Open-Source Report

Proof of knowing your stuff in CSE312

Guidelines

Provided below is a template you must use to write your reports for your project.

Here are some things to note when working on your report, specifically about the **General Information & Licensing** section for each technology.

- **Code Repository**: Please link the code and not the documentation. If you'd like to refer to the documentation in the **Magic** section, you're more than welcome to, but we need to see the code you're referring to as well.
- License Type: Three letter acronym is fine.
- License Description: No need for the entire license here, just what separates it from the rest
- **License Restrictions**: What can you *not* do as a result of using this technology in your project? Some licenses prevent you from using the project for commercial use, for example.

Also, feel free to extend the cell of any section if you feel you need more room.

If there's anything we can clarify, please don't hesitate to reach out! You can reach us using the methods outlined on the course website or see us during our office hours.

Express.js

General Information & Licensing

Code Repository	https://github.com/expressjs/express
License Type	Unlimited Use License
License Description	(The MIT License)
	Copyright (c) 2009-2014 TJ Holowaychuk <tj@vision-media.ca> Copyright (c) 2013-2014 Roman Shtylman <shtylman+expressjs@gmail.com> Copyright (c) 2014-2015 Douglas Christopher Wilson <doug@somethingdoug.com></doug@somethingdoug.com></shtylman+expressjs@gmail.com></tj@vision-media.ca>
	Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the 'Software'), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge,

	publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions: The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software. THE SOFTWARE IS PROVIDED 'AS IS', WITHOUT
	WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
License Restrictions	No restrictions



- How does this technology do what it does? Please explain this in detail, starting from after the TCP socket is created
- Where is the specific code that does what you use the tech for? You must provide
 a link to the specific file in the repository for your tech with a line number or number
 range.
 - o If there is more than one step in the chain of calls (hint: there will be), you must provide links for the entire chain of calls from your code, to the library code that actually accomplishes the task for you.
 - Example: If you use an object of type HttpRequest in your code which contains the headers of the request, you must show exactly how that object parsed the original headers from the TCP socket. This will often involve tracing through multiple libraries and you must show the entire trace through all these libraries with links to all the involved code.
- Express js is used to create an instance of the application (line 10 in app.js in our code).
- We then initialize a http server that listens for requests (line 11 and 92 in app.js in our code). To do this express uses Node.js http library to create a server:
 - Repo/File Link: https://github.com/nodejs/node/blob/7984af69a090dd6d1f60ffe7e194d5e69 bce0c20/lib/http.js#L62
 - Line 62 Function.
- Express then listens on the specified port for requests. Using the connection listener in the http server file in node
 - Repo/File Link:
 - https://github.com/nodejs/node/blob/7984af69a090dd6d1f60ffe7e194d5e69 bce0c20/lib/ https://github.com/nodejs/node/blob/7984af69a090dd6d1f60ffe7e194d5e69 bce0c20/lib/ https://github.com/nodejs/node/blob/7984af69a090dd6d1f60ffe7e194d5e69 https://github.com/nodejs/node/blob/7984af69a090dd6d1f60ffe7e194d5e69">https://github.com/nodejs/node/blob/7984af69a090dd6d1f60ffe7e194d5e69 https://github.com/nodejs/node/blob/7984af69a090dd6d1f60ffe7e194d5e69 https://github.com/nodejs/node/blob/7984af69a090dd6d1f60ffe7e194d5e69 https://github.com/nodejs/node/blob/7984af69a090dd6d1f60ffe7e194d5e69 https://github.com/nodejs/node/blob/7984af69a090dd6d1f60ffe7e194d5e69 <a href="https://github.com/nodejs/node
 - Line 614 function listens for connections
- Once a TCP socket is created (request is made, lets say a GET request in this case) it triggers the incoming message function and reg creation
 - Repo/FIIe Link:
 - https://github.com/expressjs/express/blob/0debedf4f31bb20203da0534719 b9b10d6ac9a29/lib/request.is#L31
 - Line 31
 - And
 - https://github.com/nodejs/node/blob/7984af69a090dd6d1f60ffe7e194d5e69 bce0c20/lib/ http incoming.is#L53
 - Line 53
 - Both of these (req creation and incoming message function) work together
 to handle the request from the http server. The incoming message function
 takes the TCP socket as a parameter, once a request is received it uses
 incomingMessages's variables rawHeaders, kHeaders, kHeadersCount etc
 to keep track of the headers, which get set in the addHeaderLine(s)()
 functions.

- Once the request is handled by NodeJs ^ . Expresses reg variable in this:
 - var req = Object.create(http.IncomingMessage.prototype)
 - Repo/File Link:
 - https://github.com/expressjs/express/blob/0debedf4f31bb20203da0534719 b9b10d6ac9a29/lib/request.js#L31 Line 31
 - Then the req is parsed and saved into the req variable. The pathname is stored in the reg for express to define functionality for later.
 - The .IncomingMessage attached to the http is part of the http class that allows access to response headers, requests, and data
 - https://github.com/nodejs/node/blob/7984af69a090dd6d1f60ffe7e194d5e69bce0c20/doc/api/http.md#class-httpincomingmessage
 - Link to http.IncomingMessage class documentation ^

- COOKIES:

- We also used a cookie parser module which is a part of expressJs to parse the cookies once the request was made (LINE 5, 22 of app.ts in our code)
- https://github.com/expressjs/cookie-parser/tree/master
- This code allows us to use req.cookies when parsing the cookies for a given request.
- As stated in the readme this is an api for express and all it does is set cookies variable for req on a request.

