Playground

Exported from IcePanel at 2024-02-22 21:55:00

Table of contents

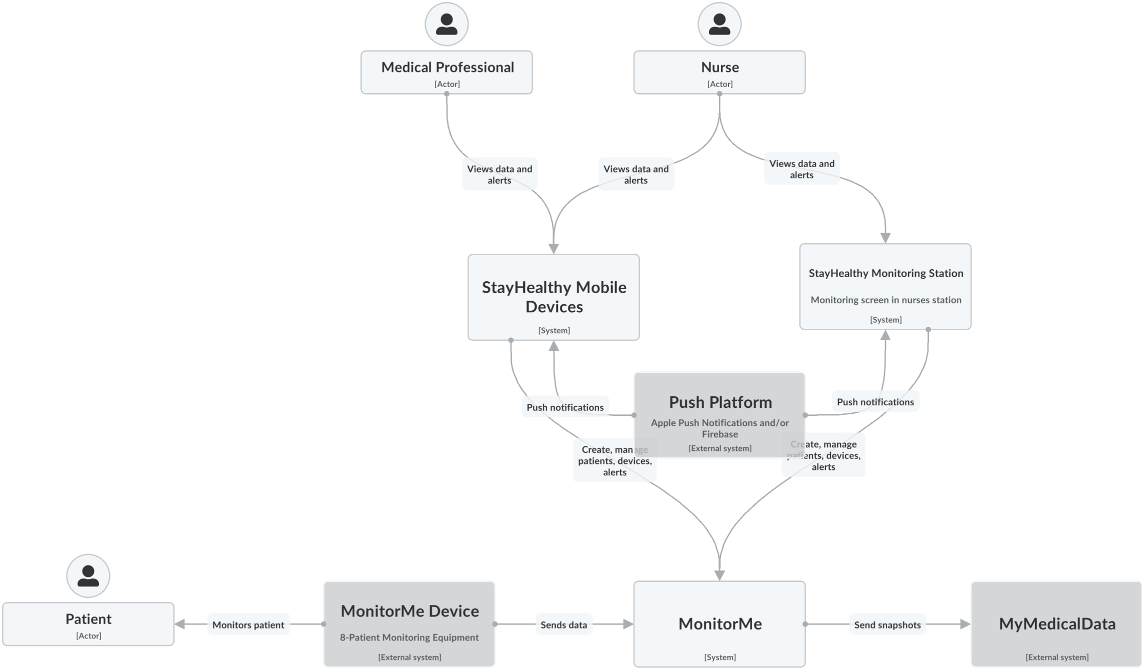
- 1. Context diagrams
  - 1a. Monitor Me - Arch Kata
- 2. Areas
  - 2.1. MonitorMed
- 3. Actors
  - 3.1. Medical Professional
  - 3.2. Nurse
  - 3.3. Patient
- 4. Systems
  - 4.1. Monitoring Device
  - 4.2. MonitorMe
    - 4.2.1. App diagrams
      - 4.2.1a. MonitorMe - App Diagram
      - 4.2.1b. Nurse assigns medical professional for alerting purposes
      - 4.2.1c. Nurse monitors vitals
      - 4.2.1d. Medical Professional and/or nurse receive alert(s)
      - 4.2.1e. MonitorMe device(s) monitor patient
      - 4.2.1f. Medical Professional and/or nurse pulls snapshot of patient vitals
      - 4.2.1g. Nurse onboards patient with devices
      - 4.2.1h. Medical Professional and/or nurse configures alerts
    - 4.2.2. Apps
      - 4.2.2.1. Alerts
      - 4.2.2.2. API Gateway
      - 4.2.2.3. Devices

    - 4.2.2.4. MonitorMe Event Broker
    - 4.2.2.5. Patients
      - 4.2.2.5.1. Component diagrams
    - 4.2.2.6. Raw Vitals
    - 4.2.2.7. Snapshots
    - 4.2.2.8. Subscriptions
    - 4.2.2.9. Vital Events
  - 4.2.3. Stores
    - 4.2.3.1. MonitorMe (Database)
      - 4.2.3.1.1. Component diagrams
        - 4.2.3.1.1a. Diagram 1
      - 4.2.3.1.2. Components
        - 4.2.3.1.2a. Alert Rules (Table)
        - 4.2.3.1.2b. Alerts (Table)
        - 4.2.3.1.2c. Devices (Table)
        - 4.2.3.1.2d. Patients (Table)
        - 4.2.3.1.2e. Patients_Alert_Rules (Table)
        - 4.2.3.1.2f. Patients_Device (Table)
        - 4.2.3.1.2g. Snapshots
        - 4.2.3.1.2h. Subscriptions (Table)
        - 4.2.3.1.2i. Users (Table)
    - 4.2.3.2. MonitorMe Raw Vitals (Database)
      - 4.2.3.2.1. Component diagrams
        - 4.2.3.2.1a. MonitorMe Raw Vitals (Database) Component Diagram
      - 4.2.3.2.2. Components
        - 4.2.3.2.2a. Raw Vitals (Table)
- 4.3. MonitorMe Device
  - 4.3.1. App diagrams

- 4.4. MyMedicalData
- 4.5. Push Platform
- 4.6. StayHealthy Mobile Devices
    - 4.6.1. App diagrams
        - 4.6.1a. Diagram 1
    - 4.6.2. Apps
        - 4.6.2.1. StayHealthy Android
        - 4.6.2.2. StayHealthy iOS
- 4.7. StayHealthy Monitoring Station
    - 4.7.1. App diagrams
        - 4.7.1a. StayHealthy Monitoring Station App Diagram
    - 4.7.2. Apps
        - 4.7.2.1. Monitoring Station App

1. Context diagrams

1a. Monitor Me - Arch Kata



2. Areas

2.1. MonitorMed

No description

3. Actors

3.1. Medical Professional

No description

3.2. Nurse

No description

3.3. Patient

No description

4. Systems

4.1. Monitoring Device

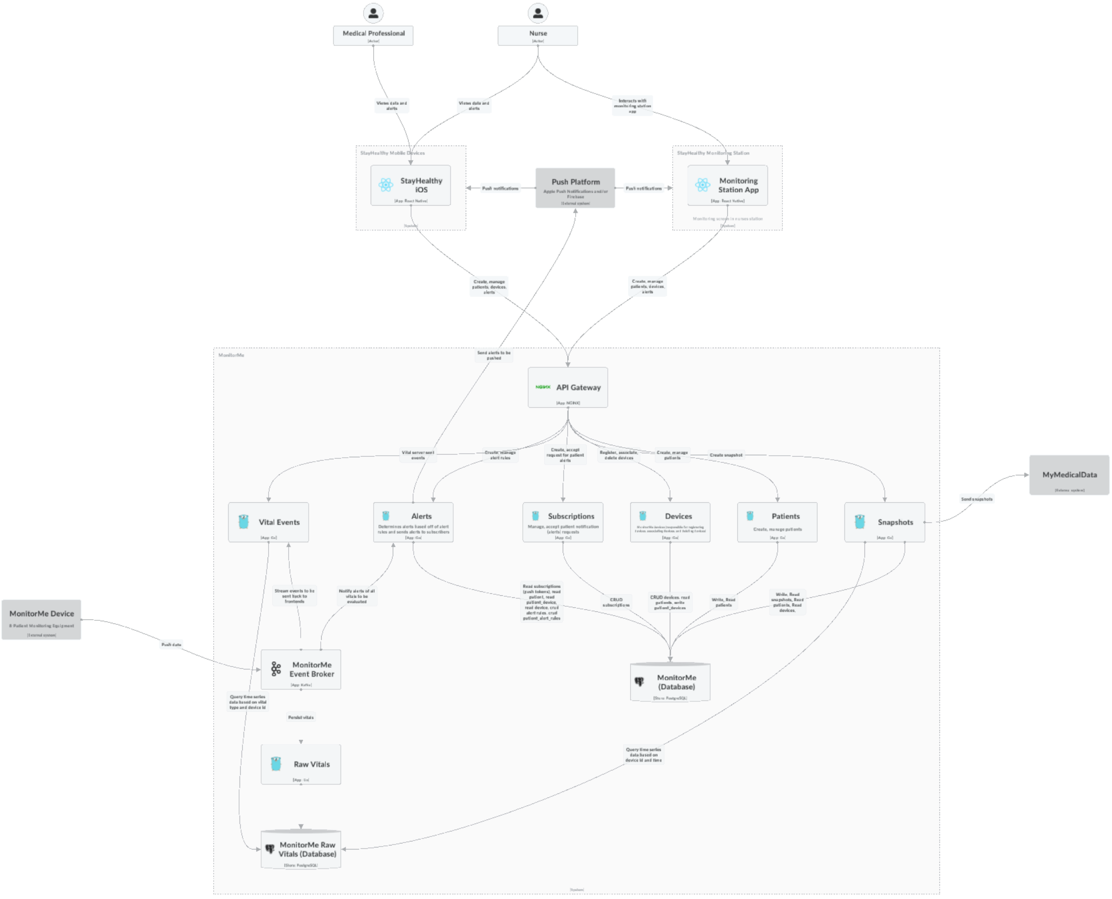
No description

4.2. MonitorMe

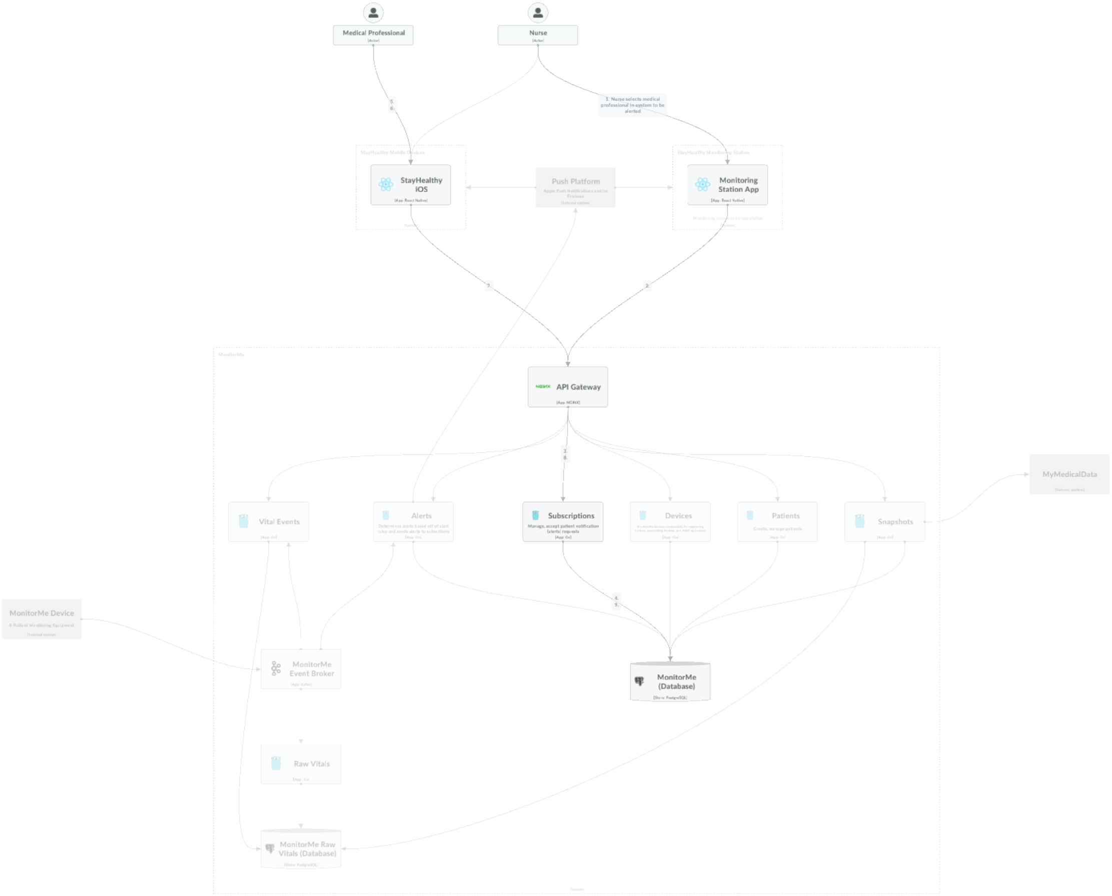
No description

4.2.1. App diagrams

4.2.1a. MonitorMe - App Diagram


Medical Professional
[Actor]
Nurse
[Actor]
Views data and alerts
Views data and alerts
Interacts with monitoring station app
StayHealthy Mobile Devices
StayHealthy iOS
[App: React Native]
[System]
Push notifications
Push Platform
Apple Push Notifications and/or Firebase
[External system]
Push notifications
StayHealthy Monitoring Station
Monitoring Station App
[App: React Native]
Monitoring screen in nurses station
[System]
Create, manage patients, devices, alerts
Create, manage patients, devices, alerts
MonitorMe
Send alerts to be pushed
API Gateway
[App: NGINX]
Vital server sent events
Create, manage alert rules
Create, accept request for patient alerts
Register, associate, delete devices
Create, manage patients
Create snapshot
Vital Events
[App: Go]
Alerts
Determines alerts based off of alert rules and sends alerts to subscribers
[App: Go]
Subscriptions
Manage, accept patient notification (alerts) requests
[App: Go]
Devices
MonitorMe devices (responsible for registering devices, associating devices, and deleting devices)
[App: Go]
Patients
Create, manage patients
[App: Go]
Snapshots
[App: Go]
Send snapshots
MyMedicalData
[External system]
Stream events to be sent back to frontends
Notify alerts of all vitals to be evaluated
Read subscriptions (push tokens), read patient, read patient_device, read device, crud alert rules, crud patient_alert_rules
CRUD subscriptions
CRUD devices, read patients, write patient_devices
Write, Read patients
Write, Read snapshots, Read patients, Read devices,
MonitorMe Device
8 Patient Monitoring Equipment
[External system]
Push data
MonitorMe Event Broker
[App: Kafka]
MonitorMe (Database)
[Store: PostgreSQL]
Query time series data based on vital type and device id
Persist vitals
Raw Vitals
[App: Go]
Query time series data based on device id and time
MonitorMe Raw Vitals (Database)
[Store: PostgreSQL]
[System]

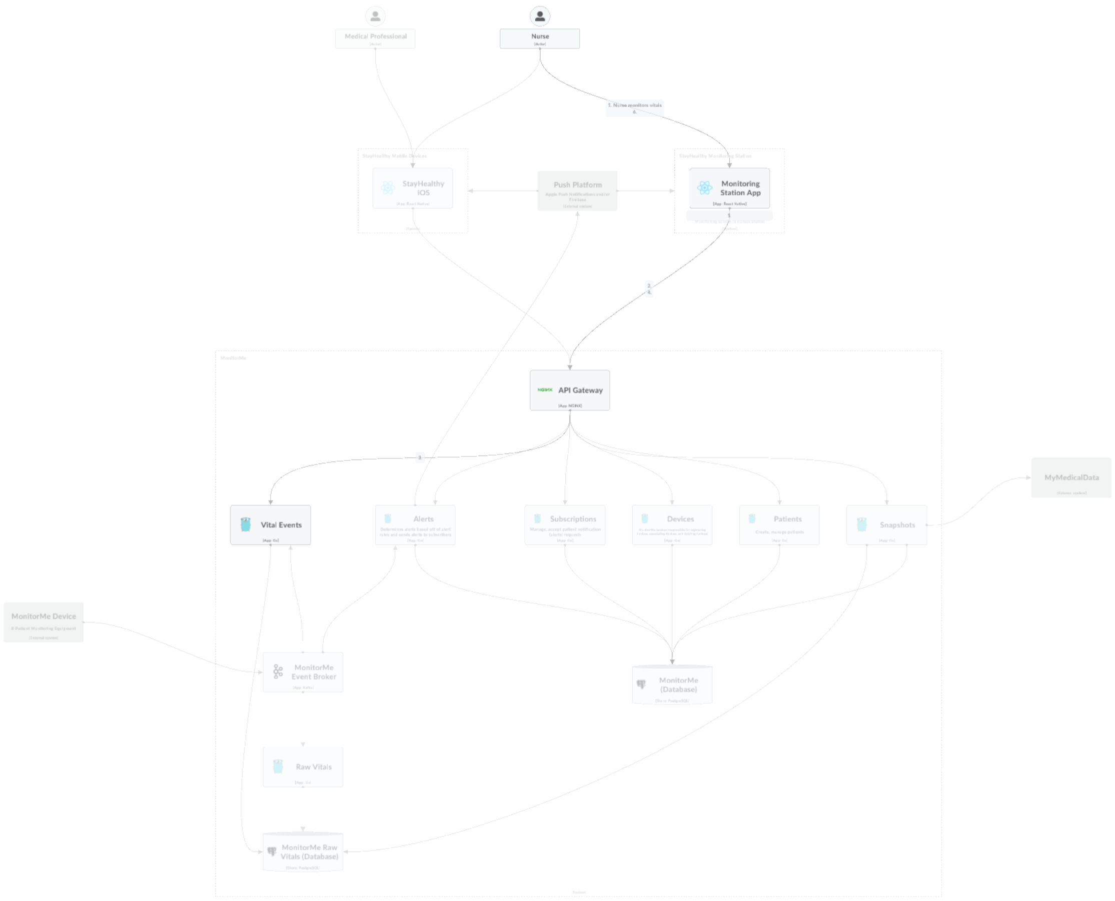
4.2.1b. Nurse assigns medical professional for alerting purposes



- Step 1: Nurse -> Monitoring Station App: Nurse selects medical professional in system to be alerted via Interacts with monitoring station app
- Step 2: Monitoring Station App -> API Gateway: App sends request to create a patient notification request via Create, manage patients, devices, alerts
- Step 3: API Gateway -> Subscriptions: Gateway sends request to subscriptions service via Create, accept request for patient alerts
- Step 4: Subscriptions -> MonitorMe (Database): Subscriptions service create subscription request for medical professional via CRUD subscriptions
- Step 5: Medical Professional -> StayHealthy iOS: Medical professional logs into the app via Views data and alerts
- Step 6: Medical Professional -> StayHealthy iOS: Medical professional accepts patient to monitor via Views data and alerts
- Step 7: StayHealthy iOS -> API Gateway: App sends push token along with the subscription request to system via Create, manage patients, devices, alerts
- Step 8: API Gateway -> Subscriptions: Gateway sends request to subscriptions via Create, accept request for patient alerts

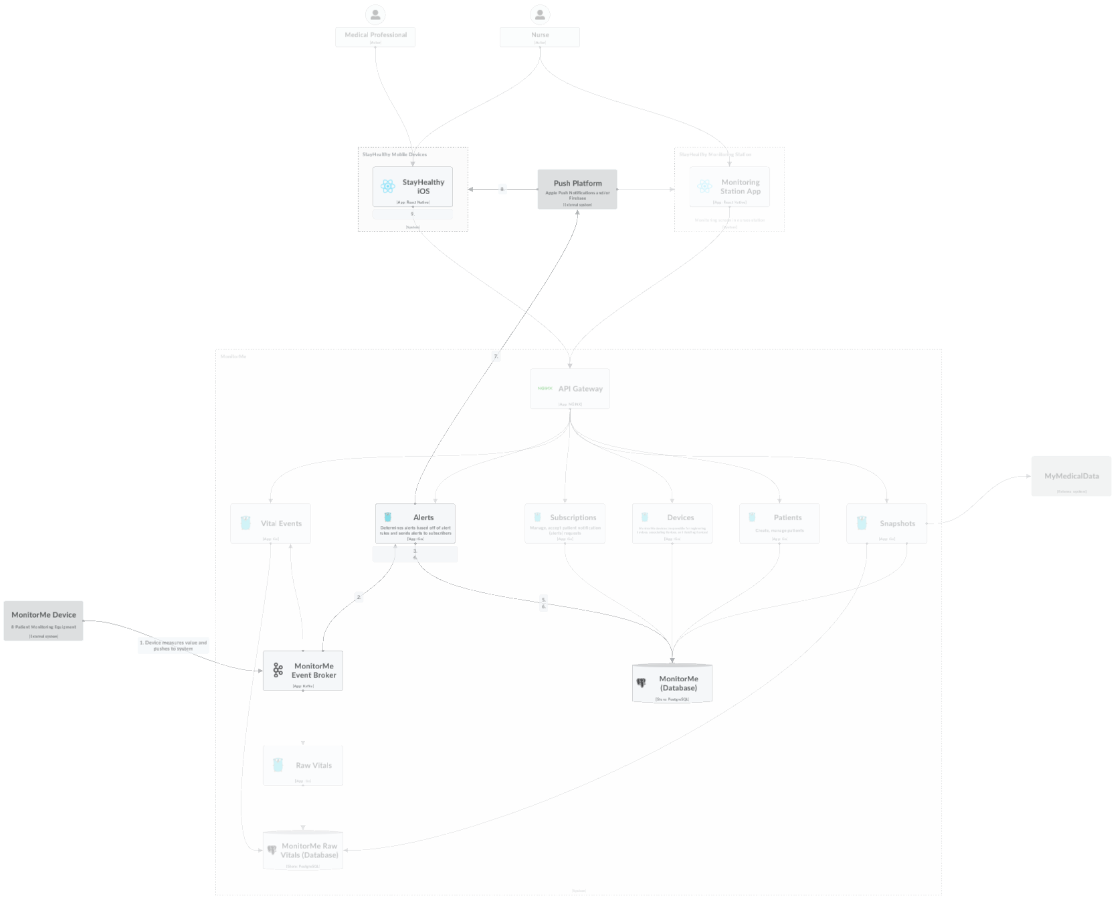
- Step 9: Subscriptions -> MonitorMe (Database): Subscriptions microservice saves push token in subscriptions table via CRUD subscriptions

4.2.1c. Nurse monitors vitals



- Step 1: Nurse -> Monitoring Station App: Nurse monitors vitals via Interacts with monitoring station app
- Step 2: Monitoring Station App -> API Gateway: Sends request to see vitals via Create, manage patients, devices, alerts
- Step 3: API Gateway -> Vital Events: Vitals creates server sent events for each vital via Vital server sent events
- Step 4: API Gateway -> Monitoring Station App: App receives events and refreshes display via Create, manage patients, devices, alerts
- Step 5: Monitoring Station App: App determines patient's devices
- Step 6: Monitoring Station App -> Nurse: App shows nurse vitals for patient via Interacts with monitoring station app

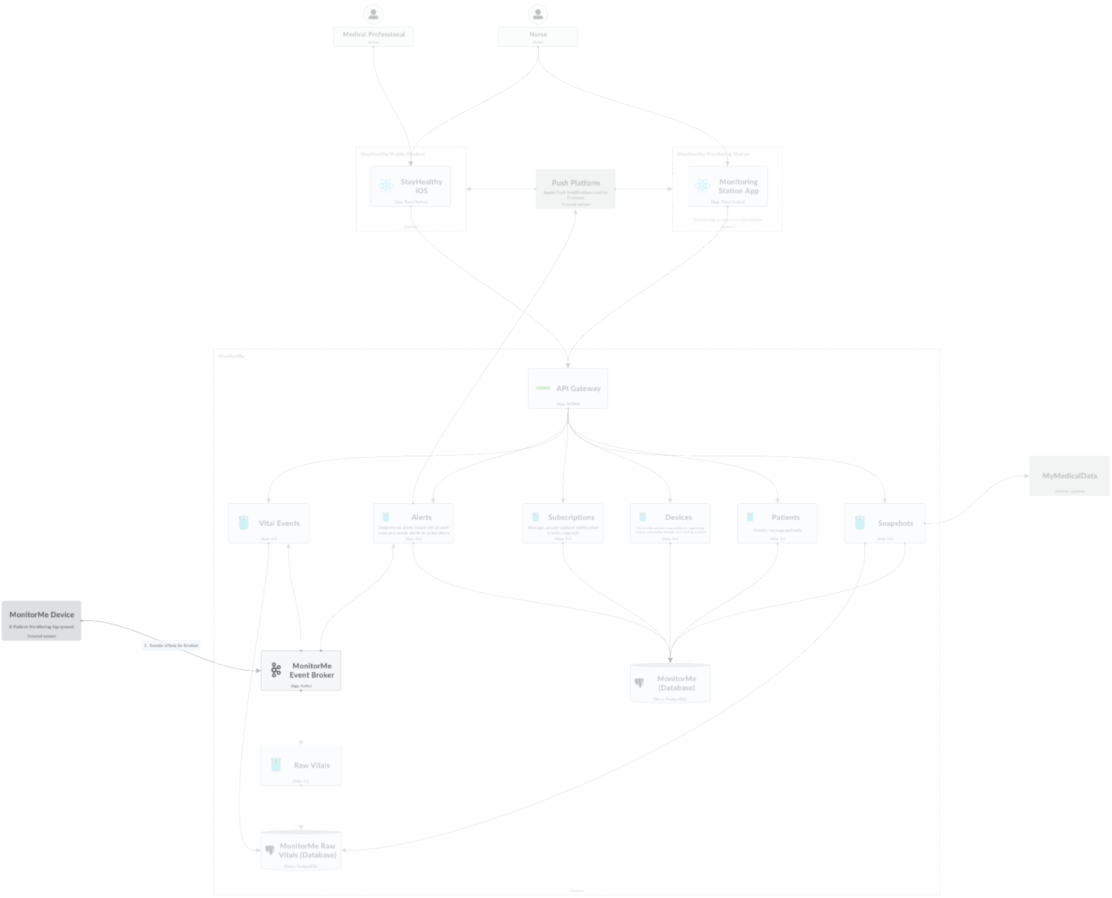
4.2.1d. Medical Professional and/or nurse receive alert(s)



- Step 1: MonitorMe Device -> MonitorMe Event Broker: Device measures value and pushes to system via Push data
- Step 2: MonitorMe Event Broker -> Alerts: Broker sends vital to alerts via Notify alerts of all vitals to be evaluated
- Step 3: Alerts: Alert checks rules
- Step 4: Alerts: Alert evaluates vital against rules
- Step 5: Alerts -> MonitorMe (Database): Alert saves alert in database via Read subscriptions (push tokens), read patient, read patient_device, read device, crud alert rules, crud patient_alert_rules
- Step 6: Alerts -> MonitorMe (Database): Alert checks subscriptions via Read subscriptions (push tokens), read patient, read patient_device, read device, crud alert rules, crud patient_alert_rules
- Step 7: Alerts -> Push Platform: Alert sends alert via Send alerts to be pushed
- Step 8: Push Platform -> StayHealthy Mobile Devices: Push platform sends notification to device via Push notifications

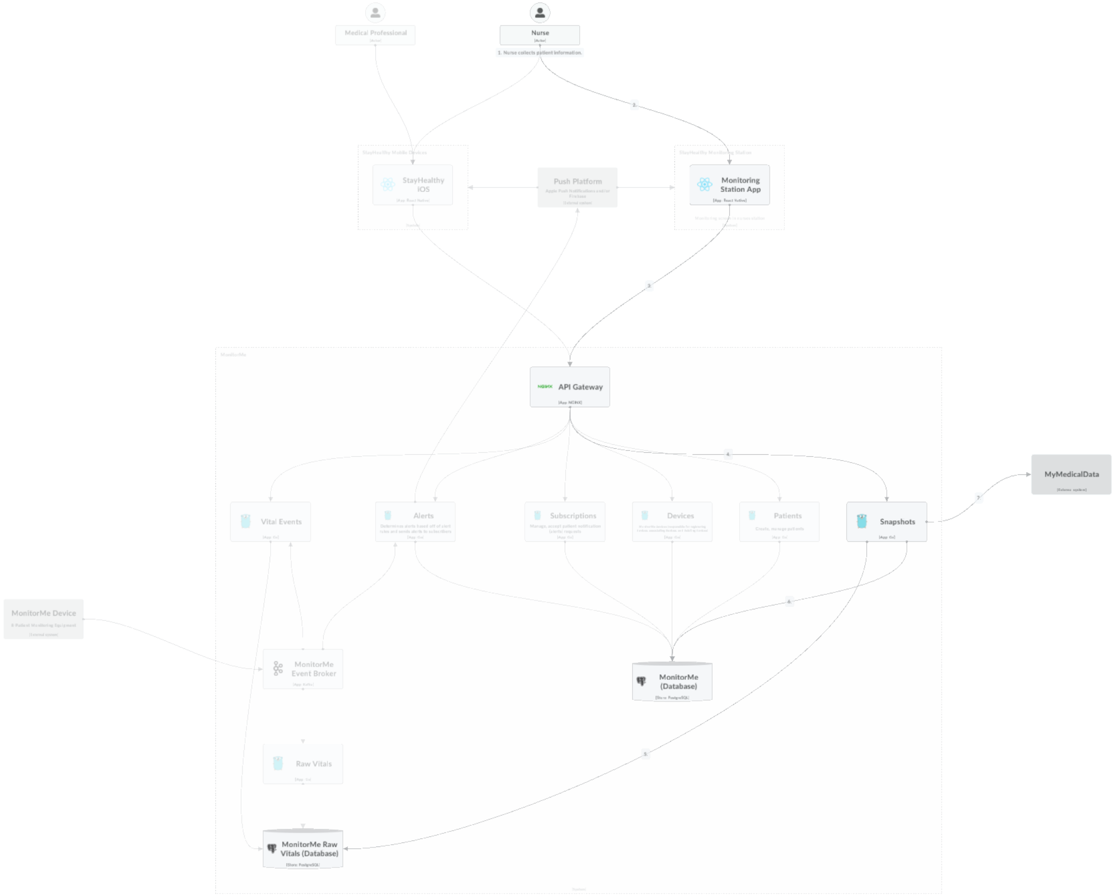
- Step 9: StayHealthy iOS: iOS alerts medical professional

4.2.1e. MonitorMe device(s) monitor patient



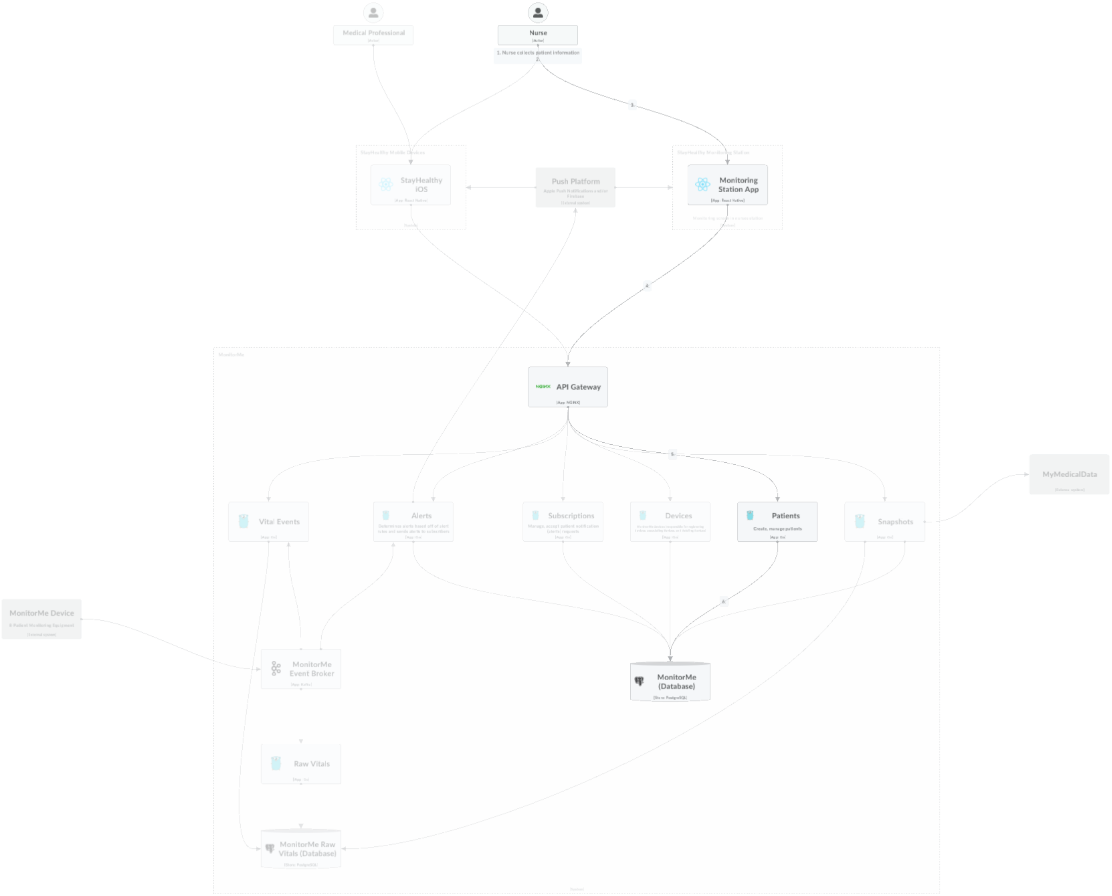
- Step 1: MonitorMe Device -> MonitorMe Event Broker: Sends vitals to broker via Push data
- Parallel path: Send vitals to consumers
    - Step 1: MonitorMe Event Broker -> Raw Vitals: Broker sends vitals to Raw Vital service via Persist vitals
    - Step 2: MonitorMe Event Broker -> Alerts: Broker sends vitals to Alerts service via Notify alerts of all vitals to be evaluated
    - Step 3: MonitorMe Event Broker -> Vital Events: Broker streams vitals to Vital Events via Stream events to be sent back to frontends

4.2.1f. Medical Professional and/or nurse pulls snapshot of patient vitals



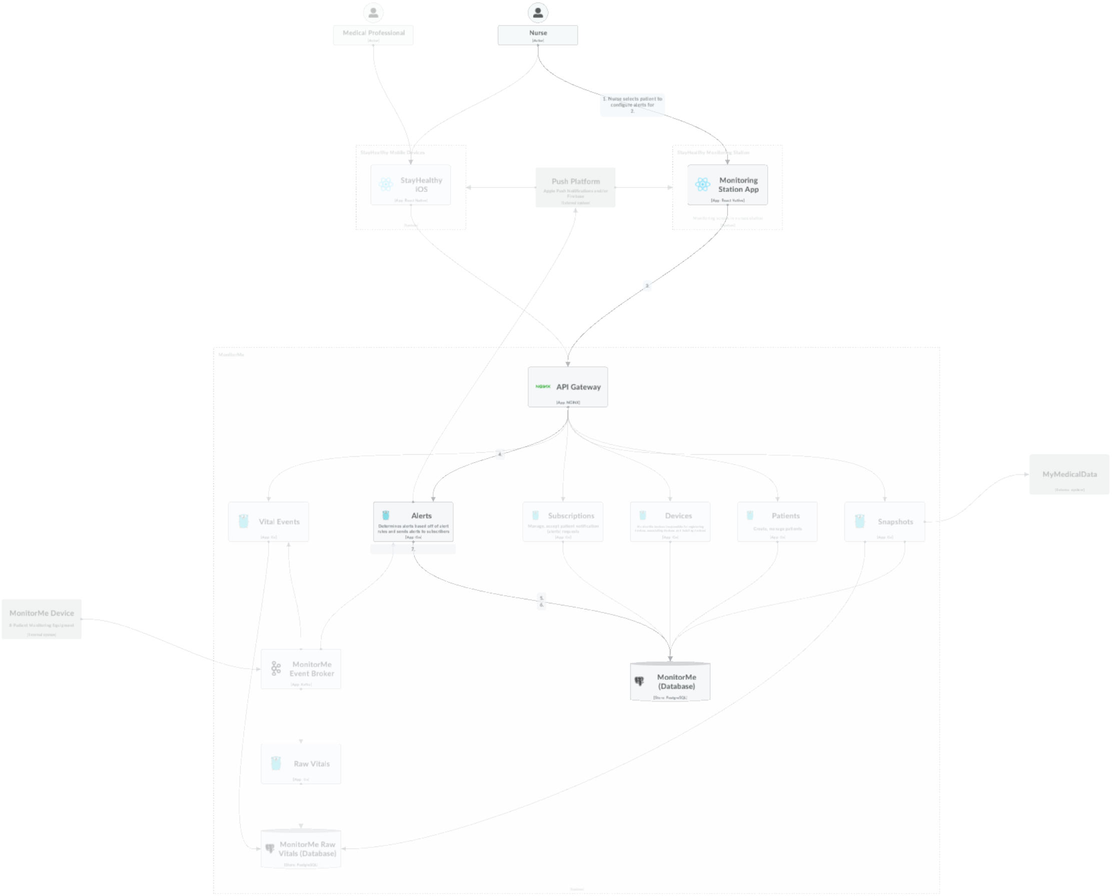
- Step 1: Nurse: Nurse collects patient information.
- Step 2: Nurse -> Monitoring Station App: Initiates snapshot creation. via Interacts with monitoring station app
- Step 3: Monitoring Station App -> API Gateway: Makes the snapshot creation request. via Create, manage patients, devices, alerts
- Step 4: API Gateway -> Snapshots: Request is sent to Snapshots service. via Create snapshot
- Step 5: Snapshots -> MonitorMe Raw Vitals (Database): Patient's raw vitals are queried. via Query time series data based on device id and time
- Step 6: Snapshots -> MonitorMe (Database): Snapshot event is saved to the database. via Write, Read snapshots, Read patients, Read devices,
- Step 7: Snapshots -> MyMedicalData: Snapshot is sent to MyMedicalData. via Send snapshots

4.2.1g. Nurse onboards patient with devices



- Step 1: Nurse: Nurse collects patient information
- Step 2: Nurse: Nurse prepares devices for patient
- Step 3: Nurse -> Monitoring Station App: Nurse registers patient with devices on the Monitoring Station App via Interacts with monitoring station app
- Step 4: Monitoring Station App -> API Gateway: App sends patient and device registration details to MonitorMe via Create, manage patients, devices, alerts
- Step 5: API Gateway -> Patients: Gateway sends request to Patients microservice via Create, manage patients
- Step 6: Patients -> MonitorMe (Database): Patients microservice creates patient, creates new device if not existing, relates patient to device via Write, Read patients

4.2.1h. Medical Professional and/or nurse configures alerts



- Step 1: Nurse -> Monitoring Station App: Nurse selects patient to configure alerts for via Interacts with monitoring station app
- Step 2: Nurse -> Monitoring Station App: Nurse configures alerts based on vital signs via Interacts with monitoring station app
- Step 3: Monitoring Station App -> API Gateway: App sends request via Create, manage patients, devices, alerts
- Step 4: API Gateway -> Alerts: Gateway sends request to alerts microservice via Create, manage alert rules
- Step 5: Alerts -> MonitorMe (Database): Alert service save new alert rules via Read subscriptions (push tokens), read patient, read patient_device, read device, crud alert rules, crud patient_alert_rules
- Step 6: Alerts -> MonitorMe (Database): Alert service associates alert rule with patient via Read subscriptions (push tokens), read patient, read patient_device, read device, crud alert rules, crud patient_alert_rules
- Step 7: Alerts: Alert rules are synced in memory

4.2.2. Apps

4.2.2.1. Alerts

Determines alerts based off of alert rules and sends alerts to subscribers

Golang

- Create alert rules given patient
- Send alerts to push platform
-

4.2.2.2. API Gateway

NGINX

No description

4.2.2.3. Devices

MonitorMe devices (responsible for registering devices, associating devices, and deleting devices)

Golang

- Registers devices
- Associate device to patient
- Delete device

4.2.2.4. MonitorMe Event Broker

Apache Kafka

No description

4.2.2.5. Patients

Create, manage patients

Golang

- Create new patient

4.2.2.5.1. Component diagrams

4.2.2.6. Raw Vitals

Golang

No description

4.2.2.7. Snapshots

Golang

- Create snapshot for patient
  - Look up patient and their devices
  - Look up the patient's usage of device timeframe
  - Look up raw vitals
  - Package and send data

4.2.2.8. Subscriptions

Manage, accept patient notification (alerts) requests

Golang

- Create request for patient notifications
- Accept medical professional request for patient notifications
  - Receives push token
-

4.2.2.9. Vital Events

Golang

- Reads raw vitals back to applications
  - Can be filtered based on vitals and patient/device ID
-

4.2.3. Stores

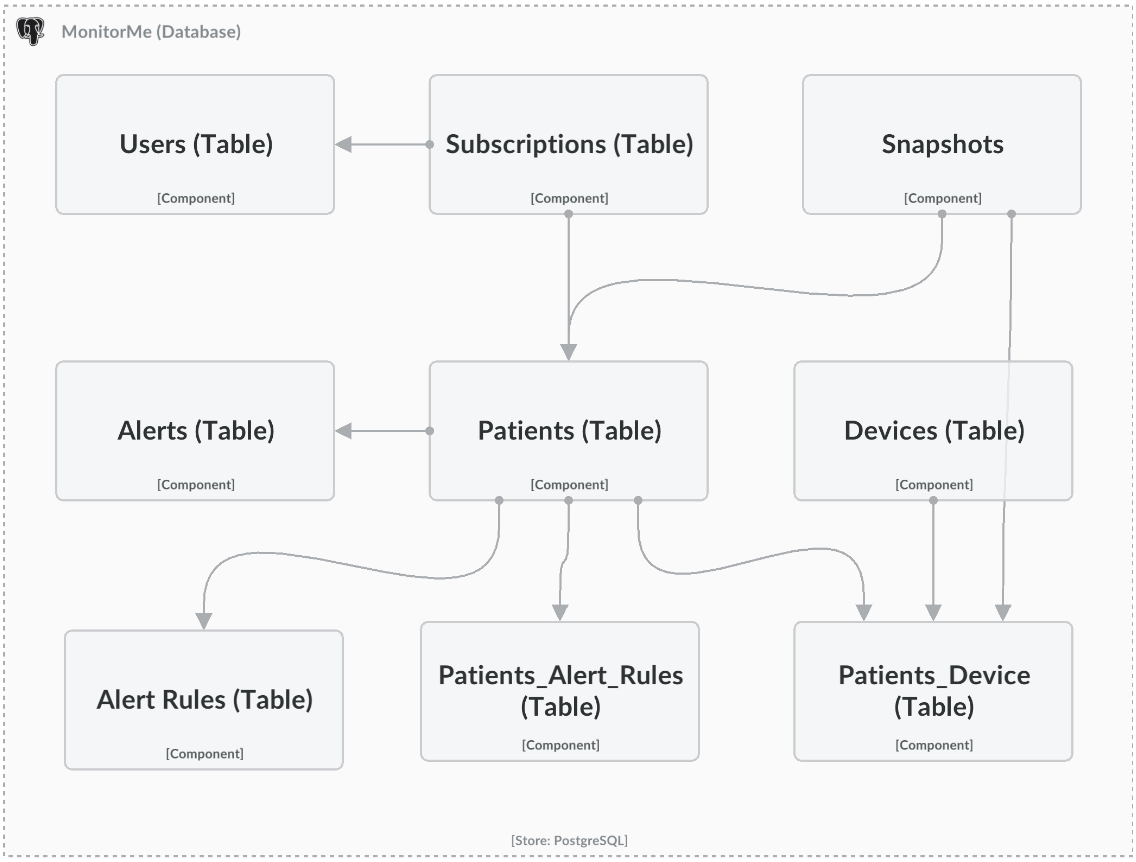
4.2.3.1. MonitorMe (Database)

PostgreSQL

No description

4.2.3.1.1. Component diagrams

4.2.3.1.1a. Diagram 1



4.2.3.1.2. Components

4.2.3.1.2a. Alert Rules (Table)

No description

4.2.3.1.2b. Alerts (Table)

No description

4.2.3.1.2c. Devices (Table)

No description

4.2.3.1.2d. Patients (Table)

No description

4.2.3.1.2e. Patients_Alert_Rules (Table)

No description

4.2.3.1.2f. Patients_Device (Table)

No description

4.2.3.1.2g. Snapshots

No description

4.2.3.1.2h. Subscriptions (Table)

No description

4.2.3.1.2i. Users (Table)

No description

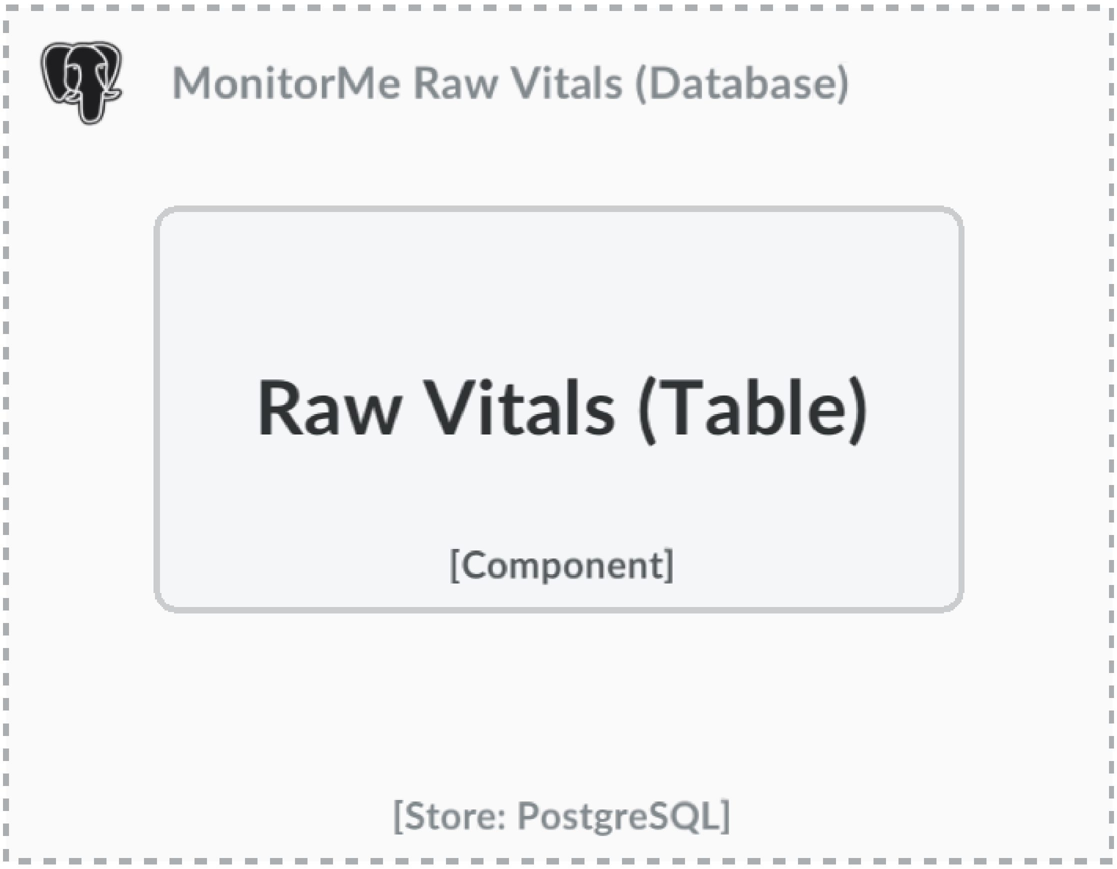
4.2.3.2. MonitorMe Raw Vitals (Database)

PostgreSQL

No description

4.2.3.2.1. Component diagrams

4.2.3.2.1a. MonitorMe Raw Vitals (Database) Component Diagram



4.2.3.2.2. Components

4.2.3.2.2a. Raw Vitals (Table)

- Device ID
- Timestamp
- Device Type (Vital type)
- Value

4.3. MonitorMe Device

External

8-Patient Monitoring Equipment

4.3.1. App diagrams

4.4. MyMedicalData

External

No description

4.5. Push Platform

External

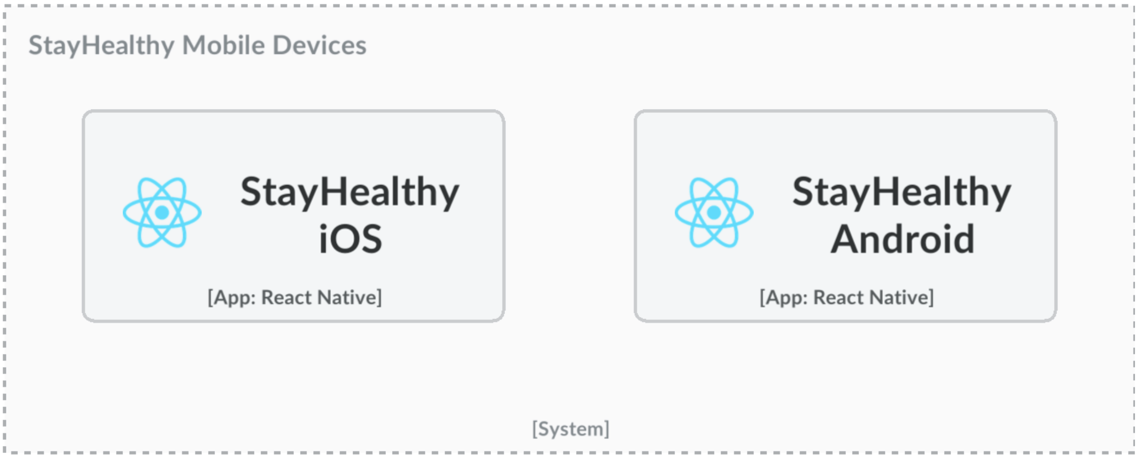
Apple Push Notifications and/or Firebase

4.6. StayHealthy Mobile Devices

No description

4.6.1. App diagrams

4.6.1a. Diagram 1



4.6.2. Apps

4.6.2.1. StayHealthy Android

React Native

No description

4.6.2.2. StayHealthy iOS

React Native

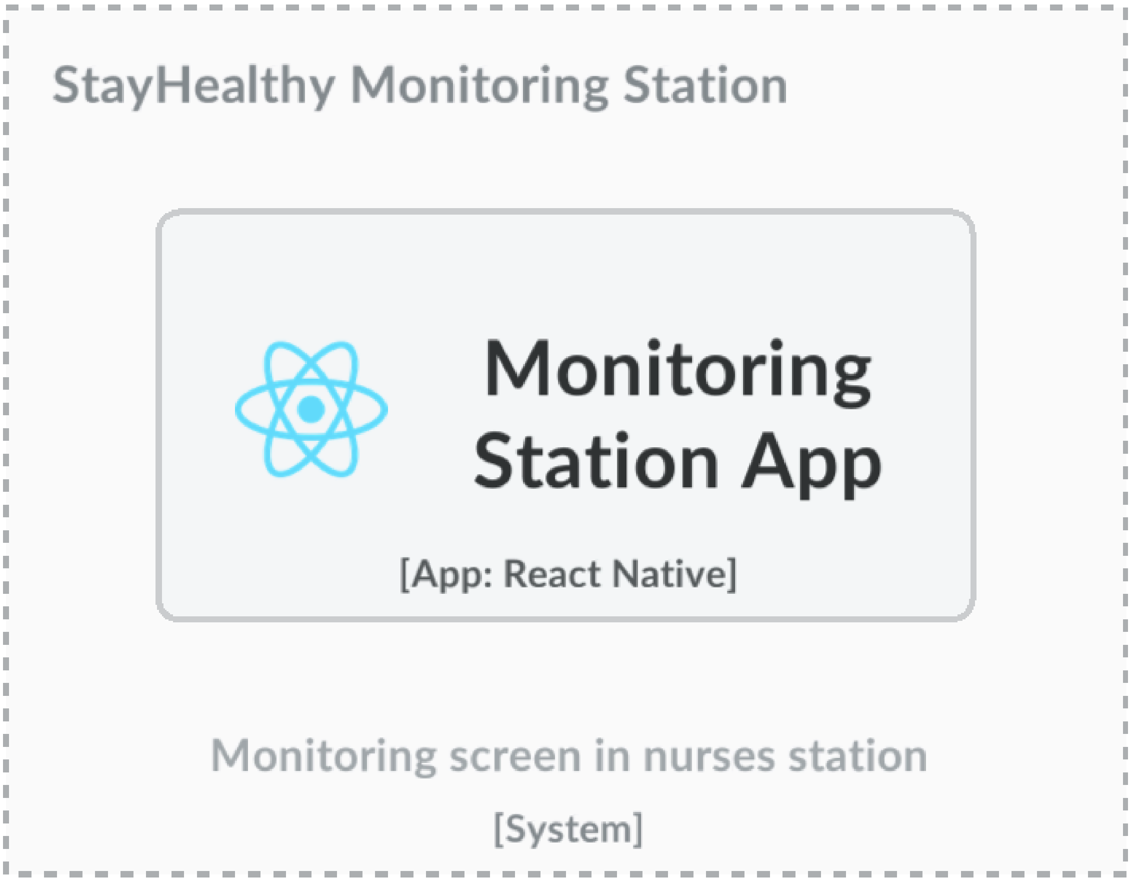
No description

4.7. StayHealthy Monitoring Station

Monitoring screen in nurses station

4.7.1. App diagrams

4.7.1a. StayHealthy Monitoring Station App Diagram


StayHealthy Monitoring Station
Monitoring Station App
[App: React Native]
Monitoring screen in nurses station
[System]

4.7.2. Apps

4.7.2.1. Monitoring Station App

React Native

No description