Story 1: Create admin account.

As an admin, I want to create an account so I can get access to administrative tools on the website.

API
Method: Post
endpoint: /signup

Request body:
Admin object
{ “firstName”: John”, “lastName”: “Doe”, “email”: “admin@email.com”, “password”: “Password123!” }

Response:
Success:
Admin object
{ “firstName”: John”, “lastName”: “Doe”, “email”: “admin@email.com”, “password”: “Password123!” }

Error:
{ “error”: “First name is required” }
{ “error”: “Last name is required” }
{ “error”: “Password must contain at least one uppercase letter, one number, and one special character” }
{ “error”: “Email must be in a valid format” }

Acceptance Criteria:

1. Account creation:
   a. Administrators can create an account using first name, last name, email, password
   b. Password must meet certain security requirements:
      i. At least 8 characters
      ii. At least one uppercase letter
      iii. At least one number
      iv. At least one special character
   c. Email must be in a valid format (e.g. admin@email.com).

2. Error handling
   a. If the password or email does not meet requirements, appropriate errors are displayed

3. Email verification:
   a. Upon successful sign-up, a verification email will be sent to the administrator’s email.

Story 2: Update admin profile

As an admin, I want to be able to update my profile information, so I can keep my profile current in case something changes.

API
Method: Put
Endpoint 1: /admins/{admin_email}/password

Request body:

{ “current_password”: “Password123!”, “new_password”: “NewPassword123!” }

Response:
Success:
{ “message”: “Password successfully updated” }

Error:
{ “error”: “incorrect password” }
{ “error”: “password must contain at least one uppercase letter, one number, and one special character” }

{ “error”: “New password must be different from old password” }

Endpoint 2: /admins/{admin_email}/email

Request body:

{ “current_email”: “admin@email.com”, “new_email”: “john.doe@email.com” }

Response:
Success:
{ “message”: “Email successfully updated” }

Error:
{ “error”: “email must be in valid format” }
{ “error”: “new email and old email must be different.” }

Acceptance Criteria:

1. Password updated:
   a. Admins can update their password successfully.
   b. Password must still meet the security requirements
   c. Admins can update their email addresses
   d. Email must still be in a valid format

2. Error handling:

   a. If the new password or email does not meet requirements, is missing, or identical to the old one, an appropriate error message should be displayed.

3. Email verification:
   a. Upon successful email change, a verification email is sent to the new email address.

Story 3: Open a new medical case

As an admin, I want to be able to open a new medical case, so I can begin to keep track of it in a simple and standardized way.

API
Method: Post
Endpoint: /patients/medical-cases

Request body:
MedicalCase object

{ "case_id?": "", "date_opened?": "", "date_closed?": "", "patient": { "first_name": "Jane", "last_name?": "", "condition": "breast cancer", "gender": "F", "dob?": "" }, "cost_usd": "1000.00", "cost_bdt": "100000.00", "paid_usd": "0.00", "description": "Jane has been diagnosed with stage two breast cancer and requires a round of chemo therapy" }

Response:
Success:

{"message": "New case opened for ${patient.first_name}"}

MedicalCase object

{ "case_id?": "1", "date_opened?": "11/11/2024", "date_closed": "", "patient": { "first_name": "Jane", "last_name?": "N/A", "condition": "breast cancer", "gender": "F", "dob?": "N/A" }, "cost_usd": "1000.00", "cost_bdt": "100000.00", paid_usd": "0.00", "description": "Jane has been diagnosed with stage two breast cancer and requires a round of chemo therapy" }

Error:

{"error": {error}}

Acceptance Criteria:

1. Medical case created:
   a. Admins can post new medical cases to the database.
   b. The MedicalCase object requires a Patient object field.
   c. If age or last name of the Patient object aren't provided the field returns "N/A"
   d. Description should be limited to 120 characters
   e. The REST API should give each entry a distinct id, and fill in the date_opened field with the date at the moment of creation.

2. Error handling:
    a. If one of the required fields is missing or improperly formatted, the front end should return an appropriate warning to the admin before making a request to the REST API.
    b. The REST API should handle all validation as well as a failsafe measure for any errors that make it through the front end. Validation should return an appropriate error message.
    c. The request should never include a value in the date_closed field.

Story 4: Get all medical cases

As an admin, I want to be able to return all medical records we have on file, so I can keep track of their statuses and payment details.

API
Method: Get
Endpoint: /patients/medical-cases

Request body:
NONE

Response:
Success:
All MedicalCase objects
OR
{ “message”: “No medical cases in the database” }

Error:
{ “error:” “{error}” }

Acceptance Criteria:

1. Return all medical cases:
    a. When an admin goes to the medical cases page, a list of all medical cases is provided, sorted on date_opened by newest to oldest, then by newest to oldest on date_closed.
    b. If the database returns an empty list a message that “no medical cases in the database” is returned.
2. Error handling:
    a. If there is an error connecting to the database or another error, the appropriate error message should return.

Story 5: Update individual medical cases by case_id

As an admin, I want to be able to update a medical case, so I can change the cost, description, or close the case.

API
Method: Post
Endpoint: /patients/medical-cases/{case_id}

Request body:
MedicalCase object

{ “case_id?”: “1”, “date_opened?”: “11/11/2024”, “date_closed?”: “”, “patient”: { “first_name”: “Jane”, “last_name?”: “”, “condition”: “breast cancer”, “gender”: “F”, “dob?”: “” }, “cost_usd”: “1250.00”, “cost_bdt”: “125000.00”, “paid_usd”: “1000.00”, “description”: “Jane’s stage two breast cancer is going into remission. New series of tests ordered.” }

Response:
Success:

{“message:” “Case updated”}

MedicalCase object

{ “case_id?”: “1”, “date_opened?”: “11/11/2024”, “date_closed?”: “”, “patient”: { “first_name”: “Jane”, “last_name?”: “”, “condition”: “breast cancer”, “gender”: “F”, “dob?”: “” }, “cost_usd”: “1250.00”, “cost_bdt”: “125000.00”, “description”: “Jane’s stage two breast cancer is going into remission. New series of tests ordered.” }

Error:

{“message:” “Case failed to update”}
{“error”: {error}}

Acceptance Criteria:

1. Updating medical case:
   a. Admins can only update the date_closed, costs, and description fields.
   b. Inputting a date for date_closed causes the case to be sorted beneath cases with no date_closed input when getting all medical cases.
   c. Date_closed cannot be changed if cost_usd !== paid_usd

2. Error handling:
   a. If date_closed is before date opened return an error
   b. If date_closed has a value and cost_usd != paid_usd, return an error
   c. Case_id, date_opened, or patient object is modified, return an error

~~**Story 6:** Donate~~

~~As a visitor, I want to donate to the organization, so I can support the causes that matter to me most.~~