ECS 162
Web Programming
too short!
Question 7

☐ Where do we display the data returned from the camera?
Question 7

1 // onclick for button
2 function buttonAction {
3     let cameraData = none;
4     console.log("Beginning download");
5     beginCameraDownload(cameraData, function () {
6         console.log("In anonymous callback function");
7     })
8 }
9 }
10 }

- Where do we display the data returned from the camera?

After line 6. You don’t have the data until the anonymous callback function runs.
System design

- Browser only communicates with Server (via AJAX)
- Server communicates with Database and API.
Translation

User types on left, hits return. Expects to see translation. What happens next?
System design

Database

Server

Cloud Translate API

Browser
Doing a translation

- Browser sends AJAX request to server with English text.
- Server sends HTTP request with English text to CT API, waits for response.
- When API response comes back, Server sends response to Browser.
See database cartoon
A database is
- A file, or collection of files, storing data, stored on some server’s disks
- Could be app’s server (will be, for us), or could be another server on the internet
- Software for interfacing to that data
Databases

- Normal files on disks and (when not in a database) are read from beginning to end.
- Say we’re looking for something near the end, or in the middle, of a really big file. Takes a long time!
- Databases have an index to help you find things quickly.
- Some kind data structure, or collection of data structures.
A big database

Index

database
server
memory

database
server
disk
We’ll use SQLite

- SQLite is a simple database, supporting the SQL query language, and accessible through many APIs (C, Python, command line etc)
- sqlite3 is the Node module interfacing to SQLite
- SQLite saves data in a single file, right in our server directory. Not suitable for big projects!
- Although the database is a single file, SQLite uses fancy file access techniques to pick out records in the middle, using an index
SQL

- SQL is one, ancient kind of database interface
- It’s so standard that databases that do not use SQL are called NoSQL
- SQL is a declarative language – you specify what you want, not what computations the database should use to do it
- Let the clever database programmers figure out the best way to do what you want
- Loose standard, many variants
SQL Database

- An SQL database is made up of tables
- A table is similar to a spreadsheet
- Columns can contain arrays or strings as well as numbers
- Our 5-column table

<table>
<thead>
<tr>
<th>Rowid</th>
<th>User</th>
<th>English</th>
<th>Korean</th>
<th>times seen</th>
<th>times correct</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>Excuse me…</td>
<td>실례합니다…</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>Where is the train station?</td>
<td>기차역은 어디 있습니까?</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
For now...

- We’ll add cards to the database in this assignment.
- User can always be 1, seen and correct can always be 0.
- Later, we’ll have multiple users.
- Later, we’ll keep track of how many times each card was seen, how many times the user got it right, and use those to decide which cards to show.
Database set up

- Before we can store any cards, we need to set up the database
- Tell SQLite3 things like file name, column names, data types
- When do we do this?
Database set up

- Before we can store any cards, we need to set up the database.
- Tell SQLite3 things like file name, column names, data types.
- When do we do this?

We need to set up the database **once**, before the server runs. Data in db file remains on disk, even when server is not running.

Database setup is it’s own program, not part of server.
Database set up

- Install sqlite3 (npm install sqlite3)
- Require it at the top of the file (call it sql)
- Open a database file:
  ```javascript
  let dbFile = 'flashcards.db';
  let db = new sql.Database(dbFile);
  ```
- db is now an object that has methods for running SQL commands.
- Stuff written to the database by our program will be stored in the file flashcards.db
Making a table

- Our database will contain one table
- SQL CREATE command sets up a table, defines its columns:

```sql
CREATE TABLE flashcards (
    user INT, english TEXT, korean TEXT, seen INT, correct INT
)
```

- This has just five columns. What happened to that ID column?
Default column “rowid”

- Always present
- Always a unique integer id identifying the row
- By default, it is the PRIMARY key, meaning that it is the fastest way to get to one particular row
In node.js, we put the command into a string, and we pass the string to the db object:

```javascript
const cmdStr = 'CREATE TABLE flashcards (user INT, english TEXT, korean TEXT, seen INT, correct INT)';
db.run(cmdStr, tableCreationCallback);
```

The SQL string contains no newlines, and it is in single quotes since SQL uses double quotes.
Callback function

- Every time we send an SQL command operation to the database, we specify a callback function, even if we are not expecting a response.

- We check for errors in the callback function. SQLite is not great about giving error messages so this is really helpful.

- Why do we need to use a callback function instead of waiting for command to return?
Actually, for table creation...

- …waiting for the command to complete would have been fine.
- But when the Server runs a database command, it’s important.
- He has to be ready to answer HTTP requests from its many clients (browsers). Can’t hang up for any reason.