CSIS 3175-002\_14970\_202110

Introduction to Mobile App Development

**Health Management**

Submitted by:

GROUP 8

Kunal Ajaykumar Jeshang 300328339

Luan Reisdoerfer Cardoso 300316053

Mahdi Abbasimoghadam 300316638

Ngoc Vy Pham 300284563 - Absent

Submitted to:

Mehwish Bashir

Douglas College

March 31st, 2021

Contents

[Number of Screens 3](#_Toc68671014)

[Main Features 3](#_Toc68671015)

[Admin 3](#_Toc68671016)

[Patient 3](#_Toc68671017)

[Doctor 3](#_Toc68671018)

[Cashier 4](#_Toc68671019)

[Java classes and activities 4](#_Toc68671020)

[Dao Package 4](#_Toc68671021)

[Model Package 4](#_Toc68671022)

[UI Package (classes in this package also have xml layouts to display screen contents) 4](#_Toc68671023)

[Database 5](#_Toc68671024)

[Admin (Admin ID, Admin Name, Admin Password) 5](#_Toc68671025)

[Patient (Patient ID, Patient Name, Patient Password, Patient Postal Code, Allergies, Diseases) 5](#_Toc68671026)

[Doctor (Doctor ID, Doctor Name, Doctor Password, Clinic Postal Code, License Number) 6](#_Toc68671027)

[Cashier (Cashier ID, Cashier Name, Cashier Password) 7](#_Toc68671028)

[Calories (Calories ID, *Patient ID*, Food List, Total Calories, Health Suggestion, Date Of Consumption) 7](#_Toc68671029)

[Appointment (Appointment ID, *Doctor ID*, AppointmentDate, *Patient ID*) 7](#_Toc68671030)

[Payment (Payment ID, *Cashier ID*, *Patient ID*, Due Payment) 8](#_Toc68671031)

[Tasks Description 9](#_Toc68671032)

[Kunal 9](#_Toc68671033)

[Mahdi 12](#_Toc68671034)

[Luan 13](#_Toc68671035)

[Screens 16](#_Toc68671036)

[References 21](#_Toc68671037)

Post-Proposal Summary

Repository Link: <https://github.com/kjeshang/CSIS3175-Group-Project.git>

# Number of Screens

Current count = 20 screens

# Main Features

## Admin

* View a list of all current users of the application
* View user details of current patients, doctors, and cashiers
* Update password, full name, postal code, allergies, and diseases (underlying health conditions) of an existing patient user
* Update password, full name, clinic postal code and license number of an existing doctor user
* Update password and full name of an existing cashier user
* Delete account of an existing patient, doctor, or cashier user from the application
* Register a new patient, doctor, or cashier user to the application

## Patient

* Sign up to the the application if account does not exist
* Log into application if account already exists
* View user details
* Update password, full name, postal code, allergies, and diseases (underlying health conditions)
* Pick a date & time for an in-person/virtual consultation from a list of available doctors
* Patient can provide details about the food they have eaten on a particular day, and in turn track their total calories
* If a patient requests online help regarding a sickness or underlying health condition, they can submit a complaint via the application; they have the option to pay for the online help in advance via MSP or out-of-pocket using PayPal
* By posting a complaint to a doctor, a patient will also have the ability to view prior chat history with the doctor

## Doctor

* Sign up to the the application if account does not exist
* Log into application if account already exists
* View user details
* Update password, full name, clinic postal code and license number
* Doctors can view a list of all patient inquiries/complaints, select a patient, and submit a solution for the patient’s complaints (i.e. medication, subscription, home remedy, etc.)
* By posting a solution to an ailing patient, a doctor will also have the ability to view prior chat history with the patient

## Cashier

* Sign up to the the application if account does not exist
* Log into application if account already exists
* View user details
* Update password and full name
* Cashier can view a list of patients that requested online help, and can clear outstanding payments of the patients

# Java classes and activities

## Dao Package

* + DatabaseHelper Java class

## Model Package

* + Admin sub-package
    - Admin Java class (1; extended from User class)
  + Cashier sub-package
    - Cashier Java class (1; extended from User class)
    - Payment Java class
  + Doctor sub-package
    - Doctor Java class (1; extended from User class)
    - Chat Java class
  + Patient sub-package
    - Appointment Java class
    - Calories Java class
    - Patient Java class (1; extended from User class)
  + User Java class

## UI Package (classes in this package also have xml layouts to display screen contents)

* + AdminUI sub-package
    - AdminAccount Java class
    - AdminAddUser Java class
    - AdminCashier Java class
    - AdminDoctor Java class
    - AdminPatient Java class
  + CashierUI sub-package
    - CashierAccount Java class
    - CashierInfo Java class
  + DoctorUI sub-package
    - DoctorAccount Java class
    - DoctorInfo Java class
    - DoctorOnlineHelp Java class
  + PatientUI sub-package
    - PatientAccount Java class
    - PatientAppointment Java class
    - PatientFindDoctor Java class
    - PatientFoodTracker Java class
    - PatientTracker Java class
    - PatientInfo Java class
    - PatientOnlineHelp Java class
  + LoginActivity Java class
  + MainActivity Java class
  + SignupActivity Java class

1. The indicated Java classes, Admin, Cashier, Doctor, & Patient, are extended from the User Java Class as the aforementioned are all users of the health management system application.
2. The DoctorAccount Java class implements the DoctorImageAdapter class in order to create a scrollable list of Patient inquiries when they require online help regarding their allergies, diseases/underlying health conditions.

# Database

## Admin (Admin ID, Admin Name, Admin Password)

* Primary key = Admin ID

|  |  |  |  |
| --- | --- | --- | --- |
| Column name | Data type | Constraint | Description |
| Admin ID | Text | Primary Key | Admin email or abbreviated name; presumed unique |
| Admin Name | Text | Not Null |  |
| Admin Password | Text | Not Null | Presumed unique |

## Patient (Patient ID, Patient Name, Patient Password, Patient Postal Code, Allergies, Diseases)

* Primary key = Patient ID

|  |  |  |  |
| --- | --- | --- | --- |
| Column name | Data type | Constraint | Description |
| Patient ID | Text | Primary Key | Patient email; presumed unique |
| Patient name | Text | Not Null |  |
| Patient password | Text | Not Null | Presumed unique |
| Patient postal code | Text | Not Null |  |
| Allergies | Text | Not Null | If patient does not have any allergies, they need to inform that upon signup |
| Diseases | Text | Not Null | If patient does not have any diseases, they need to inform that upon signup |

## Doctor (Doctor ID, Doctor Name, Doctor Password, Clinic Postal Code, License Number)

* Primary key = Doctor ID

|  |  |  |  |
| --- | --- | --- | --- |
| Column name | Data type | Constraint | Description |
| Doctor ID | Text | Primary Key | Doctor password; presumed unique |
| Doctor Name | Text | Not Null |  |
| Doctor Password | Text | Not Null |  |
| Clinic Postal Code | Text | Not Null | It is presumed that all doctors are associated with a clinic or hospital despite working remotely, although some doctors could be directly affiliated with Health Buddy as in-house doctors providing their services to patients |
| License Number | Text | Not Null | Presumed unique; doctor’s need a license to interact with patients |

## Cashier (Cashier ID, Cashier Name, Cashier Password)

* Primary key = Cashier ID

|  |  |  |  |
| --- | --- | --- | --- |
| Column name | Data type | Constraint | Description |
| Cashier ID | Text | Primary Key | Cashier email; presumed unique |
| Cashier Name | Text | Not Null |  |
| Cashier Password | Text | Not Null | Presumed unique |

## Calories (Calories ID, *Patient ID*, Food List, Total Calories, Health Suggestion, Date Of Consumption)

* Primary Key = Calories ID
* Foreign Key = Patient ID

|  |  |  |  |
| --- | --- | --- | --- |
| Column name | Data type | Constraint | Description |
| Calories ID | Integer | Primary Key, Auto Increment |  |
| Patient ID | Text | Foreign Key | Patient email; presumed unique |
| Food List | Text |  |  |
| Total Calories | Integer |  |  |
| Health Suggestion | Text |  |  |
| Date of Consumption | Date |  |  |

## Appointment (Appointment ID, *Doctor ID*, AppointmentDate, *Patient ID*)

* Primary key = AppointmentID
* Foreign keys = Doctor ID, Patient ID

|  |  |  |  |
| --- | --- | --- | --- |
| Column name | Data type | Constraint | Description |
| Appointment ID | Integer | Primary Key, Auto Increment |  |
| Doctor ID | Text | Foreign Key | Doctor email; presumed unique |
| Appointment Date | DateTime | Null | Date & time of patient’s appointment with doctor |
| Patient ID | Text | Foreign Key | Patient email; presumed unique |

## Payment (Payment ID, *Cashier ID*, *Patient ID*, Due Payment)

* Primary key = Payment ID
* Foreign key = Cashier ID, Patient ID

|  |  |  |  |
| --- | --- | --- | --- |
| Column name | Data type | Constraint | Description |
| Payment ID | Integer | Primary Key, Auto Increment |  |
| Patient ID | Text | Foreign Key | Patient email; presumed unique |
| Due Payment | Integer | Null | Payment would be null or zero if the patient has MSP coverage |

***Database Notes***

* Admin login details:
  + Username = mehwish
  + Password = mehwish123
* Not Null constraint for Patient, Doctor, Cashier, & Admin relation’s columns enforced through the ‘checkUser()’ method which is extended, from the User class, by the entity classes of the aforementioned relations
  + Based on above, ‘Not Null’ is not explicitly included in the SQLite code
  + E.g. if the cashier user does not provide their email (i.e. user ID), full name, and password, the application will not allow them to create an account
  + It is presumed that if a patient users does not have any allergies or underlying health conditions/sickness (i.e. diseases), they would still have to inform that while signing up for an account

# Tasks Description

## Kunal

General timeline

|  |  |  |
| --- | --- | --- |
| Date | Link | Description |
| March 10, 2021 | N/A | Started to work on my coding part of the project   * Searching online to supplement onto the knowledge learnt from lecture in regards to SQLite & Android Studio * Completed SQLite & Android Studio freeCodeCamp YouTube tutorial * Started coding project |
| March 14, 2021 | <https://github.com/kjeshang/CSIS3175-Group-Project/commit/2bab39802d0e79b83bab104c61269136cc74bb39> | Create GitHub repository |
|  | <https://github.com/kjeshang/CSIS3175-Group-Project/commit/6ab607d4a7a2d3f60056666ac97d14b5ffc80290> | First attempt at project   * Screens created: MainActivity, SignUpActivity, LoginActivity, PatientAccount, PatientInfo, DoctorAccount, DoctorInfo, CashierAccount, CashierInfo * Classes created: Patient, Doctor, Cashier, DatabaseHelper, PatientDAO, DoctorDAO, CashierDAO |
| March 17, 2021 | <https://github.com/kjeshang/CSIS3175-Group-Project/commit/665703260163258ec5aabc862d4bc9ca08cb06f2> | Reworked project   * Classes removed: PatientDAO, DoctorDAO, CasierDAO * Classes added: User, Admin * Screens added: AdminAccount, AdminPatient * Worked on other existing classes & screens |
| March 21, 2021 | <https://github.com/kjeshang/CSIS3175-Group-Project/commit/787e14528946a6119e981bd852a7acb44cf384d1> | In progress - updated repository at request of team member   * Screens added: AdminDoctor, AdminCashier * Removed Disease class and reworked Disease class parameters into Patient class * Worked on other existing classes & screens |
|  | <https://github.com/kjeshang/CSIS3175-Group-Project/commit/494363774a0e9523049353a35e03b56e0b7158c7> | Completed my allocated screens   * Screens added: AdminAddUser * Worked on other existing classes & screens |
|  | <https://github.com/kjeshang/CSIS3175-Group-Project/commit/a2a598b4da5d0bfc5b0c7c5cfcb302e4b7234e80> | Re-uploaded my part to repository when trying merge pull request error with another team member’s part |
| March 22, 2021 | N/A | Worked on report related to my coding part, as well as other general parts |
| March 24, 2021 | <https://github.com/kjeshang/CSIS3175-Group-Project/commit/e64e9c14ac79a4888cc3c3ea567a90c8cfed6fb6> | Due to certain circumstances, a team member left the group, so I worked on their original allocated part of the project   * Screens added: PatientFindDoctor, PatientAppointment * Classes added: Appointment class * Configured PatientAccount screen so that by clicking the ‘Find Doctor’ button, the user is moved to the PatientFindDoctor screen |
| March 24, 2021 - March 30, 2021 | N/A | * Consulted Luan and Mahdi regarding their progress, and monitored their GitHub branches for commits * Based on the above mentioned, I worked on parts of the report related to Luan and Mahdi’s coding parts of the project |
| April 6th, 2021 | <https://github.com/kjeshang/CSIS3175-Group-Project/commit/16943cad8e67f05f2de5ee7c35a5744be9da4723> | * Made some design changes to the project to reflect the visual design of the screens from the proposal * XML files worked on = AndroidManifest, colors, themes |
|  | <https://github.com/kjeshang/CSIS3175-Group-Project/commit/fd0c4e501eea719336c23506e66db6b83799359d> | * Added recent contributions again to an updated project version |

Limitations & issues experienced when coding application

* Due to SQLite’s limitation within Android Studio, a patient user can book an appointment with a doctor at a specific date & time, and then book another one with the same doctor at the same day that starts one minute after the previous one. The aforementioned is illogical and not realistic, but I was unable to implement a method that intuitively checks the desired appointment time with an existing one in terms of duration.
* From the initial screen designs, a patient user was supposed to search amongst a list of doctors based on the patient user’s distance to a doctor’s clinic’s postal code. I am aware of the edit text change listener feature on Android Studio but I was not able to configure that to work in conjunction with a SQLite query that searches the Doctor table by clinic postal code. Therefore, when a patient user chooses to find a doctor for an appointment, a simple listview shows all doctors that are registered with the application.
* Due to the patient, doctor, cashier, and admin tables utilizing user IDs (i.e. emails) as primary keys, a respective user is not able to change their email. The only way to change an email is for the admin to delete & recreate the account of the respective user along with their associated details. In a practical application, this would be illogical, but as I have yet to master the fundamentals of mobile application development, I only realized later that in the case of SQLite it is best to utilize a generic-auto incremented integer primary key for tables. As significant development time had already elapsed, I was unable to significantly change the database as it would take a lot of time, and may affect the simultaneous development of my group members allocated parts.
* After an appointment is made, a message is displayed that appointment details have been emailed to the patient user. There is no feature to check the appointment of a patient user due to limited time, but it would be a feature I would have added.
* I would have liked to include a feature where both patient and doctor could view their respective appointment dates and times in a separate screen, but due to time constraint I was not able to implement it.

## Mahdi

General timeline

|  |  |  |
| --- | --- | --- |
| Date | Link | Description |
| March 15th, 2021 | * https://www.youtube.com/watch?v=EJrmgJT2NnI - Extra reference | * Started Learning and doing research on extra features that are going to be needed for the project |
| March 20th, 2021 |  | * Sketching related database tables for my part |
| March 27, 2021 | <https://github.com/kjeshang/CSIS3175-Group-Project/commit/48d13c3bfed0c1f7dc353b41a1cbf971a9a69496> | * Screens created: PatientFoodTracker |
| March 28 2021 | <https://github.com/kjeshang/CSIS3175-Group-Project/commit/1a9962e19c3d025c455173f98b75824fd3f96376> | * Screens worked on: PatientAccount, PatientFoodTracker |
|  | <https://github.com/kjeshang/CSIS3175-Group-Project/commit/9dcc1325426e8ddfa2ec111a8ee4d4e9d1ac8487> | * Screens worked on: PatientFoodTracker |
| March 30, 2021 | <https://github.com/kjeshang/CSIS3175-Group-Project/commit/e99156879e5f7ad28d2536b78e92b0f67646d884> | * Screens created: Patient\_OnlineHelp, PatientTracker * Classes created: Calories * Classes worked on: Databasehelper |

|  |  |  |
| --- | --- | --- |
| April 04, 2021 | https://github.com/kjeshang/CSIS3175-Group-Project/commit/251f511e4344bfde2e8365ff1ab3f25f509f3699 | * Completing   PatientFood Tracker Activity   * Adding extra features like another date picker |
| April 05, 2021 | Same Link as the upper commit | * Worked on the report (addedd layouts) * Created custom drawable boxes * Changed the whole theme of the app along with Kunal’s work |
| April 05, 2021 | https://github.com/kjeshang/CSIS3175-Group-Project/commit/0c17f0631326e390c16f10d41602b9f91de89ace | * Merged all the changes in release branch to the main branch manually |
|  |  |  |

Limitations & issues experienced when coding applications.

* Struggled with retrieving data from the calories table by patient user specified date.
* Since I wanted to completely learn about GitHub, and it took me awhile, the dates that you see are not the exact days that I worked on these parts. For most of them, I commit sometime after working with the actual files.
* Implementing a new date picker was another challenge on my way, that pushed me for self-learning.
* Overall, I am totally happy with all the database limitations and barriers that this project had because it pushed us to move forward.
* Cloning and manually updating the main branch also appeared as another problem which I managed to solve by watching and learning new commands to merge from cmd.

## Luan

General timeline

|  |  |  |
| --- | --- | --- |
| Date | Link | Description |
| March 18, 2021 | <https://github.com/kjeshang/CSIS3175-Group-Project/commit/58b248dd47f872cede6d130df68fede6ee3d35d0> | Uploaded project for testing out GitHub features |
|  | <https://github.com/kjeshang/CSIS3175-Group-Project/commit/c08bf9fb311d8b60350504fb1fedc8e59ae22b3c> | Practiced merging pull request |
| March 21, 2021 | <https://github.com/kjeshang/CSIS3175-Group-Project/commit/1768803b64d12fbc3f7b9a7661956024e6796a8f> | Practiced pushing file using direct upload |
|  | <https://github.com/kjeshang/CSIS3175-Group-Project/commit/ae9dfdaae4f26270ee90eca4f1591d8635e14d3f> | Added first contribution to GitHub   * Screens worked on: DoctorAccount * Classes worked on: DatabaseHelper |
| March 29, 2021 | <https://github.com/kjeshang/CSIS3175-Group-Project/commit/192032d2956e683698c3aee215d1d0a45012f852> | * Screens worked on: DoctorAccount |
|  | <https://github.com/kjeshang/CSIS3175-Group-Project/commit/994b00bd1966367faec7ce6aff2f2cc63296cc8c> | * Screens worked on: DoctorAccount * Class created: DoctorImageAdapter * Classes worked on: DatabaseHelper |
| March 30, 2021 | <https://github.com/kjeshang/CSIS3175-Group-Project/commit/299474b3554c81a68a0dccd239245a8bc30a851f> | * Screens created: OnlineHelp (for Doctor) * Screens worked on: DoctorAccount, PatientAppointment * Classes worked on: DoctorImageAdapter |
|  | <https://github.com/kjeshang/CSIS3175-Group-Project/commit/020cfd25ac20d5bec25b95cfbce3868927e3d701> | * Classes worked on: Databasehelper |
| April 4 | <https://github.com/kjeshang/CSIS3175-Group-Project/commit/a10cb55fd8c7ec5581776704928752256bddeee3> | * Doctor’s Online Help Screen |
| April 5 | <https://github.com/kjeshang/CSIS3175-Group-Project/commit/ab1afebea6b59abac68c2eddeb05930c0fe9de23> | * Patient’s Online Help Screen |
| April 6 | <https://github.com/kjeshang/CSIS3175-Group-Project/commit/3a4f51eec262f531030ae257eed62287b8b4a644> | * Payments processing function and Cashier’s balance field and list of patients with due payment * Application Design |

Limitations & issues experienced when coding application

* Struggling to integrate the database layer with the patient inquiries that are supposed to appear in the DoctorAccount screen
* Struggling to deal with Online Help screen (for Doctor) in terms of managing & formatting chat history in order to be viewed by the user of the application

# Screens

|  |  |
| --- | --- |
| MainActivity | SignupActivity |
| LoginActivity | PatientAccount |
| PatientInfo | DoctorAccount |
| DoctorOnlineHelp | PatientOnlineHelp |
| PatientFoodTracker | PatientTracker |
| DoctorInfo | CashierAccount |
| CashierInfo | AdminAccount |
| AdminPatient | AdminDoctor |
| AdminCashier | AdminAddUser |
| PatientFindDoctor | PatientAppointment |

# References

***MVC code structure***

CSIS 3275-001 Software Engineering (Winter 2021)

***Object oriented approach to querying records & listing records in a listview using SQLite database in android studio***

Shad Sluiter, &amp; freeCodeCamp.org. (2020, October 13). SQLite database for Android - full course. <https://www.youtube.com/watch?v=312RhjfetP8&amp;list=LL&amp;index=2>.

***Singleton pattern for setting up database interaction in android studio with SQLite database***

Local Databases with SQLiteOpenHelperEdit PagePage History. Local Databases with SQLiteOpenHelper | CodePath Android Cliffnotes. (0AD). <https://guides.codepath.com/android/local-databases-with-sqliteopenhelper>.

***Formatting date and time string values in a way that can be inserted & read by SQLite in conjunction with Android Studio***

Tamada, R. (2013, October 19). Android insert datetime value in SQLite database. <https://tips.androidhive.info/2013/10/android-insert-datetime-value-in-sqlite-database/>.

mcr619619mcr619619 42711 gold badge44 silver badges1313 bronze badges, LuisLuis 11.7k33 gold badges2323 silver badges3434 bronze badges, & RajeshVijayakumarRajeshVijayakumar 9. (2013, August 1). *Inserting datetime in sqlite database android*. Stack Overflow. <https://stackoverflow.com/questions/13105609/inserting-datetime-in-sqlite-database-android>.

***Implement calendar view widget and get user selected date for further use***

CodingWithMitch. (2017, April 7). *Android Beginner Tutorial #26 -CalendarView [Getting the Date and Displaying in a TextView]*. YouTube. <https://www.youtube.com/watch?v=hHjFIG0TtA0>.

IljaIlja 34.8k6464 gold badges216216 silver badges394394 bronze badges, & Luca SepeLuca Sepe 2. (2013, January 1). *How to get date from calendarView() onCreate, with a specific format e.g. DD/MM/YYYY?* Stack Overflow. <https://stackoverflow.com/questions/22461258/how-to-get-date-from-calendarview-oncreate-with-a-specific-format-e-g-dd-mm>.

android kotlin - CalendarView get selected date. (n.d.). <https://android--code.blogspot.com/2020/06/android-kotlin-calendarview-get.html>.

***Implement time picker and get selected time for further use***

Coding Demos. (2018, March 1). *Android Timepicker – Use EditText to Show TimePickerDialog (Explained)*. YouTube. <https://www.youtube.com/watch?v=4wyA7HDhIOA>.

says:, P., & says:, D. (n.d.). *TimePicker Tutorial With Example In Android Studio*. Abhi Android. <https://abhiandroid.com/ui/timepicker>.

***Passing lists from activity to another***

(2016, November 1). How to save List to SharedPreferences? Stack Overflow. <https://stackoverflow.com/questions/28107647/how-to-save-listobject-to-sharedpreferences/28107838>.

***Passing objects from one activity to another***

Gandla, N. (2020, July 31). How to pass objects between Android Activities? Medium. <https://gaandlaneeraja.medium.com/how-to-pass-objects-between-android-activities-86f2cfb61bd4>.

***Creating & utilizing an adapter class to correctly utilize a recycler view***

Friedel, R. (2016, November 18). Android Fundamentals: Working with the RecyclerView, Adapter, and ViewHolder Pattern. WillowTree. <https://willowtreeapps.com/ideas/android-fundamentals-working-with-the-recyclerview-adapter-and-viewholder-pattern>.

***Calendar configuration to using sliding date picker in Android Studio***

CodingWithMitch. (2017, April 6). Android beginner tutorial #25 - Datepicker Dialog [choosing a date from a Dialog pop-up]Coe. <https://www.youtube.com/watch?v=hwe1abDO2Ag>.

***How to utilize spinner in Android Studio***

Androchunk. (2019, February 8). Spinner android studio tutorial. <https://www.youtube.com/watch?v=LwtBupqUdzw>.

***How to utilize SwitchCompact***

(2021, March 30) <https://developer.android.com/reference/androidx/appcompat/widget/SwitchCompat>.

***Making TextView Scrollable***

(2021, March 30) [https://stackoverflow.com/questions/1748977/making-textview-scrollable-on-android.](https://stackoverflow.com/questions/1748977/making-textview-scrollable-on-android.%20)

***Enable and disable Button according to the text in EditText in Android*** (2021, March 29)  [https://stackoverflow.com/questions/8225245/enable-and-disable-button-according-to-the-text-in-edittext-in-android.](https://stackoverflow.com/questions/1748977/making-textview-scrollable-on-android.%20)

***Converting String to StringBuilder and vice-versa***

(2021, April 4)  [https://www.tutorialspoint.com/how-to-convert-string-to-stringbuilder-and-vice-versa-java.](https://stackoverflow.com/questions/1748977/making-textview-scrollable-on-android.%20)