My Mental Health App

Group 9

Alexander, Yonatan SN: 251097927 Kien Tang Tran, Michael SN: 250735158 Aziz, Mohammad Yasin SN: 251282726



Computer Science 4470A
The Department of Computer Science
The University of Western Ontario



December 1, 2022

Please do NOT post our group's video to the course's YouTube Channel.

Contents

1	Problem Definition	2
2	Project Objective	2
3	Stakeholders List3.1 Internal Stakeholders3.2 External Stakeholders	2 2 3
4	Success/Acceptance Criteria for each Stakeholder	3
5	Use case diagram(s)	3
6	Selected Use case Descriptions only two descriptions	3
7	Sequence diagram(s) for the selected use case for descriptions	3
8	System Architecture	3
9	Detailed Class diagram(s)	4
10	State-machine diagram for the whole system, if possible	8
11	Entity Relationship Diagram (Data modelling)	8
12	GitHub Link	8
	Conclusion	9
	References	9
14		9
	1	10
		11
		11
		11
		11
		12
		13
		13
		14
		14
	.12 28-Nov-'22	15
Li	st of Figures	
	1 Class Diagram: Relationships	4
	2 Class Diagram: Medication	5
	3 Class Diagram: Mood	6
	4 Class Diagram: Symptoms	7
	Total Diagram. Dimpromo	•

1 Problem Definition

Therapists and patients encounter many small yet persistent challenges during the therapeutic process concerning the patients' ability to self-manage between sessions.

While this project's scope is not large enough to address them all, the following stood out to us:

- Mood tracking between sessions.
 - Usually, at the start of a session therapists ask for an update about the time since the last session. However, self-reporting bias usually means that if the patient is having a bad day or is in a bad mood, their view on the time since the last session is skewed, sometimes heavily. Further, memory can be corrupted and events can easily be forgotten.
- Medication management
 - Inventory (to not run out of important medications, mood stabilizers are a good example in the context of mental health especially)
 - Persistence (taking medications as required and without missing a day)
- Symptoms
 - Medical (unrelated to the patients diagnosed)
- Other self care aspects, such as sleeping, eating, etc.

2 Project Objective

- O1 Patients can add new entries.
- O2 Patients can view existing entries.
- O3 Patients can delete existing entries.
- O4 User will be able to view the entries from the past week, at a glance, in a dedicated view.
- O5 Learning the process of project development through application of knowledge acquired during the Computer Science 4471A Course.
- O6 The creation of an intuitive and user friendly self-management mobile app (Android).

3 Stakeholders List

• Patients - Internal stakeholder

3.1 Internal Stakeholders

Therapists - External stakeholder

external - anyone who has an influence including people who write certain libraries and so on (include instructor and TA as external stakeholders)

internal - programmers, maintainable, customers, users, etc [double check this point's list]

1. Patients (main users and the target audience).

3.2 External Stakeholders

- 1. Therapists
- 2. Legislation and public policy (for example security of medical information)
- 3. Governing bodies (such as the CRPO The College of Registered Psychotherapists of Ontario)
- 4. Instructor
- 5. TA
- 6. Libraries:
 - (a) Android.
 - (b) Google.
 - (c) MPAndroid (to create the graphs).
 - (d) JetBrains (makers of Android Studio).
 - (e) Java Libraries

4 Success/Acceptance Criteria for each Stakeholder

- As a user, I want to be able to store my mood on a particular day.
- As a user, I want to be able to rate my mood on a particular day on a scale of 0 to 10.
- As a user, I want to be able to store my medication details such as the medication, dosage, date, and doctor.
- As a user, I want to be able to have an organized collection of mood and medication reports for my therapy visit.
- As a user, I want to be able to create a visual representation of my recent self reported mood values.
- 5 Use case diagram(s)
- 6 Selected Use case Descriptions only two descriptions
- 7 Sequence diagram(s) for the selected use case for descriptions
- 8 System Architecture

Unfortunately, writing the code in Java meant that our chosen architecture (Model-View-ViewModel) was not feasible, resulting in needing to change the System Architecture to Model View Controller very late in development.

System Architecture: Model View Controller.

9 Detailed Class diagram(s)

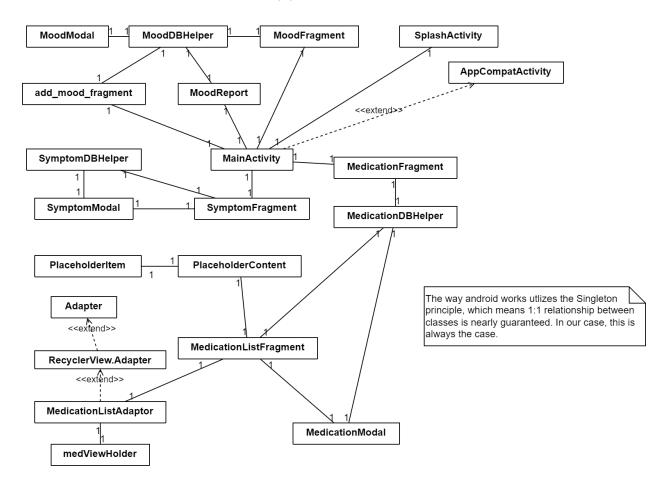


Figure 1: Class Diagram: Relationships.

Contains all classes in the system, and the relationships between them. The other class diagrams contain the respective methods.

MedicationFragment

- ARG_PARAM1:String {readOnly}
- ARG PARAM2: String {readOnly}
- medicationNameEdt: EditText
- brandNameEdt: EditText
- dosageEdt: EditText
- dosageUnitEdt: EditText
- frequencyEdt: EditText
- oldSystemName: EditText
- addMedicationBtn: Button
- readMedicationBtn: Button
- deleteMedicationBtn: Button
- updateMedicationBtn: Button
- + MedicationdbHelper: MedicationDBHelper
- + adapter: Adapter
- + MedicationFragment():
- + newInstance: MedicationFragment (String, String)
- + onCreate(Bundle): void
- + onCreateView(LayoutInflater,
- ViewGroup, Bundle): View
- + onClick(View): void

MedicationDBHelper

- DB Name: String {readOnly}
- DB_version: int {readOnly}
- Table_Name: String {readOnly}
- ID_COL: String {readOnly}
- medname_COL: String {readOnly}
- brandName_COL: String {readOnly}
- dosageQuantity_COL: String {readOnly}
- dosageUnit_COL: String {readOnly}
- frequency COL: String {readOnly}
- + onCreate(SQLiteDatabase): void
- + addNewMedication (String, String, String, String, String): void
- + deleteRecord(String): void
- + updateMedication(String, String, String, String, String, String): void
- + readMedications(): ArrayList<
- MedicationModal>
- + onUpgrade(SQLiteDatabase): void

MedicationModal

- id: long
- brandName: String
- commonName: String
- dosage: String
- frequency: String
- dosageUnit: String
- + MedicationModal(String, String, String, String, String)
- + getDosage(): String
- + setDosage(String): void
- + getFrequency(String): String
- + setFrequency(): void
- + getId(): long
- + setId(long): void
- + getBrandName():String
- + setBrandName(String):void
- + getDosageUnit():String
- + setDosageUnit(String): void
- + getCommonName(): String
- + setCommonName(String)

MedicationListFragment

- -ARG_COLUMN_COUNT: String {readOnly}
- mColumnCount: int = 1 {readOnly}
- + medicationsrc: RecyclerView
- + adapter:

MedicationListAdaptor

- + MedicationListFragment():
- + newInstance (int):
- MedicationListFragment
- + onCreate(Bundle): void
- + onCreateView(LayoutInflater,
- ViewGroup, Bundle): View
- + onClick(View): void

MedicationListAdaptor

- + list: ArrayList
- <MedicationModal>
- + context:Context
- + MedicationListAdaptor(Context, ArrauList
- <MedicationModal>):
- + onCreateViewHolder (
- @NonNull ViewGroup, int): medViewHolder
- + onBindViewHolder(
- @NonNull medViewHolder, int): void
- + getItemCount: int
- + update(ArrayList<
- MedicationModal>): void
- + onAttachedToRecyclerView:

medViewHolder

- + medCommonName: TextView
- + medBrandName: TextView
- + frequency: TextView
- + dosage: TextView
- + dosageUnit: TextView
- + medViewHolder(View):

Figure 2: Class Diagram: Medication

Contains all classes relating directly Medications. Relationships included only as part of *Class Diagram: Medication*.

add_mood_fragment - ARG Param1: String {readOnly} ARG Param2: String {readOnly} textProgress: TextView descriptionProgress: TextView myCalendar: Calendar {readOnly} mParam1: String mParam2: String moodvalue: int color: int submit button: Button update button: Button + add mood fragment() + newInstance:(String, String): add mood fragment + onCreate(Bundle): void + onCreateView(LayoutInflater, ViewGroup container, Bundle): View + onProgressChanged (SeekBar, int, boolean): void + onStopTrackingTouch(SeekBar): void

+ onStartTrackingTouch(SeekBar): void

+ onClick(View): void

+ setColor(int): void

+ getColor: int

+ setMoodValue(): void

+ getMoodValue(int): int

+ onDateSet(DatePicker, int, int, int): void

+ updateView(int) MoodReport MoodModal + barChart: BarChart - id: int + barData: BarData - moodDescription: String + barDataSet BarDataSet - moodRating: int - date: String + getMoodDescription(): + MoodReport(): + newInstance (String, String String): MoodReport + setMoodDescription(String): + onCreate (Bundle): void void + onCreateView(LayoutInflater + getId(): int ViewGroup, Bundle): View + setId(int): void + getMoodRating(): int + setMoodRating(int) + getDate(): String + setDate(String)

MoodDBHelper - DB Name: String {readOnly} - DB version: int {readOnly} - Table Name: String {readOnly} id col: String {readOnly} moodRating col: String {readOnly} - description col: String{readOnly} - date col: String {readOnly} + onCreate(SQLiteDatabase): void + addNewMood(int, String, String): + readMoods(): ArrayList<MoodModal> + updateMoods(int, String, String, String): void + deleteMood(String): void + ReadSortByDate(): ArrayList <MoodModal: + onUpgrade(SQLiteDatabase): void

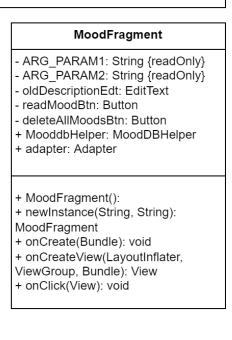


Figure 3: Class Diagram: Mood

Contains **all** classes relating directly Mood. Relationships included only as part of *Class Diagram: Medication*.

SymptomDBHelper

- + DB_Name: String {readOnly}
- + DB_version: int {readOnly}
- + Table_Name: String {readOnly}
- + id_col: String {readOnly}
- + datetime_col: String {readOnly}
- + symptom_name_col: String {readOnly}
- + symptom_description_col: String {readOnly}
- + SymptomDBHelper(Context)
- + onCreate(SQLiteDatabase): void
- + addNewSymptom(String, String, String): void
- + deleteSymptom(String symptomName): void
- +updateSymptom(String, String, String, String): void
- + readSymptoms(): ArrayList
- <SymptomModal>
- + onUpgrade(SQLiteDatabase): void

SymptomFragment

- symptomDateEdt: EditText
- symptomNameEdt:

EditText

symptomDescriptionEdt:

EditText

- addSymptomBtn: Button
- readSymptomBtn: Button
- deleteSymptomBtn: Button
- updateSymptomBtn: Button
- + SymptomdbHelper: SymptomDBHelper
- + SymptomFragment():
- + newInstance(String,

String): SymptomFragment

- + onCreate(Bundle): void
- + onCreateView(LayoutInflater, ViewGroup, Bundle)
- + onClick(View)

SymptomModal |

- + id: int
- + name: String
- + description: String
- + date: String
- + SymptomModal(String, String, String):
- + getId(): int
- + setId(int): void
- + getName():String
- + setName(String): void
- + getDescription(): String
- + setDescription(String)
- + getDate(): String
- + setDate(String): void

Figure 4: Class Diagram: Symptoms

Contains **all** classes relating directly Symptoms. Relationships included only as part of *Class Diagram: Medication*.

MainActivity PlaceholderContent + drawerLayout: DrawerLayout + ITEMS: List<PlaceholderItem> + actionBarDrawerToggle: {readOnly} ActionBarDrawerToggle + ITEM MAP: Map<String, + nv: NavigationDraw PlaceholderItem> {readOnly} COUNT: int {readOnly} # onCreate (Bundle savedInstancedState): void + onNavigationItemSelected(@NonNull MenuItem): boolean + onOptionsItemSelected(@NonNull MenuItem item): boolean - addItem(PlaceholderItem) + onCreateOptionsMenu(Menu menu): boolean createPlaceholder(int): - setMyFragment(Fragment fragment): void PlaceholderItem - makeDetails(int) : String Adapter **PlaceholderItem**

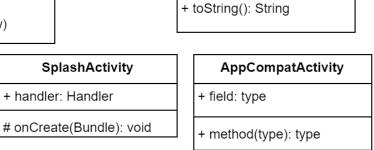
- context: Context
- Adapter(Context context) - onCreateViewHolder(@NonNull /iewGroup parent; int viewGroup) - onBindViewHolder(@NonNull viewHolder nolder, int position) - onMenuItemClick(MenuItem item): boolean - getItemCount(): int - viewHolder(@NonNull View itemView)

RecyclerView.Adapter<

medViewHolder>

+ field: type

+ method(type): type



+ id: String {readOnly}+ content: String {readOnly}+ details: String {readOnly}

+ PlaceholderItem(String,

String, String):

Figure 5: Class Diagram: Other

Contains all classes **not** already included in the other diagrams above. Relationships included only as part of Class Diagram: Medication.

- 10 State-machine diagram for the whole system, if possible
- 11 Entity Relationship Diagram (Data modelling)
- 12 GitHub Link

You MUST ask Yonatan Alexander for access, as this repository is set to private.

https://github.com/Jon-AL/4471—Therapy-Support-App

13 Conclusion

Our project, in many ways, did not go according to plan. We made a number of mistakes, but we believe the biggest was not making the project in Kotlin.

14 References

- 1. MyTherapy (app): Your personal pill reminder and medication tracker app[?]
- 2. **Bipolar UK's Mood Tracker (app):** Our new Mood Tracker app can make it much easier to record your daily mood, medications, emotions and how much sleep you've had[?]

main

Project WBS

Task Assignment Matrix

Sample of commits on the selected version control system

Wants

The following were not added to the system.

- 1. docker
- 2. gant chart
- 3. Reminder system
- 4. Privacy
- 5. Minutes
- 6. Encryption for data (including for local storage)
 - Was implemented as part of Android since Android 11.
- 7. Login system
- 8. Therapist interface

Minutes

.1 26-Sep-'22

Present:

- Yonatan
- Michael

Not Present:

• Yasin

- Development proposal and development details.
- The project will be a therapy application that helps users gather, store and present their information to be used by patients.
- Suggested so far:
 - Incremental development.
 - Smartphone app.
 - Web app.
- Suggested Programming languages:
 - Dart.
 - Python.
 - Javascript.
 - C++.
 - Java.
- Stakeholders:
 - Clients.
 - Therapists and other mental health professionals.
- Features:
 - Mood self-reported values.
 - Medication tracking.

.2 30-Sep-'22

Present:

- Yonatan
- Michael

Not Present:

• Yasin

- Project proposal was accepted.
- Instructor mentioned a number of points during lecture in the context of our project:
 - Patient data requirements with respect to privacy.
 - Identifying the stakeholders with who should be able to access data.
 - We need to identify the use cases carefully of what we are able to do.
 - We have to be able to clearly define the project objective and description.

- An aggressive timeline has been suggested to keep with coursework. This timeline is subject to change due to other courses requiring our attention.
- Github was setup by Yonatan.
- Report is in progress.

.3 04-Oct-'22 and 05-Oct-'22

Discussed:

A timeline has been suggested to be finishing up to the user acceptance criteria. QT has been suggested as a platform to build an Android app.

The report has been started in LaTex by Yonatan. Proposed statements will be made by the group once we reviewed the work.

.4 14-Oct-'22

Present:

- Yonatan
- Michael
- Yasin

Discussed:

Debate about the code is that we have no firm agreements on project libraries and frameworks. Suggestions include: salesforce as the backend. Flutter or ReactNative has also been suggested. Qt.

.5 21-Oct-'22 to 28-Oct-'22

A lot of project will be developed on the reading week. Professor wants us to use Docker, and create a Gantt chart. He explicitly states that he wants us to assign backup roles to our task assignment matrix.

.6 01-Nov-'22

Present:

- Yonatan
- Michael

Not Present:

• Yasin

Discussed:

Planned architecture:

- repository style for the databases.

- Event-driven for reminders.
- Component-based.
- Today's objective:
 - Architecture and design patterns were discussed.
 - Get the backend done behind the scenes.
 - Another meeting: 9:30 am to 11:30 am tomorrow (November 2nd, 2022), want to start writing code.

.7 02-Nov-'22

Present:

- Yonatan
- Michael
- Yasin

- We are switching the development process from incremental to a hybrid agile-iterative method.
- Scope and requirements changed significantly throughout the process.
- We are also incorporating agile methods to respond to changes over time.
- Kanban board has been created inside GitHub repository.
- User stories will be created for UI to make the interface clearer to write components and understand the problems.
- Drafts have been started but significant changes need to be created.
- An Android application will be created.
- The data will be stored locally with some encryption to get us started.
- Yonatan wants to push simple skeleton code, will require approval from Michael and Yassine.
- We want to finish baseline features executed and finished by November 6th.
- The list of things we want executed:
 - A basic UI covered by Yassine
 - Basic backend covered by Yonatan
 - Database integration covered by Michael
- Significant research must be implemented for any implementation to occur.
- No meeting will be scheduled for tomorrow.
- A few mini-meetings between Friday to Sunday.

.8 02-Nov-'22

Present:

- Yonatan
- Michael

Not Present:

• Yasin

Discussed:

- Backend talk
- Significant modifications need to be executed for medication.
- We will have to split this into 3 main Tables:
 - 1. Medication info
 - 2. Medication inventory history
 - 3. Medication intake history
- Mood will be generated next.
- Symptoms will be generated last.
- Backend sample is expected to be done by Sunday late afternoon.
- Yassine will present a frontend demo on November 6th.

.9 14-Nov-'22

Present:

- Yonatan
- Michael
- \bullet Yasin

- Front end and back end are being implemented. We are integrating today.
- Our approach has changed. We have recognized that the Model-View-ViewModel (MVVM) pattern was the best representation.
- Things to do:
 - Integrate front-end and back-end.
 - Revise databases.
 - Check on scope and extra features.
 - Testing and presentation notes to build.

.10 24-Nov-'22

Present:

- Yonatan
- Michael

Not Present:

• Yasin

Discussed:

•

- Presentation is being worked on.
- Yonatan changed color scheme and logo.
- Things to do:
 - 1. He wants a better option for moods. A better option would be spinner or radio button. A perfect solution would be adding a slider [Seek bar].
 - 2. We need to focus more on accessibility.
 - 3. Add statistics.

.11 27-Nov-'22

Present:

- Yonatan
- Michael

Not Present:

• Yasin

- Code must refactored to meet architectural standards.
- Code will be redone to the exact standard to the specified report
- Mood will be the focus.
- Medication and Symptoms are not the primary objective.
- Deadline to finish the code is tomorrow night.
- Deadline should be finished by Wednesday night.
- Deadline to finish the document components is Thursday.

.12 28-Nov-'22

Present:

- Yonatan
- Michael

Not Present:

• Yasin

- Multiple meetings today.
- We have to go with MVC. MVVM was not feasible given the circumstances.
- Code is being redone to make the sections better.
- We want to get the code ready for deployment soon.
- Things to do:
 - Clean up redundant code
 - add comments
 - fix anything that seems broken.
 - Deploy on Docker.
 - Another addition of report documents and diagrams.