***Documentation Packet [ 17 20 12 24 ] Dec 20th 2024***

|  |  |
| --- | --- |
| Student Name: |  |
| Goals:  1. Update boilerplate for faster application creation 2. Understand custom modules 3. Practice using middleware | Events:  1. Keystones this week |
| Included Documentation  1. Updated Boilerplate 2. Job Openings 3. Reflections | Required Documentation:  1. Updated Boilerplate 2. Job Openings 3. Reflections |
| Changes/Notes: | |

# Updated Boilerplate

Create a new project for an updated boilerplate application. This will allow us to quickly start projects no matter what the application’s purpose is.

1. Import all require modules
2. Import each custom module (routes.js and socket.js. See below)
3. Create a database file in the /data folder
4. Create a db object by opening the database file
5. Create an app object from the express module and set the view engine to EJS

const app = express(); app.set('view engine', 'ejs');

1. Use the express-session module to create a session object

const session = require('express-session');

1. Use the connect-sqlite3 module to create an SQLiteStore object from the session object

const SQLiteStore = require('connect-sqlite3')(session);

1. Create a server object from the app’s listen() method

const server = app.listen(PORT, () => {});

1. Create an io object from the server object

const io = socketIo(server);

1. Create a sessionMiddleware object from the session and SQLiteStore objects

const sessionMiddleware = session({

store: new SQLiteStore,

secret: 'your\_secret\_key',

resave: false,

saveUninitialized: true,

cookie: { secure: false } // Set to true if using HTTPS

});

1. Configure the app object to use the sessionMiddleware object

app.use(sessionMiddleware);

1. Configure the io object to use the sessionMiddleware object as well

io.use((socket, next) => { sessionMiddleware(socket.request, {}, next); });

1. Create the following files in /views: index.js, login.js, chat.js
2. In a folder called “/modules”, create a file called “routes.js”. In this file:
   1. Create a function that can handle each of these:
      * GET - /
      * GET - /login
      * POST - /login
      * GET - /logout
      * GET - /chat
   2. Export each of the functions to be used in other modules
   3. Create an “isAuthenticated()” function to be used only in this module
3. In “/modules”, create a file called “socket.js”. In this file:
   1. Create a function that can handle a connection event for socket.io
   2. Export that function to be used in other modules.
4. Configure the app object to use each of the functions in /modules/route.js for the appropriate endpoint.

app.get('/', routes.index }); //Example

1. Configure the io object to use the function in /modules/socket.js for ‘connection’ events

io.on('connection', socketHandler.connection); //Example

1. Configure the app object to use a static folder for all requests not in the listed functions above.

app.use(express.static(path.join(\_\_dirname, 'public')));

Write your routes.js and appropriate templates to do the following:

1. The index displays your username if logged in with a “Chat” link to /chat, or a “Login” link to /login if your are not.
2. The login is a form to login using a username and hashed password stored in the database file, and a link to log in with formbar instead.
3. The chat connects to the websocket server and sends and receives one message.
   1. The server must show the received message and the client must show the message sent back, each displayed in their respective consoles.
   2. The /chat endpoint handler and the socket handler must both use session data to check that the user is logged in before being used. The server message must contain the user’s name from session data. If not logged in, redirect the user to the login page.
      * isAuthenticated() is used for this

When complete test to make sure you can do the following:

View the index when logged in/out as per step 19. Only view and send the chat page and message on it when logged in. Can log in and out with both the database and Formbar Oauth2.

# Reflection

**How is the pace / structure of the class overall so far?**

|  |
| --- |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |

**What was one mistake you made in school or otherwise that you can recognize? What can you do in the future to prevent it from happening again?**

|  |
| --- |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |

**How to you manage your time outside of school. How much of it is dedicated to advancing your career / skills / goals? If it is not a lot, why not?**

|  |
| --- |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |

**How difficult was the assignment this week? What made it difficult/easy for you?**

|  |
| --- |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |

# A picture containing text, monitor, screen, clipart Description automatically generatedGrading

|  |  |  |
| --- | --- | --- |
| 10 | You went above and beyond expectations. You applied knowledge that was not taught in this class in addition to what was taught. Additional rewards are given | * ***If the assignment does not have its own rubric, it will default to the rubric on the left.*** * All assignments start at 10/10 possible points * 1 point is deducted per infraction   + Lateness   + Mistakes   + Unprofessionalism   + Not following instructions * Outstanding submissions, or submissions on assignments not marked in “Required Documentation” can reward pogs |
| 10 | You performed as well as can be expected for this class. You show a complete understanding and made no mistakes. You have mastered the subject. |
| 8 | Assignment is complete. You show a good understanding of the subject, but there are mistakes or minor incorrect details. You are ready to move to new subjects. |
| 7 | You show and understanding of the subject, but there are serious errors, or there are pieces you can practically use without understanding them. Remediation needed. |
| 6 | Assignment is incomplete but/or you showed that you understand at least the fundamentals of the subject. Assignment is low effort. Serious need of remediation. |
| 5 | You show minimum effort, assignment is incomplete, or have serious mistakes. You did not demonstrate that you understand the content or purpose of the submission. |
| 0 | The work was not submitted, damaged, seriously incorrect, or unprofessional. The submission is rejected. |

# Updated Boilerplate

|  |  |  |
| --- | --- | --- |
|  | Express and IO sessions |  |
|  | Database |  |
|  | Standard Login, Formbar Login, Logout |  |
|  | Socket IO |  |
|  | Custom Modules |  |

# Job Openings

|  |  |  |
| --- | --- | --- |
|  | Resume |  |
|  | Email / Cover Letter |  |
|  | Portfolio / References |  |

# DocPac and Reflection *DocPac Submission Rules, DP09*

|  |  |  |
| --- | --- | --- |
| **1** | DocPac is turned in on time |  |
| **3** | a. DocPac is neatly folded |  |
| **3** | b. DocPac is not stained or damaged |  |
| **3** | c. No doodles, scribbles, or unnecessary writing |  |
| **4** | a. Answered each question in each prompt |  |
| **4** | b. Spelling and handwriting |  |
| **4** | c. No repeated answers from other DocPacs |  |
| **4** | d. Did not paraphrase assigned work |  |
| **6** | You are prepared to justify the use of any AI (you know what it does and why) |  |