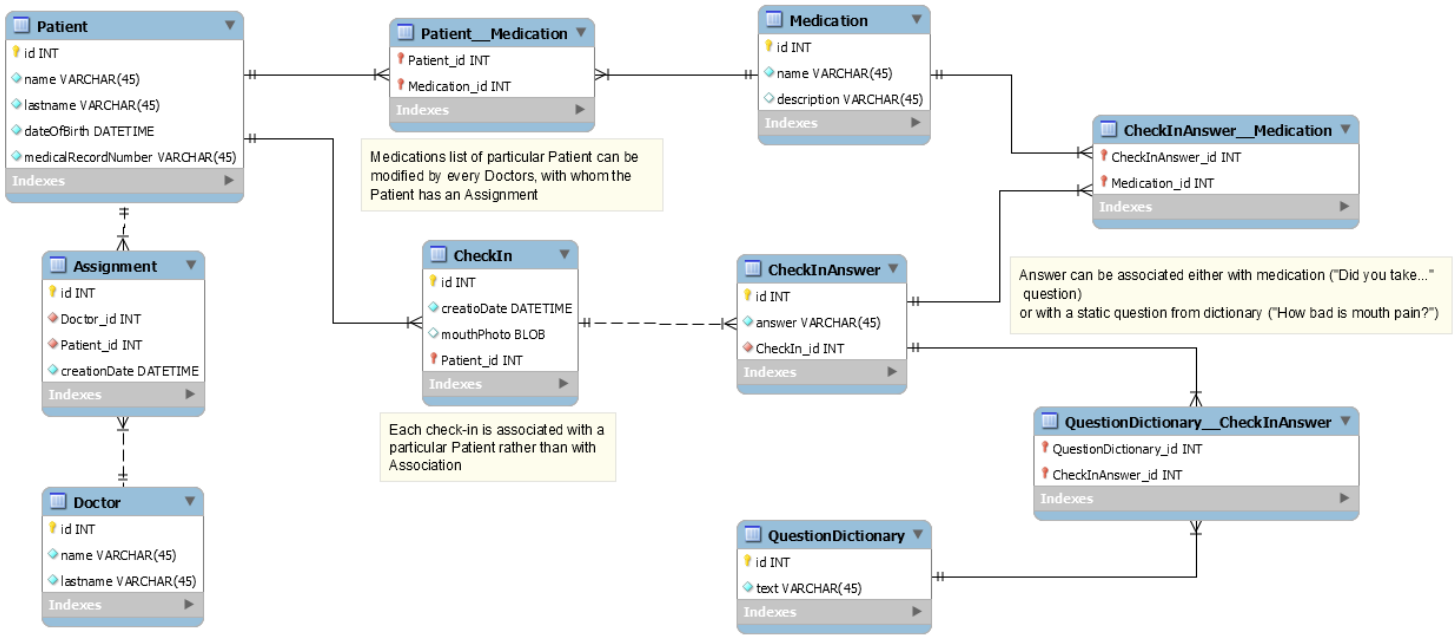
SYMPTOMS
Mid-Point Software Architecture Plan

Maciej Szarliński

1. Domain model (database)



2. Server RESTful API

PatientController

a) `POST /rest/checkIn` - create new CheckIn
b) `GET /rest/checkIn/template/patient/{id}` - get template for creating new CheckIn
c) `PUT /rest/patient/{id}/doctors/{id}` - add given Doctor to Patient's list
d) `DELETE/rest/patient/{id}/doctors/{id}` - remove given Doctor from Patient's list
e) `POST /rest/patient` - register new user with `PATIENT` role with given credentials

DoctorController

f) `GET /rest/checkIn/patient/{id}` - get all CheckIns for given Patient
g) `PUT /rest/patient/{id}/medications/{id}` - add given Medication to Patiet's list
h) `DELETE/rest/patient/{id}/medications/{id}` - remove given Medication from Patient's list
i) `POST /rest/doctor` - register new user with `DOCTOR` role with given credentials

3. Authentication

Server application is supposed to use Spring Security combined with OAuth2.0, more details can be found here: http://projects.spring.io/spring-security-oauth/docs/oauth2.html

Patient and doctor's client applications will store user encrypted credentials in local database to authenticate. User should register using rest-services 2e and 2i.

4. Data synchronization mechanisms

a) **patient's app** - user can manually adjust notification frequency, thus the requirement will be implemented using Android AlarmManager (timer task with fixed, adjustable frequency) and BroadcastReceiver executing REST call to a server.
b) **doctor's app** - looking for dangerous check-in answers will be implemented using Android AlarmManager.

Google Cloud Messaging API will not be used, but it is a good feature to be introduced in further release.

5. GUI - list of Activities

Patient's application

- **MainActivity** - contains 3 buttons: "Account" (go to AccountActivity), "My Doctors" (go to DoctorsListActivity) and "Settings" (go to SettingsActivity)
- **AccountActivity** - registering new user (with "Patient" role), managing credentials
- **SettingsActivity** - managing application settings like check-in reminder frequency
- **DoctorsListActivity** - managing list of Doctors assigned to the user, basically extends ListActivity and allows user to add and remove Doctors from the list. Patient can filter doctors list using filtering based search.

Check-in wizard

Check-in flow is launched after Patient clicks a check-in notification bar. It consists of the following activities/screens:

- **HowBadIsMouthPainActivity** - screen presenting first of *static* questions - "How bad is your mouth pain?" It contains 4 fine-skinned buttons - "Well controlled", "Moderate", "Severe" and "Next" (proceed to next question)
- **DoesPainStopFromEatingActivity** - screen presenting first of static questions - "Does your pain stop you from eating or drinking?" Contains 4 buttons - "No", "Some", "I can't eat" and "Next"
- **MedicationQuestionActivity** - for each Medication assigned to Patient, application will display separate (let's call it *dynamic* or *medication-based*) question. Activity allows user to answer "Yes" or "No" and provide a date when drug was taken
- **CompleteCheckInActivity** - contains button "Complete" which triggers an `AsyncTask` doing request to 2a.

Doctor's appplication

- **MainActivity** - contains 2 buttons: "Account" (go to AccountActivity) and "My Patients" (go to PatientsListActivity)

- **AccountActivity** - registering new user (with "Doctor" role), managing credentials
- **PatientsListActivity** - extends ListActivity and allows user to filter patients list using filtering based search. Each list item has 2 buttons - "Medications" (go to PatientMedicationsActivity) and "Check-ins" (go to PatientCheckInsActivity)
- **PatientMedicationsActivity** - extends ListActivity and allows user to add/remove medication from the list (see: 2g, 2h). Medication to be added can be searched in medication database (static or via web-service)
- **PatientCheckInsActivity** - doctor can browse graphically presented patient's check-in history (using for example: https://bitbucket.org/danielnadeau/holographlibrary/wiki/Home). User will come here from PatientsListActivity or from notification about patient's deteriorating health

6. Meeting the requirements

- How will Check-In data be transferred from a Patient's device to his/her Doctor?
  ✓ see: 2a, 2f
- How will you store the data on the server-side so that patients are associated with doctors? You may use a hard-coded set of patient and doctor user accounts provided by an in-memory UserDetailsService. You may also hard-code the relationship between patients and doctors.
  ✓ see: 1
- What will the user interface look like for a Patient so that it is quick and simple to use?
  ✓ see: 5a
- How will Reminders be delivered to a Patient in a way that will help the Patient to use the app more frequently and consistently?
  ✓ see: 4 (Reminders frequency will be adjustable by Patient using Settings screen)
- How will a Patient's data be securely transferred to the server?
  ✓ HTTPS, authentication
- How will a Doctor be able to update medication lists for a specific Patient, and how will these updates be sent to that Patient's device?
  ✓ see: 1, 2b, 2g, 2h (Patient will access medication-based questions within nearest check-in)
- What will the user interface look like for a Doctor?
  ✓ see: 5b
- How will a Doctor be alerted if a Patient has alarming symptoms?
  ✓ see: 4
- How, when, and how often will the user enter their user account information? For example, will the user enter this information each time they run the app? Will they specify the information as part of a preference screen?
  ✓ see: 3
- What user preferences can the user set? How will the app be informed of changes to these user preferences?
  ✓ see: 5a
- How will the app handle concurrency issues, such as how will periodic updates occur - via server push or app pull? How will search queries and results be efficiently processed? Will the data be pulled from the server in multiple requests or all at one time?
  ✓ see: 2
- How will the app use at least one advanced capability or API listed above?
  - For example, will you create an animation to explain the app?

    - ✓ Yes (if there will be enough time to create a screencast).
  - Will you allow patients to take pictures of mouth sores when their pain is high?
    - ✓ Yes.
  - Will you use push notifications to prompt users for pain information?
    - ✓ Yes, see: 4.
  - Will you allow users to employ simple gestures to snooze pain prompts for a set amount of time?
    - ✓ No.
- Does your app really require two or more fundamental Android components? If so, which ones? For example, this app might benefit from using a ContentProvider or from using a background Service that synchronizes local and remote data, only when the device is connected to a WiFi network.
  - ✓ See: 4 (Activity and BroadcastReceiver)