CS 4220

- Current Trends in Web Design & Development -

Cydney Auman

AGENDA

01 APIs

02 Working with NPM

03 Command Line Apps

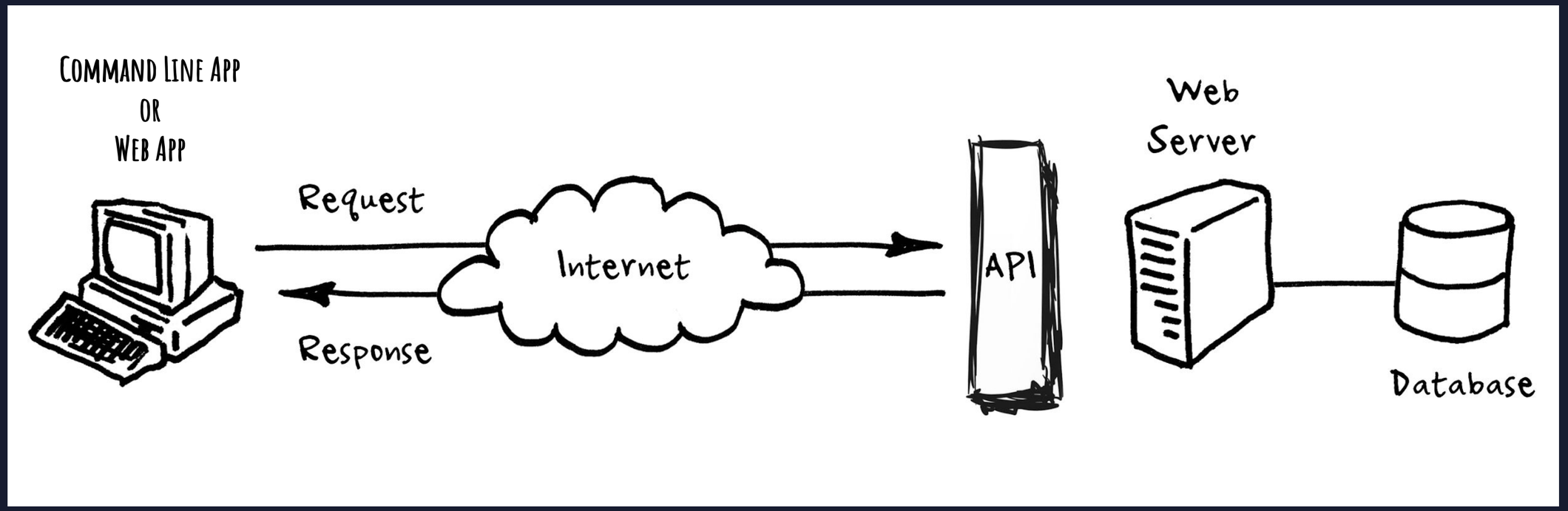
04 Code Demo

05 Midterm Discussion

06 Lab Time - HW Review & HW Week 7

What is an API?

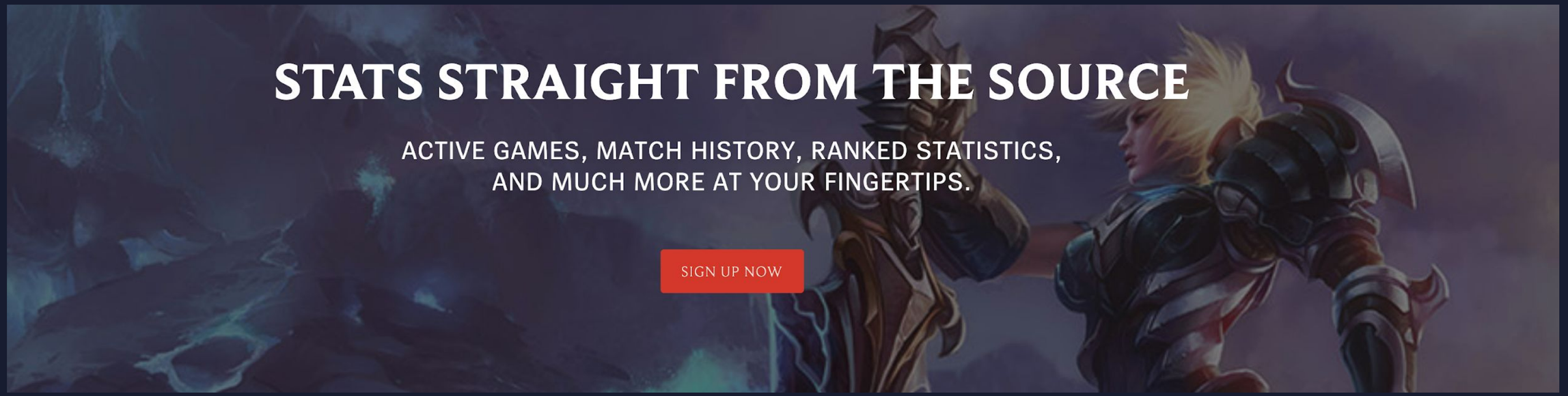
API (Application Programming Interfaces) are a large part of web applications. It is an intermediary software that allows applications to communicate with each other. Essentially, an API is the messenger that delivers your request to the provider of that data and then delivers the response back to you.



Why are APIs Important?

API orchestrate access to an application in order to allow access to some portion of a companies data.

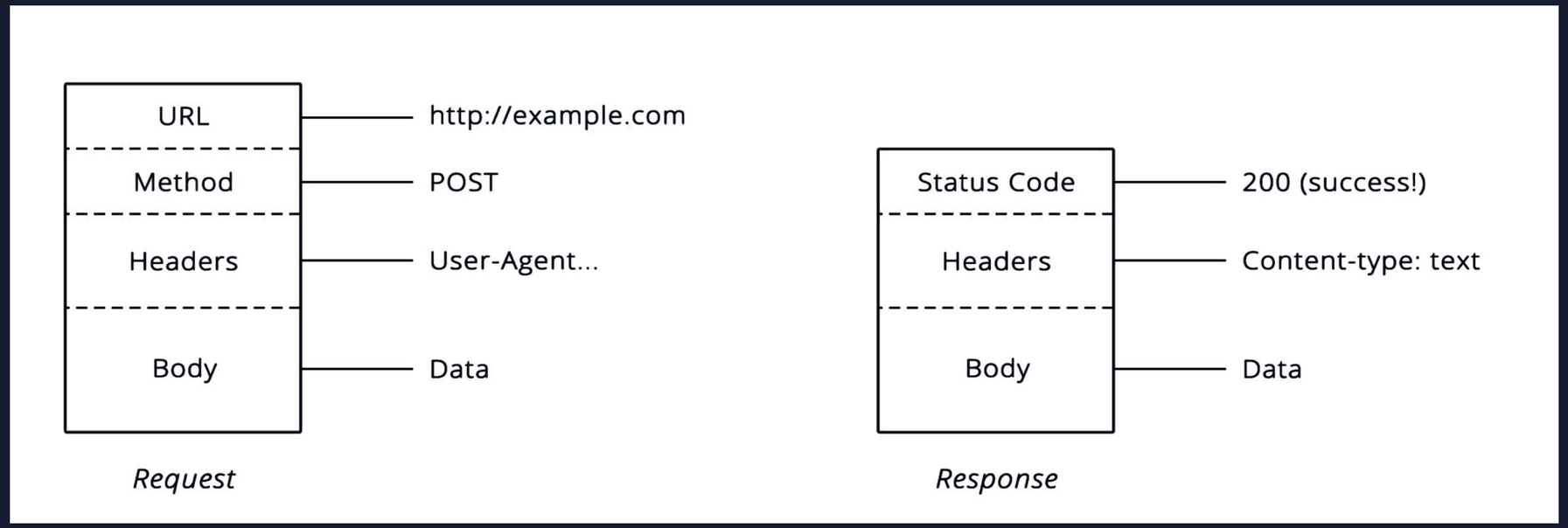
Large tech companies, including social media companies make some of their data available to the public. APIs are also built by government organizations, startups, fan sites, gaming and eSports companies and much more.



Communicating with APIs

Communication with APIs centers around of the HTTP Request-Response Cycle.

The application sends the API a **request** to do something. The API delivers the data to the provider and after it is processed sends the application a **response** saying whether or not the request could be completed.



Node Package Manager (NPM)

NPM is two things: **(1)** an online service registry where one can download/upload packages and **(2)** a CLI program (bundled with Node.js) that helps you install and manage them.

The online service portion of **NPM** is the repository of JavaScript modules, many of which are specifically written for Node.js.

When you installed Node on your computer, it came bundled with the **npm** command. This allows you to interact with this repository.

About NPM - https://docs.npmjs.com/about-npm/

NPM package.json

All npm packages and Node.js projects contain a file called **package.json.** This file holds various metadata relevant to the package and/or project.

The **package.json** is used to provide information such as the project’s name and most importantly the project's dependencies.

It can also contain other metadata such as a project description, the version of the project in a particular distribution, license information, even configuration data - all of which can be vital to both npm and to the end users of the package. The package.json file is normally located at the root directory.

NPM CLI Basics

`npm init`
- assist in creating the package.json.

`npm install`
- installs all modules in the package.json - installed in the node_modules directory.

`npm install --save <module-name>`
- installs the module by name and auto-magically adds it to package.json.

`npm uninstall --save <module-name>`
- removes the module by name and auto-magically removes it to package.json.

Node Modules

A collection of JavaScript libraries/packages. The largest repro in the world to find these is NPM.

`superagent`

- An HTTP module designed to be an easy way to make http calls while supporting many high-level HTTP features.

`yargs`

- Helps build interactive command line tools, by parsing arguments and generating an elegant user interface.

`inquirer`

- A collection of common interactive command line user interfaces. Eases the process of prompting questions and validating answers.

Command Line Interfaces

Command-line Interfaces (**CLIs**) or Command-line Applications, are programs designed to be used from the terminal/command-prompt. Command-line applications usually accept inputs as arguments. Additionally, they can accept options, often referred to as flags.

Examples:

- npm
- git (CLI for Github)
- dockly (CLI for docker)

```
$ git --help
usage: git [--version] [--help] [-C <path>] [-c <name>=<value>]
           [--exec-path[=<path>]] [--html-path] [--man-path] [--info-path]
           [-p | --paginate | -P | --no-pager] [--no-replace-objects] [--bare]
           [--git-dir=<path>] [--work-tree=<path>] [--namespace=<name>]
           <command> [<args>]

These are common Git commands used in various situations:

start a working area (see also: git help tutorial)
   clone      Clone a repository into a new directory
   init       Create an empty Git repository or reinitialize an existing one

work on the current change (see also: git help everyday)
   add        Add file contents to the index
   mv         Move or rename a file, a directory, or a symlink
   reset      Reset current HEAD to the specified state
   rm         Remove files from the working tree and from the index

examine the history and state (see also: git help revisions)
   bisect     Use binary search to find the commit that introduced a bug
   grep       Print lines matching a pattern
   log        Show commit logs
   show       Show various types of objects
   status     Show the working tree status

grow, mark and tweak your common history
   branch     List, create, or delete branches
   checkout    Switch branches or restore working tree files
   commit     Record changes to the repository
   diff       Show changes between commits, commit and working tree, etc
   merge      Join two or more development histories together
   rebase     Reapply commits on top of another base tip
   tag        Create, list, delete or verify a tag object signed with GPG

collaborate (see also: git help workflows)
   fetch      Download objects and refs from another repository
   pull       Fetch from and integrate with another repository or a local branch
   push       Update remote refs along with associated objects
```

```
console.log('Week 07');
console.log('Code Examples');
```

Lab, Homework and Prep

Lab Time

- Run the Card-App CLI
- Work on HW Week 7
- Review the Midterm Project and Requirements
- Begin Work on Midterm Project

Preparation for Next Week

- Watch YouTube Get Started with NPM (Slide 6)
- Optional Readings:

  https://docs.npmjs.com/files/package.json.html

  https://zapier.com/learn/apis/chapter-1-introduction-to-apis/