Lecture 9: Take Apart Two Apps

Team: _____ _____ _____

Get started · everyone, on your own laptop

About 3 minutes

1. **Open your codespace.** Go to `github.com/codespaces` and click the one you made before class.
2. **Make a folder and download the notebook into it.** In the codespace's terminal (**Terminal → New Terminal** if none is open), which starts in the top folder of your repo, run these one at a time:

   ```
   mkdir -p lecture-activities/lecture-09
   cd lecture-activities/lecture-09
   curl -O https://course.aiprototypes.org/resources/files/engl90bo-fa26-lecture-9-activity-skeleton.ipynb
   ```

   `mkdir -p` makes both folders at once. To skip typing the URL: on the **Course Materials** page, right-click **Activity notebook → Copy Link**, and paste it after `curl -O `. Run `ls` to check the file is there.
3. **Start JupyterLab** from the top of your repo:

   ```
   cd ../..
   uvx jupyter lab
   ```

   When it prints a link starting `http://127.0.0.1:8888/lab?token=`, **Ctrl-click** it (**Cmd-click** on a Mac) to open JupyterLab in a new tab. If it asks for a token, paste everything after `token=` from that link. **Leave this terminal running**; click **+** in the terminal panel for another.
4. **Open the notebook.** In JupyterLab's file list on the left, double-click `lecture-activities`, then `lecture-09`, then `engl90bo-fa26-lecture-9-activity-skeleton.ipynb`. Run the setup cell (`%env B=…`) with **Shift-Enter**, then work down through the Toolbox.

Part 1 · Hyperschedule

Check the key on the board

1.1 What term? Term _____ Year _____ The response is in what format? _____

1.2 How many sections? _____

1.3 The first section Title _____ Seats _____ of _____

1.4 This class ENGR sections _____ ENGR 190BO: _____ of _____ seats filled. Room left? ☐ yes ☐ no

1.5 The download _____ MB, downloaded once. So your search runs on ☐ my laptop ☐ their server

This design would be a bad idea when _____

1.6 Two hostnames

Hostname	Its `server:` header says	It delivers the app's…
`hyperschedule.io`		☐ files ☐ data
`banana.hyperschedule.io`		☐ files ☐ data

1.7 Four status codes For the 401, what would your browser have sent that `curl` didn't? _____

Code	URL (the part after `$B`)	What the body said
200	`/v4/term/current`	the current term, as JSON
404		
401		
400		

Part 2 · Wordle

2.1 Find the puzzle Requests on page load: _____ Its fields: _____

The puzzle's URL pattern: _____

2.2 Who grades your guess? One guess makes _____ requests. Colored by ☐ my browser ☐ NYT's server

The evidence, from your 2.1 output: _____

2.3 Where's your game? Local Storage key: _____ State request's status: _____

Why that code? _____ Survives switching laptops: _____

Which part of Hyperschedule does this remind you of? _____

2.4 Two different 404s URL _____ body _____

URL _____ body _____

So how many programs sit between you and the puzzle? _____

2.5 Fill in the diagram Mark each Wordle box **O** (a command or devtools showed it) or **G** (you guessed).

Layer	**Hyperschedule** (from the video)	**Wordle** (you)	**O/G**
The pixels	React components + CSS		
Logic in the browser	search; guest schedule in Local Storage		
The network	`hyperschedule.io` → GitHub Pages `banana.…` → Cloudflare		
The API server	tinyhttp: `/v4/sections`, `/v4/user/`		
The database	MongoDB (we read the source)		
Below the bottom	the colleges' registrar, polled every 60 s		

Last question. What would it take to make cheating on Wordle impossible? Which boxes would have to change? Why do you think the NYT hasn't bothered?