CREATE
OPPORTUNITY

Final Health Project

Project Background:

Super Health Inc. is a small regional healthcare company that operates a series of clinics. The company has an existing application for tracking patient encounter data that has been in service for a number of years and needs a rewrite. Super Health has hired you to rewrite the application in a modern way using the technologies that you have trained in. At this point, Super Health Inc. is looking for a proof of concept and will not require any authentication or authorization.

The database stores patient and encounter data, and its design is up to you. Any user can review, create, and update patient information and encounters. The client has expressed that they would like the project to be documented and easily maintainable.

Functional Requirements (70%)

Patients UI Features

Styling Requirements

We have no styling or layout requirements other than those described in the feature requirements. Styling decisions and how this project looks are largely up to your decision, as long as your decisions don't break other functional requirements.

Patient List View

1. Given a user has navigated to the patient's page, all patients are displayed.
2. Given a user is viewing a patient, the patient's name, age, and gender are displayed.

Create Patient

3. Given a user is on the Patient List View, there is a way to access the Create Patient form inputs.
4. Given a user is viewing the Create Patient form and has filled the form with valid information, when they hit submit, then a new patient record is successfully added to the system.
5. Given a user has just successfully added a patient to the system, they are redirected to the patient list view.
6. Given a user has just successfully added a patient to the system and been redirected to the patient list view, the newly added patient is visible without the page needing to be refreshed by the user.

Delete Patient

7. Given a user has navigated to the Patient List View, the user can delete a patient without any associated encounters
8. Given a user has been successfully deleted from the system, the Patient List View is updated without the page needing to be refreshed by the user.
9. Given a user tries to delete a patient who has associated encounters, an error message appears to inform the user of the problem.

Patient Detail View

10. Given a user is viewing the Patient List View and clicks the "View Details" button, the user is navigated to the Patient's Detail View.

11. Given a user has navigated to the Patient Detail View, the patient's id, first name, last name, ssn, email, street, city, state, zip code, age, height, weight, insurance, and gender information are displayed.
12. Given a user has navigated to the Patient Detail View, there is a way to navigate back to the Patient List View.

Edit Patient

13. Given a user is viewing the Patient Detail View and selects the "Edit Patient" button, the user is navigated to the Edit Patient feature, which displays all patient fields.
14. Given a user has been navigated to the Edit Patient feature, the form has been prepopulated with the existing patient's information.
15. Given a user makes valid changes to any or all editable fields and submits the form, those changes are persisted in the database.
16. Given a user has successfully edited a patient, they are redirected back to the Patient Detail View for that patient.

View Encounters

17. Given a user has navigated to the Patient Detail View, then all the patient's encounters are displayed. The encounter id, visit code, provider, and date should be visible for each encounter.

View Encounter Details

18. Given a user is viewing the list of encounters, each encounter should have a way to view details for that encounter. When viewing the details, that encounter's id, notes, visit code, provider, billing code, International Classification of Diseases - 10th Ed. code (icd10), total cost, copay, chief complaint, pulse, systolic pressure, diastolic pressure, and date information are displayed.

Create Encounter

19. Given a user is viewing the Patient Detail View and selects the "Create Encounter" button, the user is navigated to the Create Encounter feature, the user can edit the encounter's notes, visit code, provider, billing code, icd10 code, total cost, copay, chief complaint, pulse, systolic pressure, diastolic pressure, and date.
20. Given a user makes valid changes to any or all editable fields and submits the form, those changes are persisted in the database.
21. Given a user has successfully created an encounter, they are redirected back to the Patient Detail View for that patient.
22. Given a user has just successfully added an encounter to the system and been redirected to the Patient Detail View, the newly added encounter is visible without the page needing to be refreshed by the user.

Edit Encounter

23. Given a user is viewing Encounter Details and selects the "Edit Encounter" button, the user is navigated to the Edit Encounter feature, which displays all encounter fields.
24. Given a user has been navigated to the Edit Encounter feature, the form has been prepopulated with the existing encounter information.
25. Given a user makes valid changes to any or all editable fields and submits the form, those changes are persisted in the database.

26. Given a user has successfully edited an encounter, they are redirected back to the Patient Detail View for that patient.

Patient Form Validation

27. Given a user is creating or editing a patient, the user can successfully enter / edit the patient's first name, last name, social security number (ssn), email, street, city, state, zip code, age, height, weight, insurance, and gender.
28. Given a user is creating or editing a patient and clicks the submit button and any inputs are empty, or only white space, an error message appears near the empty input and any values entered in an input are preserved.
29. Given a user is creating or editing a patient and clicks the submit button and the age is not a number, an error message appears near the age input and any values entered in an input are preserved.
30. Given a user is creating or editing a patient and clicks the submit button and the height is not a number, an error message appears near the height input and any values entered in an input are preserved.
31. Given a user is creating or editing a patient and clicks the submit button and the weight is not a number, an error message appears near the weight input and any values entered in an input are preserved.
32. Given a user is creating or editing a patient and clicks the submit button and the address does not have a valid two letter state code, an error message appears near the state input and any values entered in an input are preserved.
33. Given a user is creating or editing a patient and clicks the submit button and the ssn is not a valid ssn format of *xxx-xx-xxxx (ex. 123-45-6789)*, an error message appears near the ssn input and any values entered in an input are preserved.
34. Given a user is creating or editing a patient and clicks the submit button and the email is not a valid email format of *x@x.x (ex. test@test.com)*, an error message appears near the email input and any values entered in an input are preserved.
35. Given a user is creating or editing a patient and clicks the submit button and the zip code is not a valid zip code format of *xxxxx* or *xxxxx-xxxx* (ex. 12345 or 12345-6789), an error message appears near the zip code input and any values entered in an input are preserved.
36. Given a user is creating or editing a patient and clicks the submit button and the gender is not "Male", "Female", or "Other", an error message appears near the gender input and any values entered in an input are preserved.

Encounter Form Validation

37. Given a user is editing or creating an encounter and clicks the submit button and the visit code is empty or not a valid visit code format of *LDL DLD (ex. A1S 2D3)*, an error message appears near the visit code input and any values entered in an input are preserved.
38. Given a user is editing or creating an encounter and clicks the submit button and the provider is empty, or only white space, an error message appears near the provider input and any values entered in an input are preserved.
39. Given a user is editing or creating an encounter and clicks the submit button and the billing code is empty or not a valid billing code format of *xxx.xxx.xxx-xx* (ex. 123.456.789-12), an error message appears near the billing code input and any values entered in an input are preserved.

40. Given a user is editing or creating an encounter and clicks the submit button and the icd10 code is empty or not a valid icd10 code format of *LDD (ex. A12)*, an error message appears near the icd10 code input and any values entered in an input are preserved.
41. Given a user is editing or creating an encounter and clicks the submit button and the total cost is empty or not a number, an error message appears near the total cost input and any values entered in an input are preserved.
42. Given a user is editing or creating an encounter and clicks the submit button and the copay is empty or not a number, an error message appears near the copay input and any values entered in an input are preserved.
43. Given a user is editing or creating an encounter and clicks the submit button and the chief complaint is empty, or only white space, an error message appears near the chief complaint input and any values entered in an input are preserved.
44. Given a user is editing or creating an encounter and clicks the submit button and the pulse is not a number, an error message appears near the pulse input and any values entered in an input are preserved.
45. Given a user is editing or creating an encounter and clicks the submit button and the systolic pressure is not a number, an error message appears near the systolic pressure input and any values entered in an input are preserved.
46. Given a user is editing or creating an encounter and clicks the submit button and the diastolic pressure is not a number, an error message appears near the diastolic pressure input and any values entered in an input are preserved.
47. Given a user is editing or creating an encounter and clicks the submit button and the date is empty or not a valid date format of YYYY-MM-DD (ex. 2020-01-01), an error message appears near the date input and any values entered in an input are preserved

API Requirements

48. Given a GET request is made to "/patients", then all patients are returned in a JSON array, and the status code is 200.
49. Given a patient exists, when a GET request is made to "/patients/<that-patient-id>", then that patient is returned, with a status code of 200.
50. Given a GET request is made to "/patients/<non-existent-id>", then an error response is returned with a status code of 404.
51. Given a valid patient has been provided in a JSON request body, when a POST request is made to "/patients", then a patient entity is stored in a database, and a JSON representation of the newly created patient is returned, with a status code of 201.
52. Given a valid patient has been provided in a JSON request body, and given that patient's email has been used already, when a POST request is made to "/patients", then an error response is returned with a status code of 409.
53. Given an invalid patient has been provided in a JSON request body, when a POST request is made to "/patients", then an error response is returned with a status code of 400.
54. Given a valid encounter for a patient has been provided in a JSON request body, when a POST request is made to "/patients/<that-patient-id>/encounters", then a new encounter for that patient is stored in a database, and a JSON representation of the newly created encounter is returned, with a status code of 201.

55. Given an invalid encounter for a patient has been provided in a JSON request body, when a POST request is made to "/patients/<that-patient-id>/encounters", then an error response is returned with a status code of 400.
56. Given a patient exists, and given that patient's JSON representation has been altered and provided in a request body, when a PUT request is made to "/patients/<that-patient-id>", then the patient is updated in the database, and a JSON representation of the newly updated patient is returned, with a status code of 200.
57. Given a PUT request is made to "/patients/<non-existent-id>", then an error response is returned with a status code of 404.
58. Given JSON of an existing patient, and given the email field has been updated to a different patient's email, when a PUT request is made to "/patients/<that-patient-id>", then an error response is returned with a status code of 409.
59. Given JSON of an existing patient, and given updates have caused any field to become invalid, when a PUT request is made to "/patients/<that-patient-id>", an error response is returned with a status code of 400.
60. Given a patient with encounters exists, and given an encounter's JSON representation has been altered and provided in a request body, when a PUT request is made to "/patients/<that-patient-id>/encounters/<that-encounter-id>", then the encounter is updated in the database, and a JSON representation of the newly updated encounter is returned, with a status code of 200.
61. Given a PUT request is made to "/patients/<some-patient-id>/encounters/<non-existent-id>", then an error response is returned with a status code of 404.
62. Given JSON of an existing encounter for a patient, and given updates have caused any field to become invalid, when a PUT request is made to "/patients/<that-patient-id>/encounters/<that-encounter-id>", an error response is returned with a status code of 400.
63. Given a patient without encounters exists, when a DELETE request is made to "/patients/<that-patient-id>", then that patient is deleted, and a status code of 204 is returned.
64. Given a patient with encounters, when a DELETE request is made to "/patients/<that-patient-id>", then that patient is not deleted, and a status code of 409 is returned.
65. Given a DELETE request is made to "/patients/<non-existent-id>", then a status code of 404 is returned.

All Domains

66. Given the database is not running, when a request is made to a registered endpoint, then an error response is returned with a status code of 503, and the frontend displays an appropriate error message.

Non-Functional Requirements (30%)

67. Frontend README includes a minimum of description, pre-requisites, usage, and testing sections.
68. Frontend README specifies any external dependencies to the project, including links if appropriate.
69. Frontend encounter form validation is arranged into one function
70. Frontend encounter validation method is unit tested >= 80% line coverage
71. Frontend README specifies how to run ESLint via an npm script using Airbnb's configuration.

72. There are no linting errors or warnings in the frontend code.
73. Frontend includes proper function level and inline documentation using JSDoc style.
74. Validation that does not require checking values in the database should be done on the frontend, i.e. does not make a call to the api to validate.
75. Frontend includes a valid .gitignore file, and ignored files do not exist in the remote repository.
76. Backend README includes a minimum of description, pre-requisites, usage, and testing sections.
77. Postman collection is exported to a JSON file and present in backend GitLab repository. The postman collection demonstrates all of the 2XX and 4XX functional requirements.
78. Backend adheres to appropriate coding standards, instructions on how to lint the project are included in the README, and all files have been linted.
79. Backend includes appropriate code comments on all logic, service, and controller files (e.g. Javadocs).
80. Backend unit tests cover at least 80% of any services/logic (e.g. validation). Instructions for how to run the unit tests with code coverage are included in the README.
81. Backend Integration Testing covers payload, content type, and status code tests of all 2XX scenarios.
82. Backend Integration Testing covers payload, content type, and status code tests of all 4XX scenarios.

Validation Requirements

The following charts list the data model requirements for the client's data. Validation information is supplied in the description of some fields. Data should be validated against these patterns.

Validation patterns for data:
D = Numeric digit
L = Alphabetic character
N = Alphabetic Names character (includes hyphens and apostrophe)
A = Alphanumeric character
* = Previous character can repeat

Patients

Field	Type	Description
id	**Required**	Database id
firstName	String, **Required**	Patient first name Validation (N*)
lastName	String, **Required**	Patient last name Validation (N*)
ssn	String, **Required**	Social Security Number Validation(DDD-DD-DDDD) Ex. 123-45-6789
email	String, **Required, Unique**	Email format (A*@L*.L*)
street	String, **Required**	Street address
city	String, **Required**	City
state	String, **Required**	Two character state code Validation (LL) Ex. MD
postal	String, **Required**	Postal code Validation (DDDDD or DDDDD-DDDD) Ex. 12345 or 12345-1234
age	Number, **Required**	Patient age Validation (D*) Ex. 25
height	Number, **Required**	Patient height in inches Validation (D*) Ex. 72
weight	Number, **Required**	Patient weight in pounds Validation(D*) Ex. 170
insurance	String, **Required**	Patient insurance provider
gender	String, **Required**	Gender of patient. Valid inputs (“Male”, “Female”, "Other")

Encounters

Field	Type	Description
id	**Required**	Database id
patientId	**Required**	Patient database id
notes	String, **Optional**	Notes about the encounter
visitCode	String, **Required**	Office visit code Validation(LDL DLD) Ex. H7J 8W2
provider	String, **Required**	Name of provider
billingCode	String, **Required**	Encounter billing code Validation(DDD.DDD.DDD-DD) Ex. 123.456.789-12
icd10	String, **Required**	ICD10 code for encounter Validation(LDD) A22
totalCost	Number, **Required**	Total cost of encounter including copay in US dollars
copay	Number, **Required**	Patient's copay for encounter in US dollars
chiefComplaint	String, **Required**	Initial complaint of the patient at the start of the encounter
pulse	Number, **Optional**	Patient's pulse in beats per minute.
systolic	Number, **Optional**	Systolic portion of blood pressure Validation(D*) Ex. 100
diastolic	Number, **Optional**	Diastolic portion of blood pressure Validation(D*) Ex. 80
date	Date, **Required**	Date of encounter in ISO 8601 Date Format: YYYY-MM-DD Ex. 2019-12-31

Request Body Requirements

We require that you follow the following formats for the request bodies:

Patient Object

```
{
"id": 1,
"firstName": "Hulk",
"lastName": "Hogan",
"ssn": "123-45-6789",
"email": "hulksnewemailaddress@wwf.com",
"age": 66,
"height": 79,
"weight": 299,
"insurance": "Self-Insured",
"gender": "Male",
"street": "8430 W Sunset Blvd",
"city": "Los Angeles",
"state": "CA",
"postal": "90049"
}
```

Encounters Object

```
{
"id": 1,
"patientId": 1,
"notes": "new encounter",
"visitCode": "N3W 3C3",
"provider": "New Hospital",
"billingCode": "123.456.789-00",
"icd10": "Z99",
"totalCost": 0.11,
"copay": 0,
"chiefComplaint": "new complaint",
"pulse": "",
"systolic": "",
"diastolic": "",
"date": "2020-08-04"
}
```