CS3200 Final Report

Jonathan Chiu

Chi Zhang

Group Name: ChiuZhangChi

**Group Project: A bilingual Chinese and English Dictionary**

1. **README**

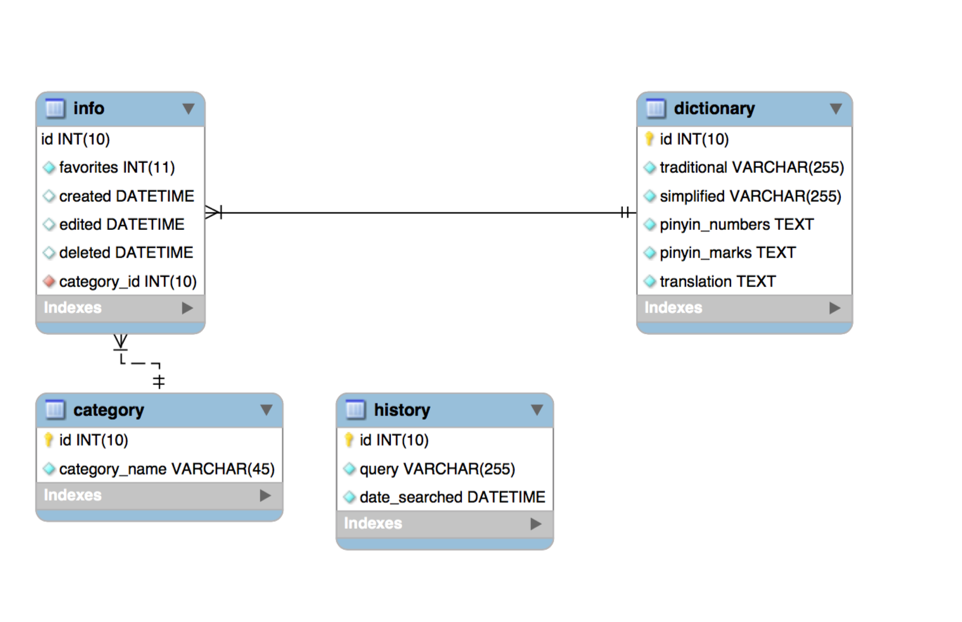
Provide a README for running the project. Specify all libraries, software, etc. needed to run the application. Specify expected installation directories.

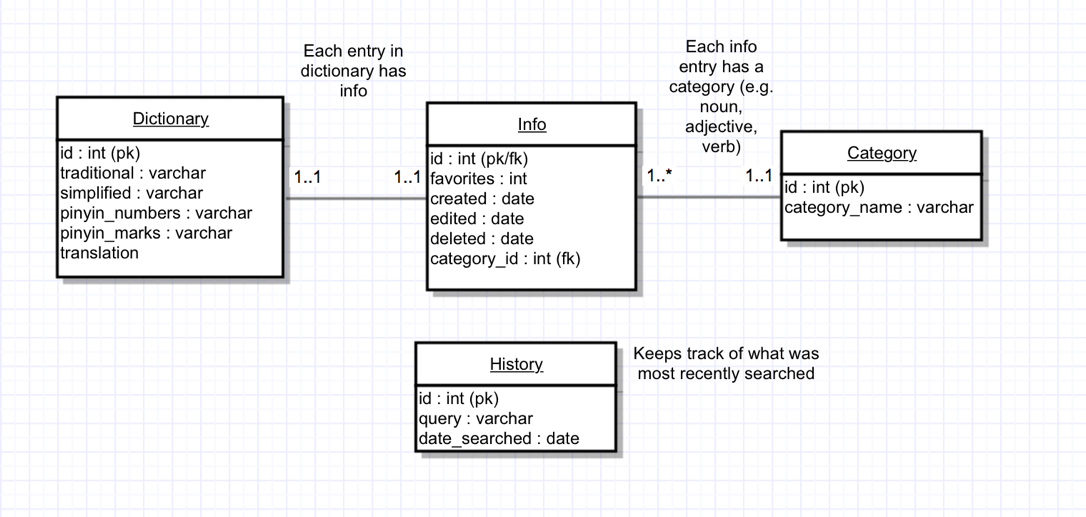
1. Make sure you have [nodejs](https://nodejs.org/en/) installed
2. By default npm will be installed as part of nodejs. You can check if you have node/npm installed by running `node –v` or `npm –v` in your terminal.
3. Run the included SQL install script by going to MySQLWorkBench, File 🡪 Run SQL Script (do not go another route that involves actually opening the file as your computer will experience painful lag)
4. Clone the repository and cd to the repository directory
5. Run `npm install` which will install all dependencies of this project, as specified in package.json
6. Run node index.js which will start the web server
7. Navigate to <http://localhost:3000>
8. Enjoy!

The above README (in prettier form) can be found at: <https://github.com/jonathanchiu/ZhongEngDict>

There you can clone or fork the directory

1. **UML and EER Mode**





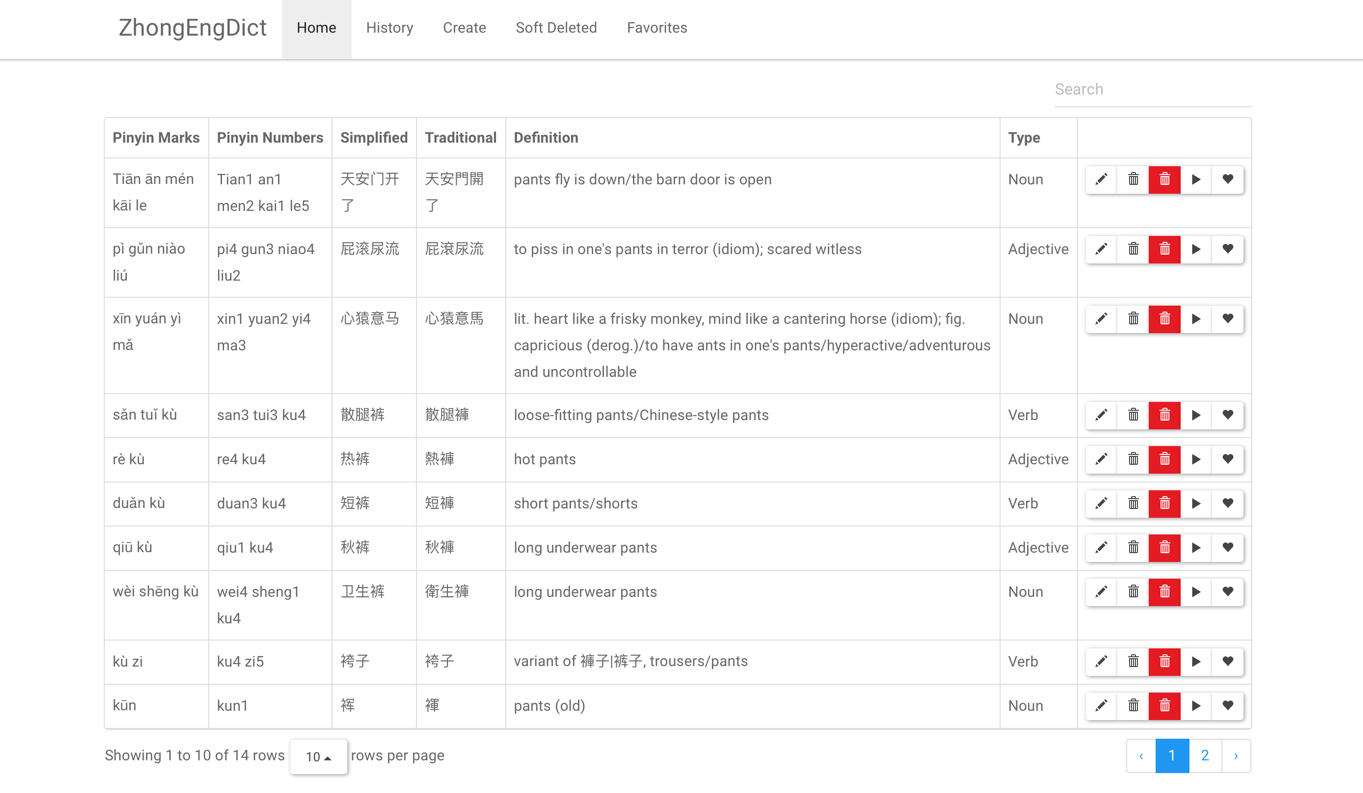
Provide the final user flow of the system. List the commands or method the users perform to interact with the system. How the structure works is that all words are contained in the dictionary table. This includes both the Chinese and English. Associated with each dictionary entry is an entry in info, which represents info such as creation date of the word or the word’s category (which can be one of noun, adjective, or verb). The history table simply keeps track of what users have previously searched for in the search bar.

1. **Instruction**

Our application is a web based bilingual dictionary. It allows the users to search Chinese pinyin equivalents, and traditional or simplified Chinese characters and get the English word translations, as well as to search English vocabularies and find the corresponding Chinese translations.

Open the website, users can choose to search Chinese or English by clicking the Simplified, Traditional or English Icon in the Home page.

Click the blue “search” button, the website will go to the result page.



Users can navigate to the next result pages by using the right bottom page navigation bar.

Users can update the vocabulary by clicking the “update” button 

Users can “soft” delete a vocabulary by clicking the “trash bin” button 

If soft deleted, users can navigate to the “Soft Deleted” link in the navigation bar to restore the words.

Users can “hard” delete a vocabulary by clicking the red “trash bin” button. There is no recovery option for this, so be sure you want to delete the entry.

Users can play the audio (will read the Chinese) for the vocabulary by clicking the “Play” button 

Users can put a vocabulary into the favorite list by clicking the “favorite” button 

**History Page:**

Users can view the most recently searched items by clicking the “History” link in the navigation bar

**Favorite Page:**

Users can favorite a vocabulary multiple times and users can check their favorite words in the “Favorites” page. The list of favorite words is in a descending order, so that the most favorite word is on the top.

**Deleted Page:**

Users can find all the deleted words in the “Deleted” page. If users want to restore a word, they can click the “restore” button.

**Create Page:**

Users can open the “Create” page to create new vocabulary and click the “ADD NEW VOCABULARY” button to add the new word into the database.

1. **Lessons learned section**

One of the main lessons we learned from doing this project was regarding how online dictionaries are actually structured on the web. It was actually quite difficult for us to conceptualize and implement a robust and fully fleshed out schema for a dictionary because the concept of a dictionary is inherently simple and straightforward (from the limited knowledge we possess). Often times, it felt like we were sacrificing the core simplicity of a dictionary, with complicated tables and schemas. Perhaps a NoSQL/non-relational approach would have been better for the purposes of this project, instead of attempting to force our dataset into a relational model.

1. **Future Work**

One aspect of the project we would like to work on fixing is the category/type for each word. Unfortunately, the third-party dataset we utilized didn’t include whether a particular word was classified as a noun, adjective or verb. For the purposes of having more data and being able to display it, we simply randomly assigned a category to every word in our database. This is obviously not correct, so in the future, we would like to find a clever and logical way to programmatically (and correctly) assign a category to every word in the database.

Another aspect is our semi working favorites button. Currently, a user can navigate to the site and theoretically click the favorites button of a particular word infinitely, potentially crashing the web server and database server. A better approach perhaps would’ve been to limit the number of clicks or even better, implement the concept of user accounts and create a favorites table to keep track of favorites by user accounts.