# Module 1 Project 2

Michael Grandori

Arizona State University

IFT: 458

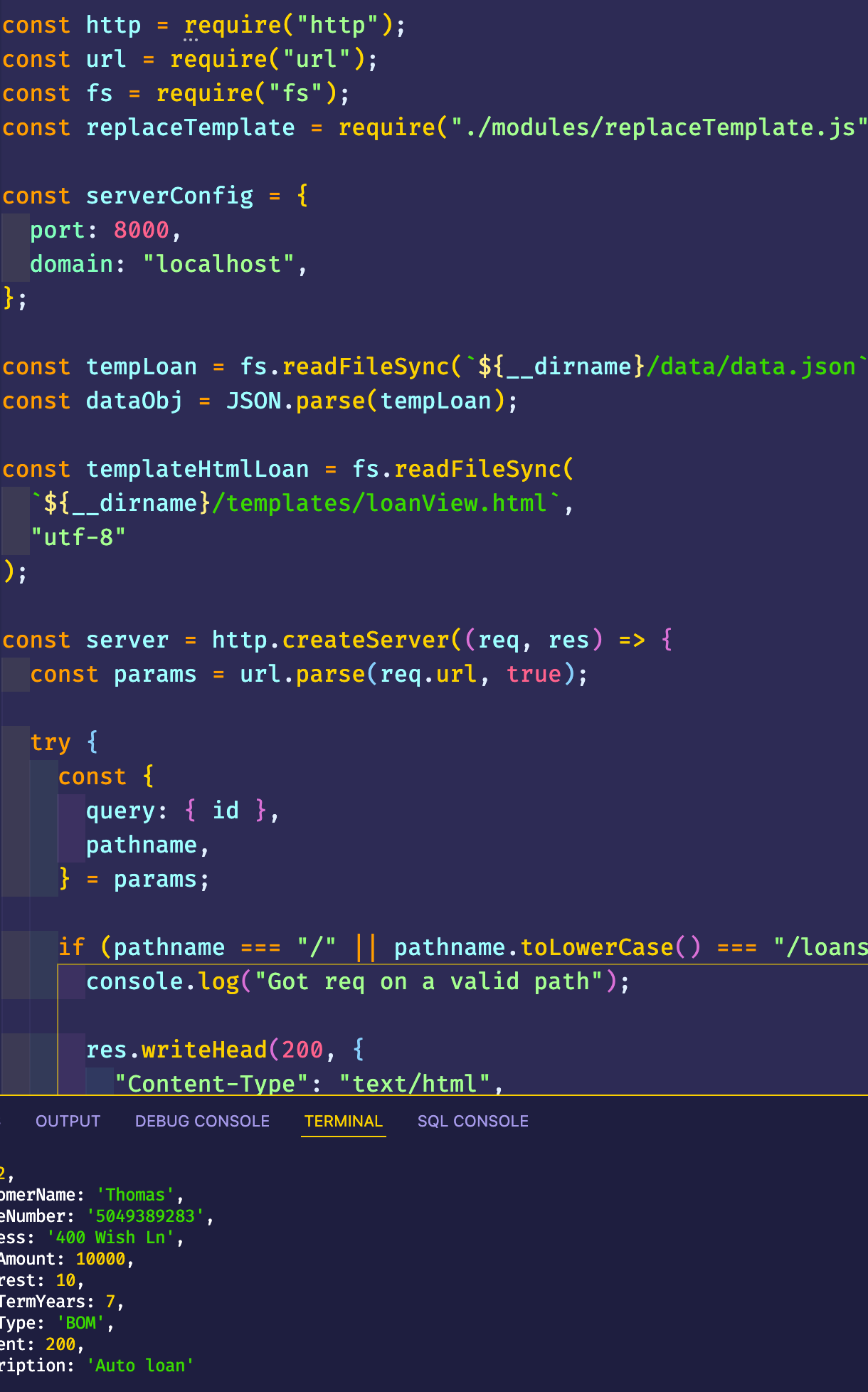
Professor Dinesh Sthapit

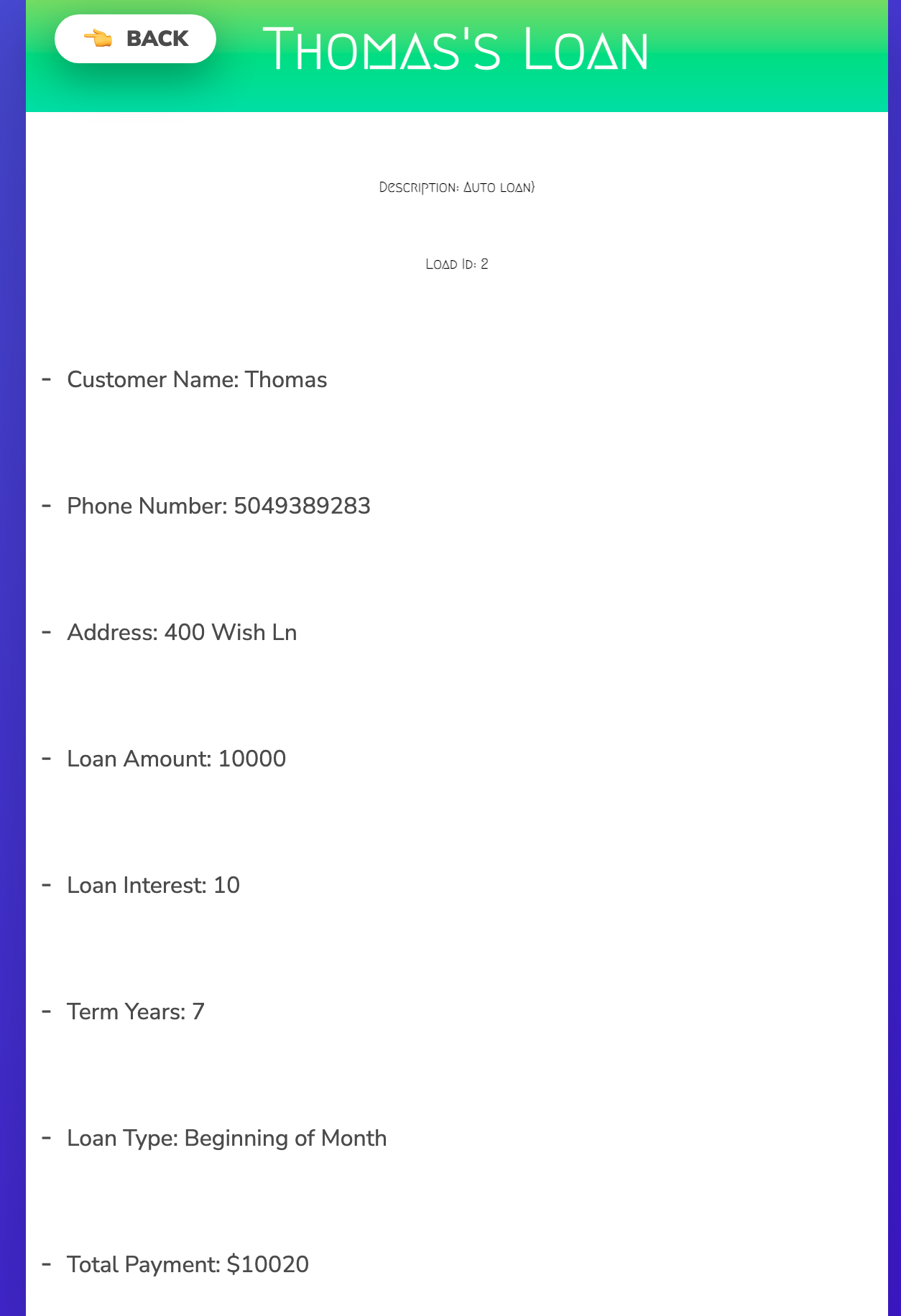
August 29, 2022

## Creating a Loan Calculator

Creating a loan calculator has several steps. The first thing we did was created a bunch of JSON objects that were in the defined model. (Note: I added “payment” in order to make the formula work in a way that I determined best for my imaginary customer’s needs after some research. Even if business requirements are not well defined, it is up to us as developers to make adjustments as necessary).

The web server is a simple Node server that will respond to requests that are corresponding to certain available paths such as `/` or ‘/loans’. It will parse an ID passed in via query params in order to return the correct server rendered HTML page to the requestor.



The view returned to the user will have information about the requested loan as well as the computed total loan amount to include interest over the term. This functionality is generated by a module that returns the total loan amount to the template so that it can be rendered appropriately. No calculations are done client side, it is all done on the Node server. This is beneficial because there is some calculations we would not want to tax the client side with. There are also calculations that may use private data that we would not want to expose to the client since the browser can see what is returned and this could be a security issue.

**Status Codes**

HTTP status codes are an important part of development since HTTP is a protocol that many client/server architectures use. The idea of a protocol is that there is a certain implicit understanding of how data should be shared between two or more systems. Status codes allow a contract between the server and the client that can relay what kind of response is being sent back without explicitly stating it or inventing a bunch of new ways to relay the same idea.

Generally speaking, anything under 400 is a non-error response. By a non-error response, I mean a response that is either relaying information (100-199), a success (200-299) or a redirect (300-399). It is important to note that any of these responses CAN be returned for any situation, even if there was a server error, and for that reason it is important to have a clear understanding of which status code you should return under certain circumstances.

Error responses are generally OVER 399. Client side errors live between 400-499 and server errors live between 500-599. There are many times where there can be a non-fatal error (one that is caught on the server) and therefore an appropriate response in the 400s is returned to the client to relay what went wrong. This can often be included along with a message with more information or a generated response to show to the client.

**HTTP Headers**

HTTP headers are another important part of the HTTP protocol. Headers can give more metadata information about a request such as the content type that is in the request body, the accept header that is what format you expect to get data back in, or the authorization header that may include a token to pass to the server to determine if you can access the requested resource. There are many headers that can pass information about many things that are less common as well, such as cache control or access control based.

Cookies are another huge part of headers. Cookies are bits of data passed between requests to be able to maintain state. Since HTTP is a stateless protocol, these cookies allow things like tokens, CSRF (Cross-Site Request Forgery) protection, or other bits of information about a session or user to be maintained without having to re-authenticate or revalidate on every request. They can also contain expiration dates, be tied to a single session in the browser, or be set to only be modified by the server instead of both the client and the server. Many applications use cookies today, and some in a rather nefarious way since they can track your movement between websites as long as your cookies are not cleared.

HTTP Headers also can return things such as content length (in bytes) and information about the browser and OS being used, commonly referred to as the User Agent. Between all of this vast functionality, headers are a very useful tool and are present on every single request that uses HTTP as a protocol.

**Resources**

*HTTP response status codes - HTTP: MDN*. HTTP | MDN. (n.d.). Retrieved August 29, 2022, from https://developer.mozilla.org/en-US/docs/Web/HTTP/Status

Archiveddocs. (n.d.). *HTTP response headers*. Microsoft Docs. Retrieved August 29, 2022, from https://docs.microsoft.com/en-us/previous-versions/windows/it-pro/windows-server-2012-R2-and-2012/hh831707(v=ws.11)