**Route Specifications**

**1. Routes for users**

**1.A) route to create a new user:**

Endpoint: POST /users

Request body expected:

{

“firstName”: “some string”,

“lastName”: “some string”,

“jobPosition”: “some string”, //selected from radio buttons

“email”: “example@email.com”, //should be a valid email address

“username”: “some string”, //max 20 characters, cannot include spaces

“password”: “some string”

}

Successful response body:

{

userID: uuidv4(),

password: “hashed password”, //hashed using bcrypt

…req.body

}

**1.B) route to log in a registered user to create a JWT:**

Endpoint: POST /auth

Request body expected:

{

“email”: “example@email.com”,

“password”: “some string”

}

Successful response body: JWT

**1.C) route to get a listing of all users when a valid JWT is provided:**

Endpoint: GET /users

Request body expected: N/A

Successful response body: an array of all users

**1.D) route to get a specific user when given an ID alongside a valid JWT:**

Endpoint: GET /users/:id

Request body expected: N/A

Successful response body: user object

**1.E) route to delete a specific user when given an ID alongside a valid JWT:**

Endpoint: DELETE /users/:id

Request body expected: N/A

Successful response body: N/A

**1.F) route to update a specific user when given an ID alongside a valid JWT:**

Endpoint: PATCH /users/:id

Request body expected (same as for route 1.A):

{

“firstName”: “some string”,

“lastName”: “some string”,

“jobPosition”: “some string”, //selected from radio buttons

“email”: “example@email.com”, //should be a valid email address

“username”: “some string”, //max 20 characters, cannot include spaces

“password”: “some string”

}

Successful response body: updated user object

Notes:

-we could use route 1.D to populate a form with the user’s current values

**2. Routes for tickets**

**2.A) route to create a new ticket:**

Endpoint: POST /tickets/entries

Request body expected:

{

“userID”: “ID from currently logged in user”,

“content”: “ticket message goes here”

}

Successful response body:

{

ticketID: uuidv4(),

…req.body

}

**2.B) route to get a listing of all tickets:**

Endpoint: GET /tickets/entries

Request body expected: N/A

Successful response body: an array of all tickets

**2.C) route to get a specific ticket submission when given an ID alongside a valid JWT:**

Endpoint: GET /tickets/entries/:id

Request body expected: N/A

Successful response body: ticket object

**3. Routes for patients**

**3.A) route to create a new patient:**

Endpoint: POST /patients

Request body expected:

{

“firstName”: “some string”,

“lastName”: “some string”,

“address”: “some string”,

“phoneNumber”: “7091234567”, //must be a 10-digit number

“email”: “example@email.com”, //should be a valid email address

“DOB”: “yyyy/mm/dd”, //taken from date input field

“OHIP”: “some string” //validate for length??

}

Successful response body:

{

patientID: uuidv4(),

…req.body

}

**3.B) route to get a listing of all patients when a valid JWT is provided:**

Endpoint: GET /patients

Request body expected: N/A

Successful response body: an array of all patients

**3.C) route to get a specific patient when given an ID alongside a valid JWT:**

Endpoint: GET /patients/:id

Request body expected: N/A

Successful response body: patient object

**3.D) route to delete a specific patient when given an ID alongside a valid JWT:**

Endpoint: DELETE /patients/:id

Request body expected: N/A

Successful response body: N/A

**3.E) route to update a specific patient when given an ID alongside a valid JWT:**

Endpoint: PATCH /patients/:id

Request body expected:

{

“firstName”: “some string”,

“lastName”: “some string”,

“address”: “some string”,

“phoneNumber”: “7091234567”, //must be a 10-digit number

“email”: “example@email.com”, //should be a valid email address

}

Successful response body: updated patient object

Notes:

-assume DOB and OHIP don’t change

-we could use route 3.C to populate a form with the patient’s current values

4. TODO define routes for patient notes

5. TODO define routes for patient records

6. TODO define routes for patient revision history