DataSource: We could use another datasource like database or
cloud platform, so DataSource is generic for this purpose
In this case DataSource will get data from Monash FHIR API

Identifier

system: string
value: string

+ Identifier(system: string, value: string)
+ toString(): string

<<interface>>
DataSource

+ getPractitioner(practitionerIdentifier: Identifier):
Promise<MaybePractitioner>
+ getPatientList(practitionerIDs: string[], progressCallback: (data: Patient[]
progress: Progress) => void): Promise<Patient[]>
+ getCholesterol(patientID: string): Promise<Observation | null>;

<<provider>>
store

+ patients: PatientsState
+ practitioner: PractitionerState
+ system: SystemState

components

<<views>>
MainContent

<<views>>
PatientsMonitor

- patients: Patient[]

<<views>>
PatientsSelect

- patients: Patient[]
- loading: boolean

Sidebar

PatientInfoDialog

PatientsTable

FHIRServer

+ rootUrl: string;

- extractNextUrl(data: {link:[]}): string
- decodePatients(data: any[]): Patient[]

Practitioner

+ identifier: Identifier
+ ids: string[]
+ name: Name[]
+ address: Address[]

+ Practitioner(identifier: Identifier, ids:
string[], name: Name[], address: Address[])

Patient

+ id: string
+ name: Name[]
+ gender: string
+ birthDate: string
+ address: Address[]
+ isMonitored: boolean
+ totalCholesterol: Observation | undefined |
null
+ cholesterolLoading: boolean

+ Patient(newId: string, newName: Name[],
newGender: string, newBirthDate: string,
newAddress:Address[])

Name

+ family: string
+ given: string[]
+ prefix: string[]
+ use: string

+ Name(newFamily: string, newGiven: string[],
newPrefix: string[], newUse: string)
+ toString(): string

Address

+ line: string[]
+ city: string
+ state: string
+ postalCode: string
+ country: string

+ Address(newLine: string[], newCity: string,
newState: string, newPostalCode: string,
newCountry: string)

Observation

+ effectiveDateTime: string
+ value: Measurement

+ Observation(value: Measurement,
effectiveDateTime: string)

Measurement

+ unit: string
+ value: number

+ Measurement(newUnit: string, newValue:
number)
+ toString(): string

1..1
1..1
1..1
1..1
1..1
1..1
1..1
1..1
1..1
1..1
1..1
0..1
1..1
1..1
1..1