



# 21/Q4 Mindvitals Technical Specification

Written by Poitier Stringer, Last updated on @November 18, 2021

## Problem & Goals

### Need Specification

This document will provide clear descriptions of the proposed application's functionality. Based on those descriptions, it will describe the APIs and frontend components that need to be implemented as well as the tasks associated with the development process. At the end it will propose actionable steps and describe open questions that need to be addressed to move forward.

## User Stories

### Main Patient Workflow

MindVitals Lite (MindVitals Implemented w/o any backend codes)

#### Video Walkthrough (Patient Side)

#### Interactive Mockup (Patient Side, Desktop)

1. User initiates the assessment there are 2 possible routes for this to happen:
  - a. user receives a link in their email which they follow to the assessment page.
  - b. user scans a QR code with their mobile device or provided mobile device. The device then navigates to an assessment page.
2. Upon navigating to the link, the user is presented with a series of questions, shown one at a time with a back button at the bottom.
  - a. When an answer is selected, the current assessment is updated using selected answers and the question index is incremented
  - b. When the back button is clicked, the question index is decremented
3. Upon completing the assessment, the user is shown the result screen with the following features:
  - a. results summary
  - b. option to be sent a copy of results
  - c. option to be contacted by a care navigator

### Main Provider Workflow

#### Video Walkthrough (Provider Side)

#### Interactive Mockup (Provider Side)

1. Provider authenticates with their clinic's version of mindvitals.
2. Upon authentication, the provider is directed to the patient dashboard where they see a list of the patients in their care.
3. Clicking on the patient's row will direct the user to a detail page for that patient
4. Each patient has an action button associated with them. Clicking a patient's action button shows 3 options to initiate a survey.
  - a. Text
  - b. Email

- c. QR code
- 5. In each case we are using the same generated link as the source
- 6. In the case of text and email, a snack-bar will be enqueued with the delivery notification
- 7. In the case of the QR code, it will display on the provider's screen for the patient to capture on either their mobile device or one provided by the provider.

## Patient detail page workflow

### Video Walkthrough (Provider Side)

### Interactive Mockup (Provider Side)

1. Upon navigating to the Patient Detail Page, the user sees 3 main cards:
  - a. Profile Card
  - b. GAD-7 and PHQ-9 charts
  - c. A **Data Grid** containing information on previously taken assessments including
    - i. timestamp
    - ii. mild to severe classification of PHQ-9 score
    - iii. mild to severe classification of GAD-7 score
2. Above profile information, there are two buttons
  - a. The **Send Request** button serves the same purpose as the action button described in the main provider workflow.
  - b. The **Refer Patient** button opens a dialog with a form that allows the provider to select the types of care required by the patient, the patient's insurance, and any relevant notes
    - i. The dialog's **Submit** button will notify a referral coordinator with Prairie Health for follow-up.
3. After a referral request is submitted, the provider is shown a confirmation dialog with a close button

## Data Model

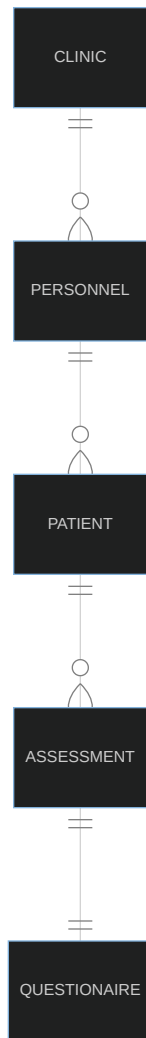
**CLINIC:** The organization with which personnel are associated

```
CLINIC {
  pk: clinicId
  fk: personnelId
  string: name
}
```

**PERSONNEL:** Personnel overseeing a part of the patient population on behalf of the clinic. These are the authenticated users.

```
PERSONNEL {
  pk: personnelId
  fk: clinicId
  string: type // 'nurse' | 'pcp' | 'staff'
  string: privilege // 'admin' | 'standard'
}
```

**PATIENT:** The PATIENT entity represents the person receiving care. This is not an authenticated user. Patients are not required to authenticate to take an assessment. Assessment results are associated with the PATIENT entity.



```

PATIENT {
  pk: patientId
  fk: providerId //references personnel
  string: patientIdentifier // UUID used for link generation
  string: //
}

```

**ASSESSMENT:** The ASSESSMENT entity represents assessments completed by the patient. Answers will be stored on this entity as an array of integers, each in the range of [0, 3] corresponding to the strings 'Not at all', 'Several days', 'More than half the days', 'Nearly every day'. Each assessment is associated with exactly one questionnaire which contains the question data that will be rendered on the frontend.

```

ASSESSMENT {
  pk: assessmentId
  fk: patientId
  fk: questionnaireId // references one of the assessments
  string: type // type of assesment taken, phq9 or gad7
  int[9]: answers // each element defaults to -1 for unanswered, otherwise, a value in [0, 3]
  int: score // the total assesment score
}

```

**QUESTIONNAIRE:** The QUESTIONNAIRE entity represents the questionnaire to be associated with completed assessments. It contains the actual questions to be rendered during the assessments as an array of strings. As well as the type of assessment (PHQ-9 or GAD-7).

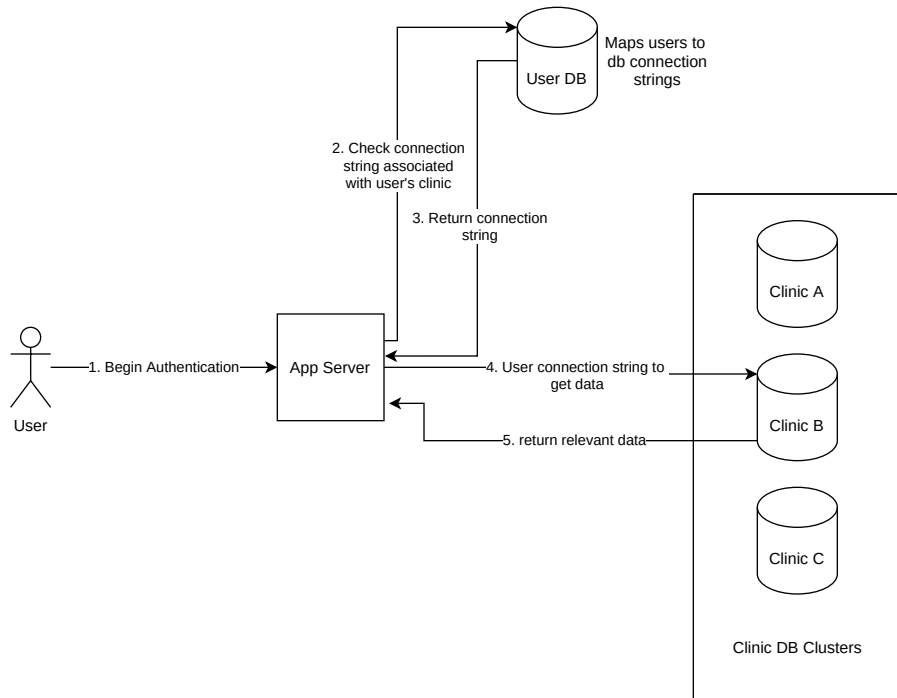
```

QUESTIONNAIRE {
  pk: questionnaireId
  string[ ]: questions // quesitons to be rendered in the frontend
  string[4]: answers //possible answers for each question
}

```

## Architecture

In the interest of keeping data separate for each clinic there needs to be a separate cluster for each clinic. This can be achieved by maintaining a separate database matching users to connection strings.



Upon successful user authentication, the db connection string associated with the user's clinic will be returned. Using this method, we can work with a single instance of the app at one domain, and accommodate as many clinics as we can store in a db.

## Proposed Stack

Requirement	Proposed Solution	Potential Cons to Proposed Solution	Alternative
App Server Deployment	<b>GCP:</b> I don't currently see a reason we can't have a similar deployment setup as our current applications.		AWS
DB Management	<b>MongoDB Atlas:</b> Supports sharding if the size of the dataset for each clinic grows in the future, automatic management of replica sets. Multiple methods to automate deployment of new replica sets for onboarding clinics (CLI, terraform).	about 2x more expensive than manual set-up on a general cloud provider.	Deploy replica sets on GCP or AWS. It is possible to implement all of the features mentioned on GCP or AWS. The monetary cost would be a lower, with more complexity to manage. I don't think it would be a good idea for MVP unless one of our engineers is already really skilled in this area, or we have a dedicated dev-ops person.

Requirement	Proposed Solution	Potential Cons to Proposed Solution	Alternative
Backend framework for app server	<b>Nest.js:</b> Built on top of Express.js, which the team is already familiar with. Enforces modular design conventions, includes CLI to decrease the amount of boiler-plate code to write. Jest integrated by default, automatically scaffolds tests for new modules. Supports typescript by default but not required to use aside from decorators. Includes extensive documentation for a significant number of use cases. Supports injection of asynchronous providers which could help provide a clean way to manage connections to multiple DBs.	These cons are mainly based around developer preference. The dependency injection system in nest may be annoying to use for some developers. Nest is fairly opinionated about how projects should be scaffolded. While the use of most Typescript features is not required, it wouldn't really make sense to use nest without decorators. And while data-transfer objects are not necessary, their use would make for cleaner, more maintainable code.	I've yet to come across any backend javascript frameworks comparable to nest in terms of feature set. The alternative is to stick with a lighter framework such as Express or Fastify and establish conventions as a team to ensure the codebase's maintainability in the future.
frontend tooling	<b>Create-React-App:</b> Everybody is familiar with it and it gets the job done. We should probably adopt some conventions such as in bulletproof react <a href="https://github.com/gsoft-inc/craco">https://github.com/gsoft-inc/craco</a>	As the app grows, default configurations may become outdated. At such a time we can eject the app or use something like: <a href="https://www.npmjs.com/package/react-app-rewired">https://www.npmjs.com/package/react-app-rewired</a>	Next.js is a popular framework for building production react apps, but most of the features it offers don't seem very relevant to this use case.
testing	<b>Jest, React Testing Library, Cypress for e2e.</b>		
Monitoring	<b>Sentry</b>		

## Technical Requirements

### Required API Routes

Verb	Route	Role	Description	Workflow	Relevant Stories	Related to Required Frontend Routes (Relevant APIs)
GET	/assessments/:patient_id	Patient	Retrieve all assessments given a patient id	Main Patient	<u>Either there is an un-finalized assessment saved from a previous session. In which case the user will be prompted to continue the previous or start a new assessment. If the user chooses to start a new assessment, the un-finalized one will be discarded.</u>	/start-assessment/:uuid
POST	/assessments/:patient_id	Patient	Create a new assessment under the specified patient	Main Patient	<u>In the case that there is no un-finalized assessment, a new one will be created.</u>	/start-assessment/:uuid

≡ Verb	≡ Route	≡ Role	≡ Description	≡ Workflow	Aa Relevant Stories	↗ Related to Required Frontend Routes (Relevant APIs)
PUT	/assessments/:assessment_id	Patient	Update an assesment's data given its id	Main Patient	<u>Upon clicking next, the current assessment is updated using selected answers.</u>	/assesment/assesment_id
GET	/patients	Provider	Retrieve patients in the care of the authenticated user	Main Provider	<u>the provider is directed to the patient dashboard where they see a list of the patients in their care.</u>	/patient
POST	/actions/notify/:patient_id/:method	Provider	Send a notification via the specified method (email, sms) to take the assessment	Main Provider	<u>Each patient has an action button associated with them. Clicking a patient's action button shows 3 options to initiate a survey.</u>	/patient
GET	/patients/:patient_id	Provider	All of the information required for display on the patient detail page will be returned by this api.	Patient Detail Page	<u>Upon navigating to the Patient Detail Page, the user sees 3 main cards:</u>	
POST	/patients/refer/:patient_id	Provider	Submit referral information and notify relevant parties on the backend	Patient Detail Page	<u>The Refer Patient button opens a dialog with a form that allows the provider to select the types of care required by the patient, the patient's insurance, and any relevant notes</u>	

≡ Verb	≡ Route	≡ Role	≡ Description	≡ Workflow	Aa Relevant Stories	↗ Related to Required Frontend Routes (Relevant APIs)
POST	/login	Provider	Login	Authentication Main Provider	<u>Provider authenticates with their clinic's version of Mindvitals.</u>	
POST	/register	Provider	Register a new user, it still needs to be determined how to have a user join their organization when signing up.	Authentication Onboarding	<u>Provider authenticates with their clinic's version of Mindvitals.</u>	
POST	/clinic	Provider	Create a clinic using the form info and assign the authenticated user as its admin	Onboarding	<u>Provider submits form and Prairie is notified.</u>	
GET	/personnel	Admin Owner	retrieve a list of personnel with relevant information to display in the datagrid	Personnel Managment	<u>Upon navigating to the personnel management page, the user is presented with a data grid showing current personnel and their roles.</u>	
PUT	/personnel/:personnel_id	Admin Owner	Update a personnel document	Personnel Managment	<u>Upon clicking Update, the user is asked to confirm their modifications before submission. After confirming the dialog is closed. If the user rejects the confirmation, they are returned to the form</u>	

≡ Verb	≡ Route	≡ Role	≡ Description	≡ Workflow	Aa Relevant Stories	↗ Related to Required Frontend Routes (Relevant APIs)
DELETE	/personnel/:personnel_id	Admin Owner	delete a personnel document	Personnel Managment	Each row contains an edit button. When the button is clicked a dialog form appears containing the modify-able properties of the org-member. And a button that will allow the user to delete this personnel document upon confirmation.	
					Untitled	

#### Required Frontend Routes

Aa Route	≡ Description	≡ Role	≡ Workflow	≡ Relevant Story	↗ Relevant APIs
/start-assessment/:uuid	This is the page the patient is directed to when an assessment is requested. A UUID associated with the patient is used as a route param to make for a unique link to generate the QR code. It is also used to make the request for assessments	Patient	Main Patient	Either there is an un-finalized assessment saved from a previous session. In which case the user will be prompted to continue the previous or start a new assessment. If the user chooses to start a new assessment, the un-finalized one will be discarded.	<u>Either there is an un-finalized assessment saved from a previous session. In which case the user will be prompted to continue the previous or start a new assessment. If the user chooses to start a new assessment, the un-finalized one will be discarded. In the case that there is no un-finalized assessment, a new one will be created.</u>
/assesment/:assessment_id	This is the page where the user will take their assessment be it newly created, or a previously un-finalized one.	Patient	Main Patient	The user is presented with a series of questions, shown one at a time with two prominent navigation buttons.	<u>Upon clicking next, the current assessment is updated using selected answers.</u>









Aa Route	Description	Role	Workflow	Relevant Story	Relevant APIs
<a href="#">/patient</a>	This page displays information on all of the providers patients and allows the provider to request assessments from patients	Provider	Main Provider		<u>Each patient has an action button associated with them. Clicking a patient's action button shows 3 options to initiate a survey., the provider is directed to the patient dashboard where they see a list of the patients in their care.</u>
<a href="#">/patient/patient_id</a>	Displays patient information, summary of charts and assesments	Provider	Patient Detail		
<a href="#">/admin</a>	Allows privileged users to make changes to personnel and organization level info.	Admin Owner	Personnel Managment		

## Relevant Tasks

- ☐ Preparation
  - ☐ Decide on onboarding flow and produce mockups (product)
  - ☐ Decide mockups for personnel management (product)
  - ☐ Add specificity to data model specification (eng)
  - ☐ Decide on deployment strategy (Requires deliberation and research) (eng)
  - ☐ Decide on standards conventions/frameworks (eng)
  - ☐ Set-up app server (eng)
  - ☐ Set-up auth db to map users to clinics to connection strings (eng)
  - ☐ Set-up test database clusters (eng)
  - ☐ Set-up frontend project (eng)
- ☐ Development
  - ☐ Implement Authentication Flow
  - ☐ Implement db provider to select appropriate cluster based on user's org
  - ☐ Main Provider Workflow
    - ☐ Implement notification system
      - ☐ patient assessment requests
      - ☐ patient referrals
    - ☐ Implement apis associated with main provider workflow
    - ☐ Implement frontend for main provider workflow
      - ☐ Sign in
      - ☐ Registration
      - ☐ Sidebar and navigation
      - ☐ Provider Dashboard
      - ☐ Patient Detail Page

- ☐ Patient Detail Dashboard
  - ☐ Implement apis related to patient details
  - ☐ Implement frontend for patient details
- ☐ Main Patient Workflow
  - ☐ Implement apis associated with the patient detail page
  - ☐ Implement frontend for patient detail page
  - ☐ Implement apis associated with main patient workflow
  - ☐ Implement frontend for main patient workflow
    - ☐ Implement QR code generation and display
- ☐ Testing
  - ☐ user testing
  - ☐ set-up unit testing
  - ☐ set-up CI-CD

#### Project Timeline

 Name	 Assignee	 Date	 Description	 Due Date	 Section/C
<a href="#">Set-up App Project</a>		@November 23, 2021 → November 24, 2021	<a href="https://www.notion.so/prairiehealth/Set-up-App-Project-4cb298a883d445e2bb7055622c7a006f">https://www.notion.so/prairiehealth/Set-up-App-Project-4cb298a883d445e2bb7055622c7a006f</a>	11/24/2021	Not start
<a href="#">Set-up Auth Database</a>		@November 25, 2021 → November 26, 2021	<a href="https://www.notion.so/prairiehealth/Set-up-Auth-Database-8e65cb0022fa46a78221383cd45cdc7e">https://www.notion.so/prairiehealth/Set-up-Auth-Database-8e65cb0022fa46a78221383cd45cdc7e</a>	11/26/2021	Not start
<a href="#">Set-up clinical databases for testing</a>		@November 27, 2021	<a href="https://www.notion.so/prairiehealth/Set-up-clinical-databases-for-testing-2bd28552800e4cb78163d2a11d4adc23">https://www.notion.so/prairiehealth/Set-up-clinical-databases-for-testing-2bd28552800e4cb78163d2a11d4adc23</a>	11/27/2021	Not start
<a href="#">Implement Authentication Flow</a>		@November 29, 2021 → November 30, 2021	<a href="https://www.notion.so/prairiehealth/Implement-Authentication-Flow-b525534f25634367ade6df653dcc5548">https://www.notion.so/prairiehealth/Implement-Authentication-Flow-b525534f25634367ade6df653dcc5548</a>	11/30/2021	Not start
<a href="#">Implement APIs for Main Provider Workflows</a>		@December 1, 2021 → December 3, 2021	<a href="https://www.notion.so/prairiehealth/Implement-APIs-for-Main-Provider-Workflows-612316138c7d4855bef22cc13df0cf65">https://www.notion.so/prairiehealth/Implement-APIs-for-Main-Provider-Workflows-612316138c7d4855bef22cc13df0cf65</a>	12/3/2021	Not start
<a href="#">Implement APIs for Patient Detail Page</a>		@December 6, 2021	<a href="https://www.notion.so/prairiehealth/Implement-APIs-for-Patient-Detail-Page-54bb76f0b32e472cb55ad91cd0e6eda2">https://www.notion.so/prairiehealth/Implement-APIs-for-Patient-Detail-Page-54bb76f0b32e472cb55ad91cd0e6eda2</a>	12/6/2021	Not start
<a href="#">Implement Notification System</a>		@December 7, 2021 → December 8, 2021	<a href="https://www.notion.so/prairiehealth/Implement-Notification-System-b533e71ecf7749e786698b8b0b441970">https://www.notion.so/prairiehealth/Implement-Notification-System-b533e71ecf7749e786698b8b0b441970</a>	12/8/2021	Not start
<a href="#">Implement APIs for Main Patient Workflow</a>		@December 6, 2021 → December 10, 2021	<a href="https://www.notion.so/prairiehealth/Implement-APIs-for-Main-Patient-Workflow-8ef5cdb8ef8d4467a36388464da516dc">https://www.notion.so/prairiehealth/Implement-APIs-for-Main-Patient-Workflow-8ef5cdb8ef8d4467a36388464da516dc</a>	12/10/2021	Not start

Name	Assignee	Date	Description	Due Date	Section/C
<u>Implement Frontend for Patient Workflow (Mindvitals Lite)</u>		@December 13, 2021 → December 17, 2021	<a href="https://www.notion.so/prairiehealth/Implement-Frontend-for-Patient-Workflow-Mindvitals-Lite-b923302b84b54eeba257c1081179035e">https://www.notion.so/prairiehealth/Implement-Frontend-for-Patient-Workflow-Mindvitals-Lite-b923302b84b54eeba257c1081179035e</a>	12/17/2021	Not start
<u>Implement Frontend for Provider Workflow</u>		@December 13, 2021 → December 17, 2021	<a href="https://www.notion.so/prairiehealth/Implement-Frontend-for-Provider-Workflow-6b43ad072a974e828a02d0701f9e69ad">https://www.notion.so/prairiehealth/Implement-Frontend-for-Provider-Workflow-6b43ad072a974e828a02d0701f9e69ad</a>	12/17/2021	Not start
<u>Implement Frontend for Patient Detail Page</u>		@December 13, 2021 → December 17, 2021	<a href="https://www.notion.so/prairiehealth/Implement-Frontend-for-Patient-Detail-Page-3c76855188c149628f3b7daea47b4fc6">https://www.notion.so/prairiehealth/Implement-Frontend-for-Patient-Detail-Page-3c76855188c149628f3b7daea47b4fc6</a>	12/17/2021	Not start
<u>Implement Automation for adding new clinical databases</u>		@December 13, 2021 → December 14, 2021	<a href="https://www.notion.so/prairiehealth/Implement-Automation-for-adding-new-clinical-databases-748561a163cc40ff91555cd8a243a80b">https://www.notion.so/prairiehealth/Implement-Automation-for-adding-new-clinical-databases-748561a163cc40ff91555cd8a243a80b</a>	12/14/2021	Not start
<u>Add Monitoring</u>		@December 15, 2021	<a href="https://www.notion.so/prairiehealth/Add-Monitoring-f1b460acf1e14473a811944d7f10a3b8">https://www.notion.so/prairiehealth/Add-Monitoring-f1b460acf1e14473a811944d7f10a3b8</a>	12/15/2021	Not start
<u>User testing</u>		@December 20, 2021 → December 23, 2021	<a href="https://www.notion.so/prairiehealth/User-testing-5c04c52601c74640a0e886065dcc74aa">https://www.notion.so/prairiehealth/User-testing-5c04c52601c74640a0e886065dcc74aa</a>	12/23/2021	Not start
<u>Corrections</u>		@December 21, 2021 → December 29, 2021	<a href="https://www.notion.so/prairiehealth/Corrections-fc1f5c441e6346eda49b42d2c16a4ac8">https://www.notion.so/prairiehealth/Corrections-fc1f5c441e6346eda49b42d2c16a4ac8</a>	12/29/2021	Not start

## Notes

### Proposed development steps

The Main Provider workflow should be finished first. After that, the patient workflow and admin dashboards can be completed in parallel.

### Potential blockers & biggest hurdles

Deciding how this will be deployed to clinics will require more research/discussion with engineering team. Some things on my mind

### Outstanding questions that need to be answered

#### Out of scope for MVP

- For role based authentication, we need to clarify the privilege levels for different users at the clinics/prairie.
- We need to clarify the onboarding flow for new clinics and users a potential flow is proposed in this document. It still requires mockups

Personnel Management workflow (No mockups yet).

Potential Clinic Onboarding Flow (No mockups yet).