How to Use this Template

1. Make a copy [File → Make a copy...]
2. Rename this file: "**Capstone_Stage1**"
3. Replace the text in green

Submission Instructions

1. After you've completed all the sections, download this document as a PDF [File → Download as PDF]
2. Create a new GitHub repo for the capstone. Name it "**Capstone Project**"
3. Add this document to your repo. Make sure it's named "**Capstone_Stage1.pdf**"

GitHub Username: janlep47

NurseHelper

Description

This app is intended for use by nursing homes and other long-term care facilities. The app acts as both a reminder and list of current care-plan and medications for each resident in the facility.

The app also is used to electronically chart all day-to-day care, including medications administered and assessments, for the residents of the health care facility.

Intended User

Users will be nurses and aides, which are employees of the nursing home that has adopted this app to keep track of patient care data. Other employees may use a separate component to maintain resident health-related data, which is kept on the facility's private database server. This component will be packaged with the app, but will not run from the NurseHelper app.

Features

This app has two main features. One feature is electronic charting. This makes charting much faster, in that it can be done almost immediately right at the point-of-care. This saves a lot of time for the nurses and aides, by eliminating the tedious chore of paper charting. In addition, information can be accessed easily and practically instantly, rather than having to go through pages of paper located far from the patients' rooms.

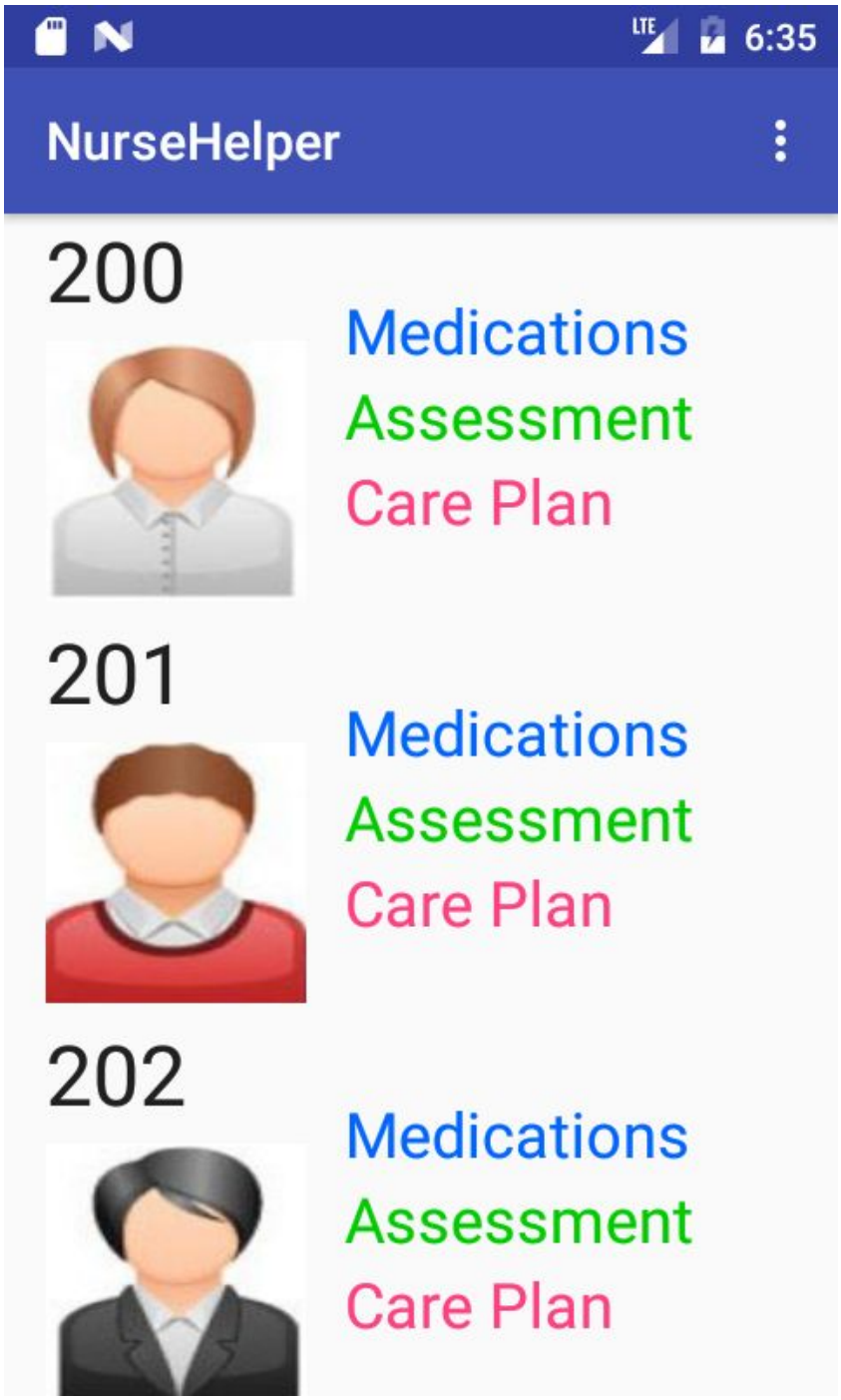
A second feature is that this app acts as medications-due reminder (also care-items due), to aid the nurses and aides in giving the best care to all the residents. As nurses in these facilities are very often tasked with the care of 24 patients at a time, this help can be vital.

- Saves all nurse/aide caregiving information by resident.
- Displays list of residents, showing room# and portrait of resident.
- Can click on resident within list to show current list of medications, care plan, and assessment page.
- Receives changes to care-plan, and medications database from facility's central server database, via Google cloud.
- Uploads local app database changes, e.g. medications administered and assessment results, to facility's central server.
- Allows entry of new patient information, including "portrait" (picture taken of new resident from nurse's android device). This data can be uploaded to the facility's central server database.

User Interface Mocks

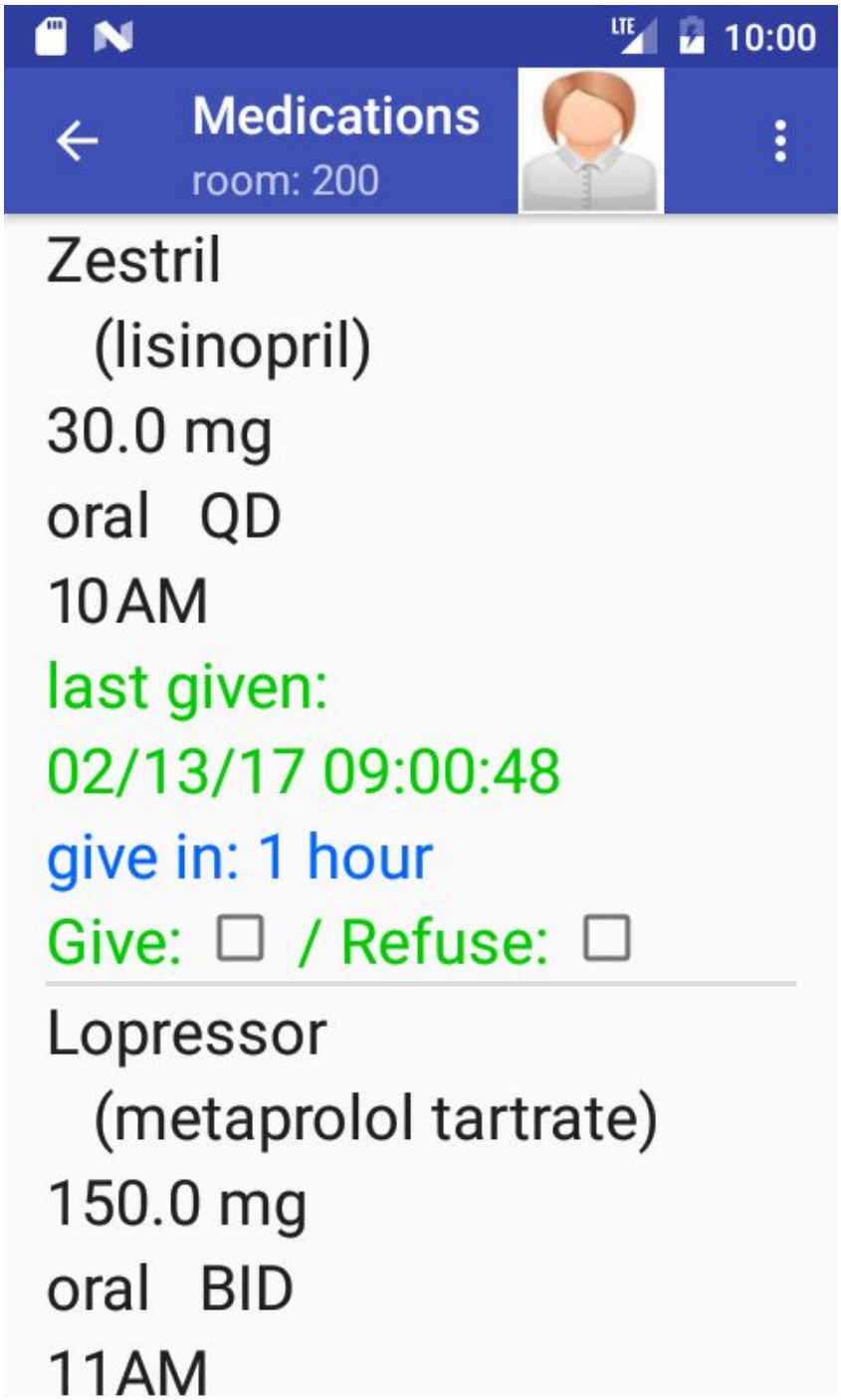
These can be created by hand (take a photo of your drawings and insert them in this flow), or using a program like Photoshop or Balsamiq.

Screen 1



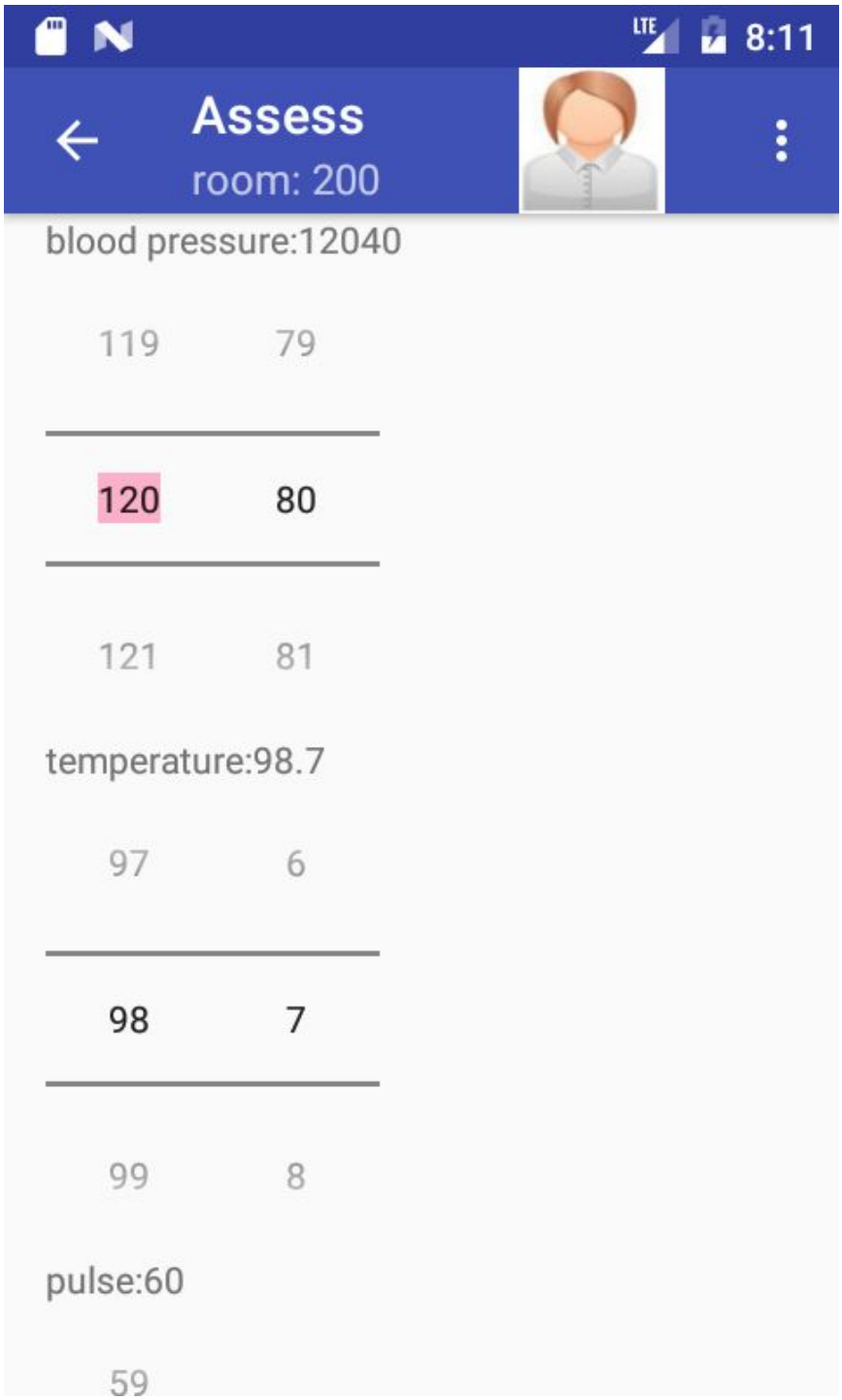
This is the initial screen, or “Resident list” screen. This appears after logging in with id and password. Screen 1 shows a list of all the care giver’s assigned patients. The nurse can choose either “Medications”, “Assessment”, or “Care Plan” next to the patient’s room# and image, to access that data relevant to the selected patient.

Screen 2



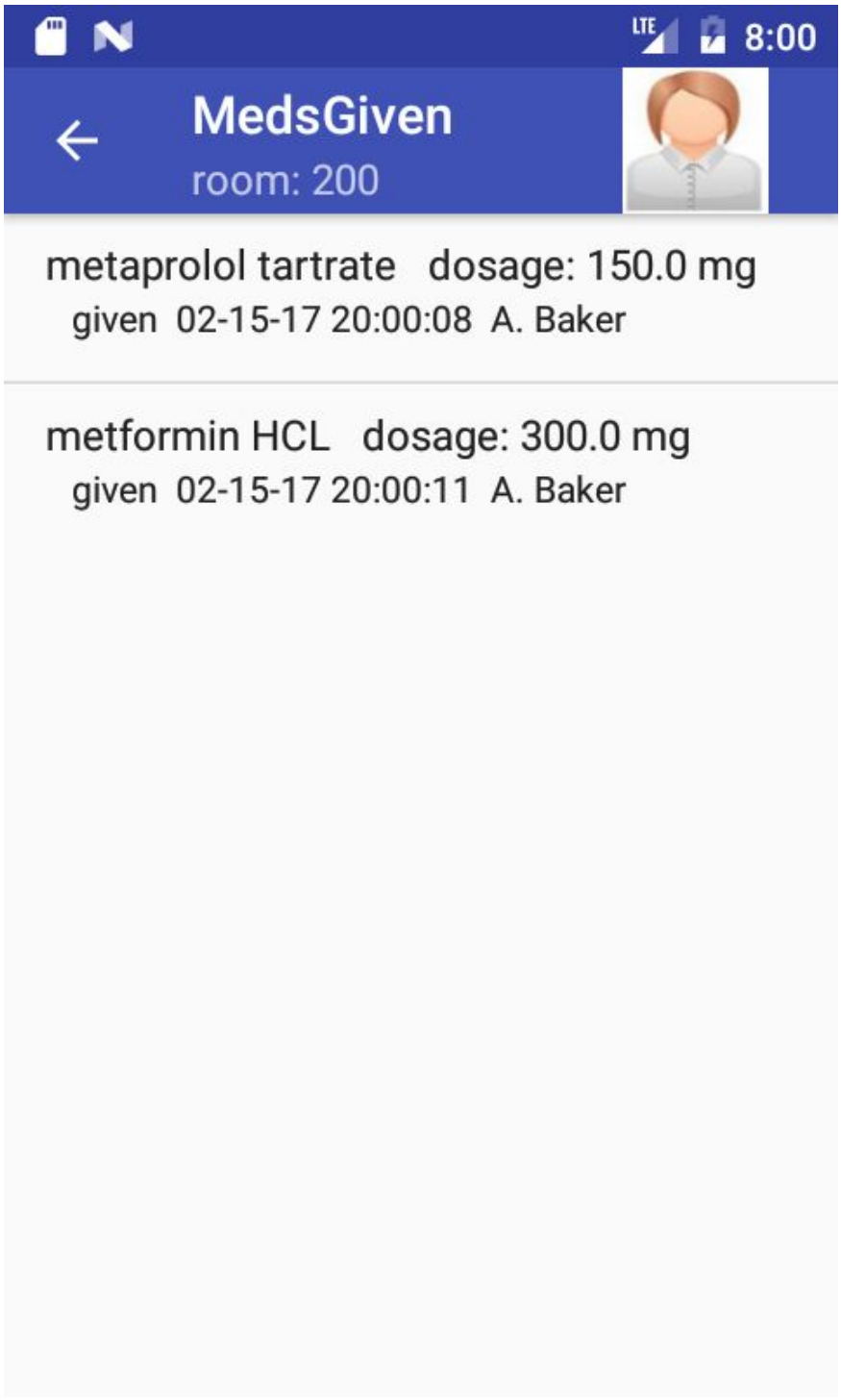
This screen shows a hypothetical list of current medications, for the selected patient - in this case, the patient in room 200, with the portrait shown. Screen 2 appears when the user presses “Medications” from screen 1 (Resident List screen).

Screen 3



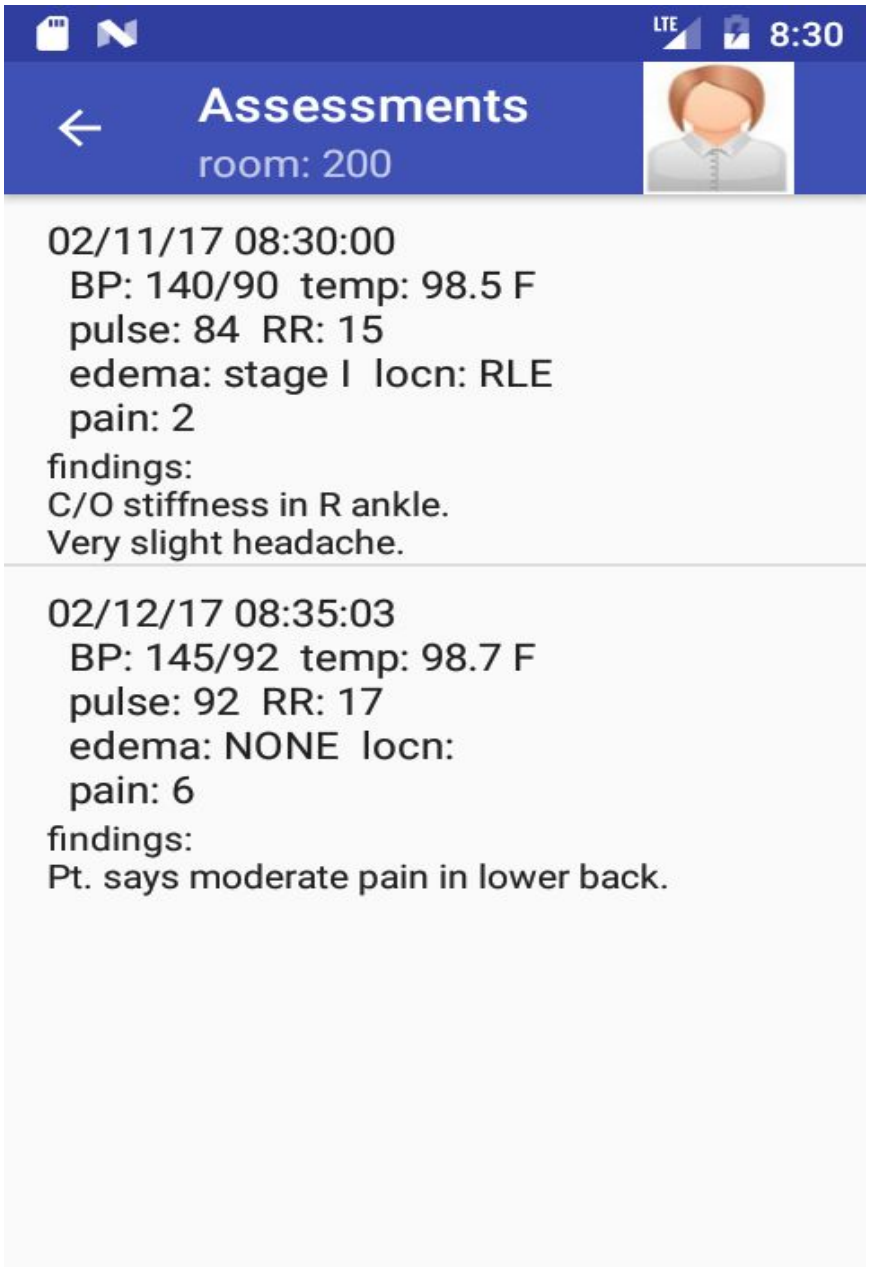
This screen allows the user to enter assessment data for the given patient. Screen 3 appears when the user presses “Assessment” from screen 1 (Resident List screen).

Screen 4



This screen show all medications administered to a given patient within some configurable past period of time. Screen 4 appears when the user presses the “show medications given” option from the menu in Screen 2 (Medications List screen)

Screen 5



This screen displays the all past assessments for a given patient, within some configurable past period of time. Screen 5 appears when user selects “show past assessments” from the menu option in Screen 2 (Assessment screen).

The final screen (Screen 6) will consist of an app bar similar to screen2-5, but will simply show an image of that current patient’s care plan. Screen 6 will appear when the user presses “Care Plan” from Screen 1 (the Resident List screen).

Add as many screens as you need to portray your app’s UI flow.

Key Considerations

How will your app handle data persistence?

This app includes a Content Provider to handle data persistence.

Describe any corner cases in the UX.

There are only two screens where data is entered by the user: screen 2 (Medications list screen) and screen 3 (Assessment screen). As the individual medications from screen 2 are checked either as given or refused, they are updated as such in the database. So there is no corner case for screen 2. Screen 3 does have a potential corner case: the assessment data entered on the screen, is not updated in the database until the user presses the “done” button. So, it’s possible that some data could be lost from this screen, if that activity somehow became stopped or destroyed. However, this is not a serious problem, as either the data could be re-entered - if remembered, or the assessment could be simply redone.

Describe any libraries you’ll be using and share your reasoning for including them.

The Picasso library is used to handle the loading and caching of resident images. This library is used to simplify coding.

Describe how you will implement Google Play Services.

One Google Play Services which will be used is Google sign-in “Identity”. The secure sign-in feature is vital to protect patient privacy. The app will run on devices assigned to a single user (nurse or aide), and only that user may have access to NurseHelper, on that device. Another Google Play Services that is useful is location, to verify that this app is only executable from the facilities whose data the app is using.

The Google Cloud Platform will also be used, to provide a backend which will sync data between the individual mobile charting devices, and the central private database server.

Next Steps: Required Tasks

This is the section where you can take the main features of your app (declared above) and decompose them into tangible technical tasks that you can complete incrementally until you have a finished app.

Task 1: Project Setup

Determine data needed for this app, and design database tables accordingly:

- Data needed for each screen/activity
- Data which will need to be configurable - shared preferences
- How data will be updated, uploaded, and downloaded to/from main private server database

Task 2: Implement UI for Each Activity and Fragment

Implement the UI and screen layouts, for both phone and tablet, for:

- User login screen
- MainActivity (Resident List - screen 1)
- Medications List (screen2)
- Assessment (screen3)
- Medications Given List (screen 4)
- Past Assessments List (screen 5)
- Care plan (screen 6)

Task 3: Google Play Services

Implement Google Play Services:

- Identity
- Location

Tie these services in with the related shared preference in the app.

Task 4: Show tasks and medications due

Design and implement alerts to tasks/medications due:

- Add configurable time periods to the list of shared preferences
- Add flags (alert yes/no) for types of alerts, to the list of shared preferences
- Implement a medications-due alert system, across all app activities, including the widget.
- Implement a task-due alert system, across all app activities, including the widget

- Implement a notification system, for downloaded changes from the private network database.

Task 5: Implement uploading and downloading of resident data

Use google cloud for periodic uploading of local patient data, and downloading of patient data (which can occur with changes made to the server database):

- Implement a service which uploads any changes since the last upload, based on a configurable time period - from shared preferences
- Handle download changes in patient data: e.g. medications, care plan, or special instructions.

Add as many tasks as you need to complete your app.

Submission Instructions

1. After you've completed all the sections, download this document as a PDF [File → Download as PDF]
2. Create a new GitHub repo for the capstone. Name it "**Capstone Project**"
3. Add this document to your repo. Make sure it's named "**Capstone_Stage1.pdf**"