clicking one of the blue links in the above page brings up the second layer of the “Learn New Words Page”.

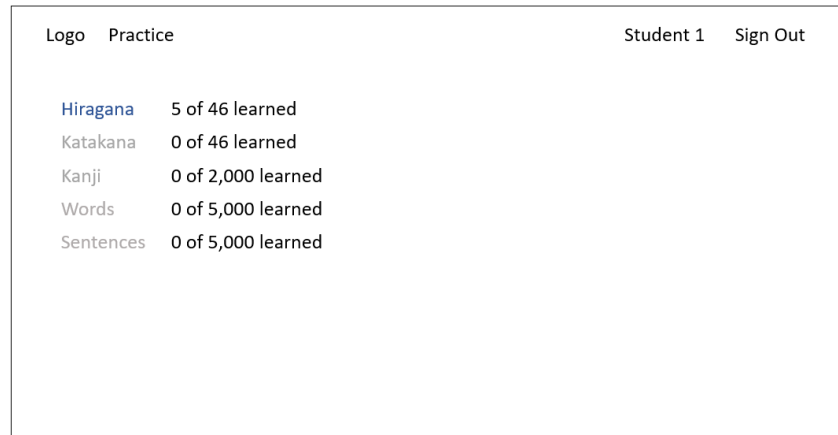
Learn New Words Page

Logo	Learn	Student 1	Sign Out
Hiragana	5 of 46 learned		
Character	Romanji	Description	Learned?
か	ka	Hiragana: ka	
き	ki	Hiragana: ki	
く	ku	Hiragana: ku	
け	ke	Hiragana: ke	
こ	ko	Hiragana: ko	

This page shows up to 5 characters for memorization. The user confirmed that they have memorized the word by clicking the button in the “Learned?” column. After all 5 buttons have been clicked, a new set of words appears. The user can exit the learning page at any time by clicking the logo in the top left to return to the home page.

The “Practice Learned Words” page below is accessed via the home page by clicking on the “Practice Learned Words” link.

Practice Learned Words Page



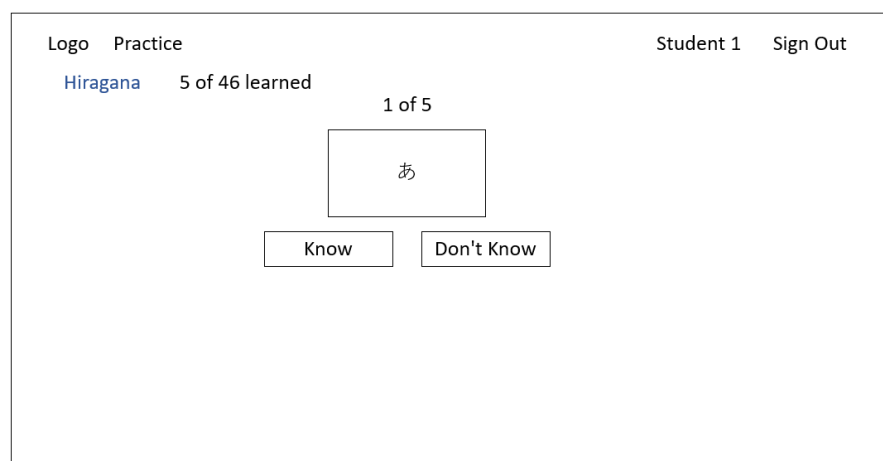
Logo Practice Student 1 Sign Out

Hiragana	5 of 46 learned
Katakana	0 of 46 learned
Kanji	0 of 2,000 learned
Words	0 of 5,000 learned
Sentences	0 of 5,000 learned

On this page the user can select the category of words they want to practice (available categories are shown in blue and unavailable categories are shown in grey) or return to the home page by clicking the logo in the top left.

Clicking one of the blue links in the above page brings up the second layer of the “Practice Learned Words Page”.

Practice Learned Words Page



Logo Practice Student 1 Sign Out

Hiragana 5 of 46 learned

1 of 5

あ

Know

Don't Know

Practicing learned words allows the user to train their memory. A user is given a flash card containing a word in the category they are practicing and are asked if they know it. The user can see the definition and romanization of the word if they click 'Don't Know'. How frequently the 'Know' and 'Don't Know' button have been pressed affects the SRS algorithm, which determines the level of mastery the user has over a particular word and how frequently it is presented in practice.