eColi - Ziyad Hamed, Nafiyu Murtaza, Chloe Wong, Jayden Zhang
SoftDev
p01 - Spanish Studying Service™
Target Ship Date: 2024-12-17

REST APIs:

Merriam-Webster's Spanish-English Dictionary

- English to Spanish and Spanish to English dictionary
- Takes an English or Spanish word and identifies its part of speech, its translation, an example phrase, and other information
- Requires API key, 1000 requests per day

Unsplash

- Library of royalty-free photos
- Can search for images using a keyword and retrieve image descriptions
- Requires API key, 50 requests per hour

Datamuse

- Provides linguistic data for a given word or phrase, including synonyms, related adjectives, and a pronunciation guide
- Accepts constraints in query parameters, such as requirements for words to start with a specific letter or be of a certain character length
- Supports both the English and Spanish language
- Does not require API key, 100000 requests per day

Program Components:

Bootstrap:

Role: Frontend framework

- CSS library for easy website configuration, and UI implementation. Chosen as it offers a wide library of UI assets that are helpful for site creation and navigation. Dynamic grid-system which can be used for flashcards and term displays.

Flask Framework:

Role: Web server, frontend to backend connection

- The flask server will facilitate app navigation between pages, and serve dynamically coded elements. Displays language content, processes input

User Registration/Authentication:

Role: Allow users to create, sign in, and sign out of accounts

- “signup.html” - allows the user to create an account provided the username is unique and the password is acceptable. Upon creating an account data is rendered in the database. Users will be redirected to this page if no account is found in session.
- “signin.html” - allows users to sign in with previously created accounts

Home Page:
Role: Displays user’s most recent lesson, and lessons to complete. (Maybe a learning streak)

- “index.html” - The homepage will act as the connecting piece between different pages and mainly serve site navigation

Lesson / Study Module:
Role: Allow users to take Spanish lessons and practice Spanish through tests. Should populate “nextLesson.html” with the appropriate lesson from the SQL database. Serves lessons and flashcards.

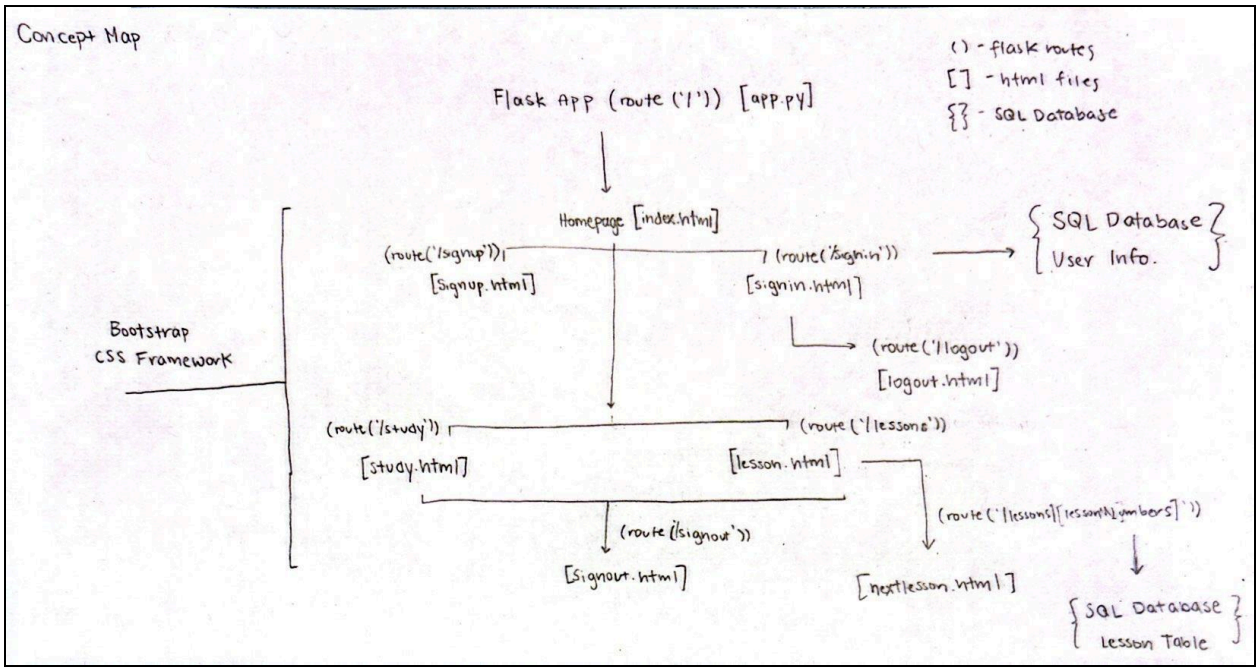
Frontend Templates (HTML/CSS):
Role: HTML pages to be displayed by flask middleware, CSS to style it

- signup.html
    - HTML page displaying forms for email and password to create a new account and store it in the database.
- signin.html
    - HTML page that retrieves information from the database to check whether or not an account is present from forms on the page. If so, the account will be retrieved and displayed on the homepage. Otherwise, the user will be redirected to the signup page.
- logout.html
    - HTML page that displays a form to logout of one’s account.
- index.html
    - HTML homepage for the user. Without an account, the homepage will contain a description of the website. With an account, the homepage will contain information on previous lessons completed, active lessons, test analytics, etc.
- study.html
    - HTML page containing a practice test that uses aforementioned APIs to generate randomized questions such as a translating a picture from unsplash API to Spanish or a random word in English to Spanish (creating from scratch a random word generator and inputting into the linguistics API). Each of these tests will be graded and displayed on the homepage.
- lesson.html

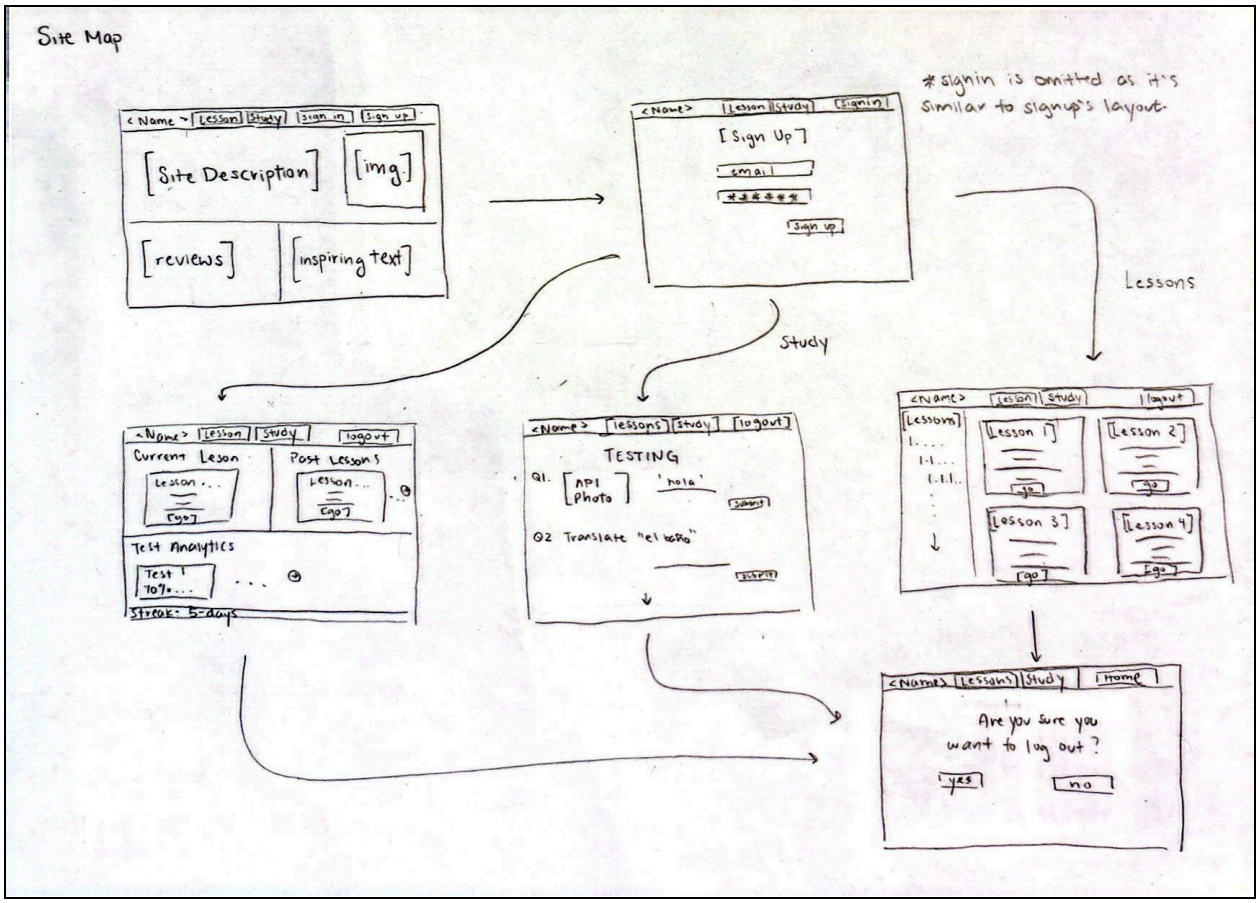
    - HTML page containing all of the lessons provided (retrieved from the database). Content will be **mostly pre-made**, with users being able to generate their own lessons to help other users.
- nextLesson.html (base template)
    - Will be the HTML lesson template in case a new lesson is added or a lesson needs to be revised. Otherwise, information from the SQL database is retrieved and displayed using this HTML page.
- “error.html”
    - Custom page for 404 and 500 errors, giving users feedback and allowing them to return to the homepage
    - To add in concept and site maps, arrows would point from error.html to index.html and from signup.html and signin.html to error.html

---

Concept Map:



Site Map:



SQLite3 Database:

Role: Storage for user accounts, lessons, and progress.

- Database will keep track of user accounts and passwords, and different lesson chapters.
- Stores information about any active lessons the user is taking, as well as, any past lessons.
- Stores data on previous tests that have been scored such as percentage and number of questions right/wrong.

SQLite3 Tables:

Main Components	**Example**
Users [via ID]	[UserID1, UserID2, UserID3]
Lessons [via ID]	[LessonID1, LessonID2, …]

The table above represents the main components within the website as it stores all of the users and lessons that have been curated already.

User	**Example**
Active Lessons [via ID]	[LessonID3]
Past Lessons [via ID]	[LessonID1, LessonID2]
Tests	[TestID1, TestID2]
Email	“pleasegiveusa100@bmcc.edu”
Password	********
Name	“Mr. Ducky”
Login Timestamps (in Unix Epoch)	[1732713957, …]

The User table above stores information on the user, such as the amount of active lessons, past lessons, tests taken, email address, password, and name. This is important as the lessons and tests will be displayed on the home page. The login timestamps will be measured in unix epoch to understand the login patterns of the users (to assign a streak if daily logins are met). Each bracket ([]) represents a new table (e.g., a table for lesson IDs, userIDs, and test IDs), omitted for redundancy.

Lessons	**Example**
Title	“Lesson 1: Grammar”
Text	“This section deals with Spanish superlatives…”
Author	“Mr. Yapper”
Completed (%)	100

The Lessons table above represents how we plan to store the information in our lessons as well as allow users to create new lessons. All of this data will be sent to the user’s end to display as a functional lesson. The “completed” section will be null until the user opens it, which will help keep track of active vs past lessons on the user’s end.

Tests	**Example**

Questions	[QuestionID1, QuestionID2, …]
UserID	“UserID3”
Number of Correct Answers	4
Total Questions	50
ID	“TestID2”

The Tests table will be used primarily in the study section, storing information about its questions and the User that had prompted this test. It will also store information about the number of correct answers that the user has gotten. The ID of the User is used here as each test will be randomly generated for each user, meaning that no one test can be the same, thus why it’s not stored within the main components table.

Questions	**Example**
Text	“How would you say thank you very much in Spanish?”
Image Path	app/unsplash_images/ImageFile.jpg
ID	“QuestionID3”
Solution	“muchas gracias”

The Questions table will be used as a sub-table within the tests table as it stores and provides information on the questions that were generated for the user. It will store any image paths, any text, and the solution within an ID that can be retrieved by the test table and, ultimately, by the user on the home page. The image paths will lead to locally cached responses from the Unsplash API to reduce the number of calls needed and help avoid reaching the hourly request limit.

Delegation of Tasks:

- PM, Project Maintenance / Enforcer, Database Backend - Ziyad
- Flask Server and Python Middleware - Jayden, Chloe
- HTML/CSS Frontend - Nafiyu