FAKULTA RIADENIA A INFORMATIKY
ŽILINSKÁ UNIVERZITA V ŽILINE

Semestrálna práca

z predmetu vývoj aplikácií pre mobilné zariadenia

Kanban board

Damián Hrubják
Študijná skupina: 5ZYI23
2020/2021

Popis a analýza riešeného problému

Zadanie

Úlohou mojej semestrálnej práce bolo vytvoriť aplikáciu na spravovanie osobných projektov a úloh s názvom **Kanban board**. Táto aplikácia umožní používateľovi evidovať svoje projekty a úloh, ktoré k nim prislúchajú.

Podobné aplikácie

Na obchode Google Play existuje veľké množstvo aplikácií na spravovanie projektov a úloh, no každá aplikácia poskytuje používateľovi iné možnosti použitia.

Aplikácie:

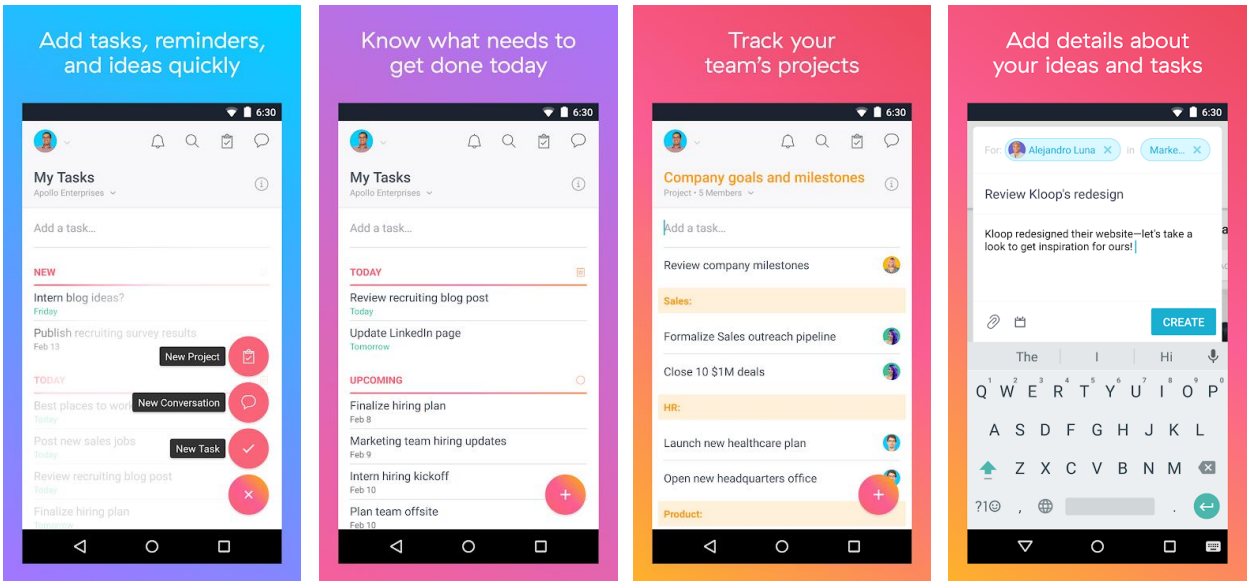
- Asana: Your work manager
- Trello
- ClickUp - Manage Teams & Tasks

Asana: Your work manager

Aplikácia umožňuje pridávať rozličné projekty, úlohy a jednotlivé úlohy prideliť členom tímu. Tiež umožňuje zobraziť projekty na časovú os, aby používatelia mali lepší prehľad o priebehu úloh.

Odlišnosti:

- Pridelenie úloh členom tímu a možnosť nastavenie priority
- Viac možností prispôsobenia projektov a úloh
- Pridávanie obrázkov a súborov
- Aplikácia funguje na všetkých platformách Web, iOS, Android

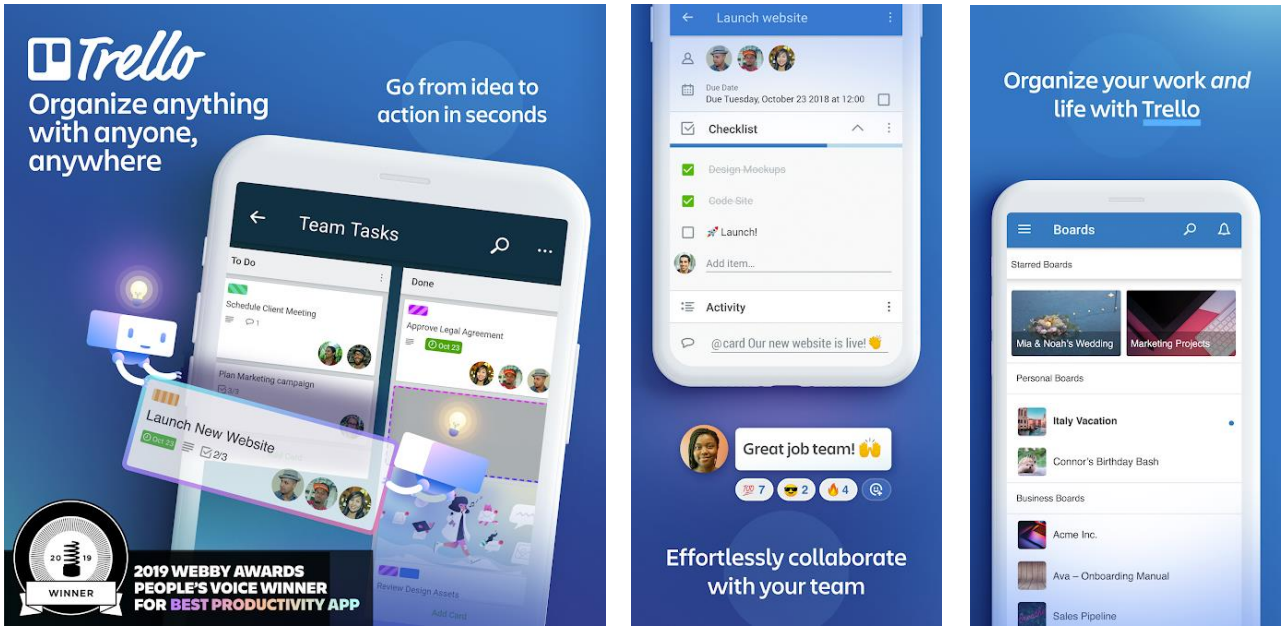


Trello

Jednoduchá na používanie. Okrem tradičných sekcii To do, Doing a Done, aplikácia umožňuje tieto sekcie premenovať, pridať ďalšie sekcie a pod.

Odlišnosti:

- Rôzne možnosti zobrazenia prvkov – karty, listy, tabuľka
- Pridávanie obrázkov a súborov a reakcií na úlohy a komentáre
- Aplikácia funguje na všetkých platformách Web, iOS, Android

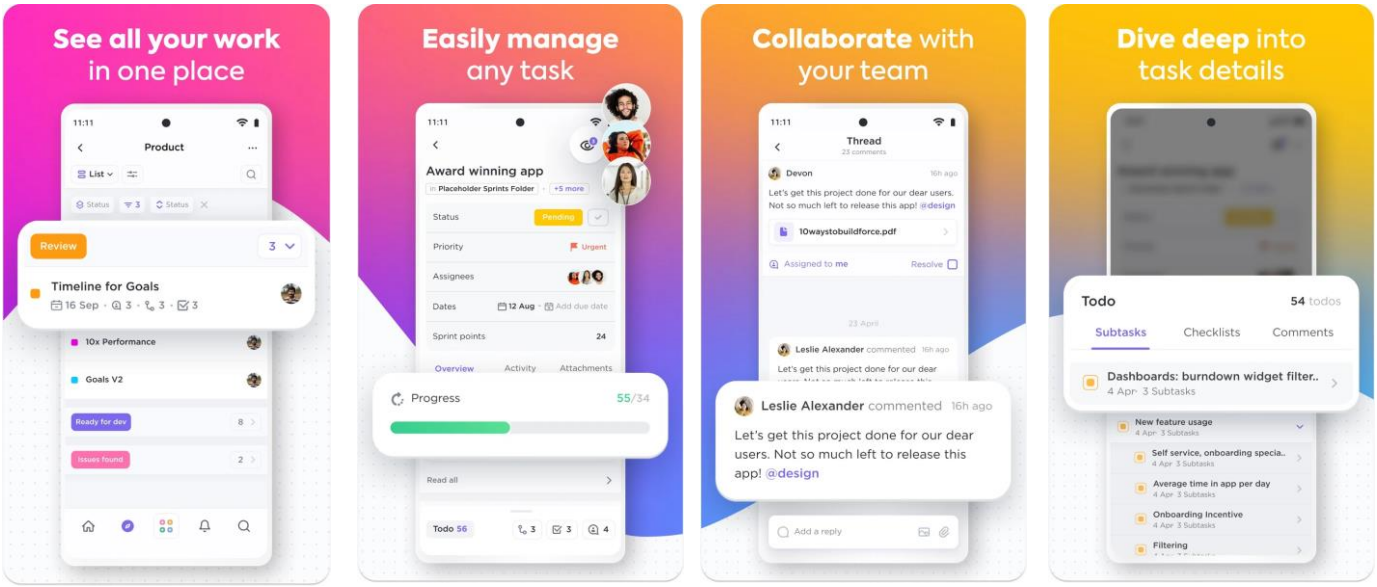


ClickUp - Manage Teams & Tasks

Aplikácia má najkrajší dizajn, pôsobí intuitívne. Poskytuje asi najlepší prehľad o úlohách. Tiež aplikácia zobrazuje rôzne štatistiky.

Odlišnosti:

- Pridelenie úloh členom tímu a možnosť nastavenie priority
- Pridávanie obrázkov a súborov a reakcií na úlohy a komentáre
- Prehľadný a čistý dizajn



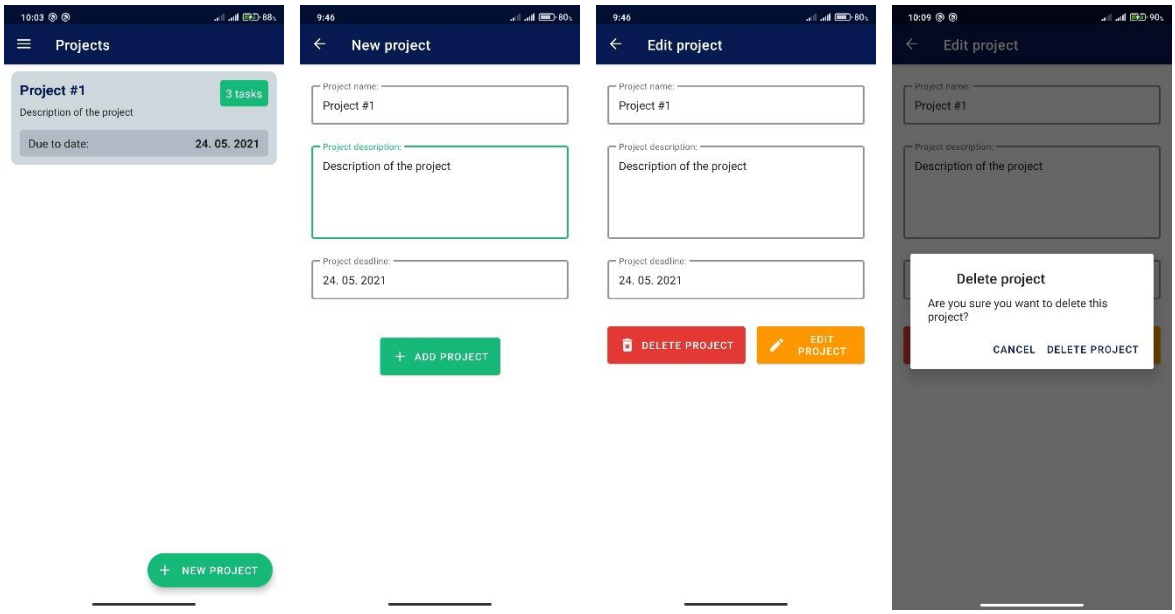
Analýza navrhovanej aplikácie

Aplikácia bude slúžiť na evidovanie osobných projektov a úloh. Používateľ môže každý projekt, alebo úlohu upraviť alebo vymazať. Pri jednotlivých úlohách môže používateľ meniť farbu a meniť stav úlohy, priamo na fragmente úloh. Aplikácia taktiež dokáže posielať notifikácie používateľovi, ak termín odovzdania projektu je blízko. Tieto notifikácie môže používateľ zapnúť, alebo vypnúť a tiež si môže prispôsobiť čas príchodu týchto notifikácií.

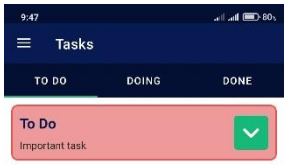
Návrh aplikácie

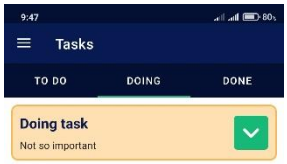
Aplikácia sa skladá z 2 aktivít, prvá aktivita je Splash Sreen aktivita a druhá aktivita, je hlavná aktivita aplikácie. Splash Sreen aktivita je aktivita, ktorá slúži ako úvodná obrazovka s logom a názvom aplikácie, ktorá sa zobrazí na 1.25 sekundy. Potom sa zobrazí úvodná strana, na ktorej budú vypísané projekty s ich termínom, názvom, a popisom projektu. Po kliku na projekt sa zobrazia všetky jeho úlohy v sekciách To do, Doing, Done. Keď používateľ podrží prst na projekt / úlohu, bude presmerovaný na fragment, kde môže úlohu / projekt upraviť, alebo vymazať. Navigačná zásuvka obsahuje Projekty a Nastavenia, kde používateľ môže zapnúť / vypnúť notifikácie, alebo prispôsobiť čas ich príchodu.

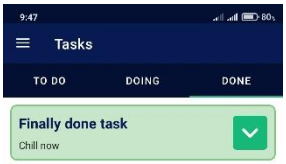
Okná projekty

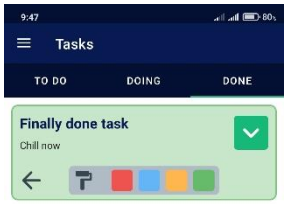


Okná úlohy


Tasks
TO DO
DOING
DONE
To Do
Important task


Tasks
TO DO
DOING
DONE
Doing task
Not so important


Tasks
TO DO
DOING
DONE
Finally done task
Chill now

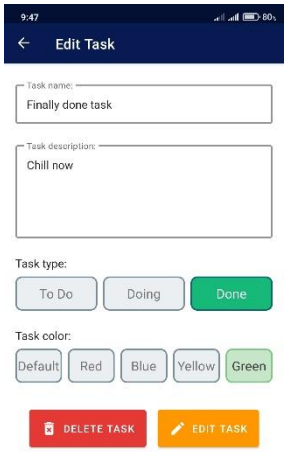

Tasks
TO DO
DOING
DONE
Finally done task
Chill now

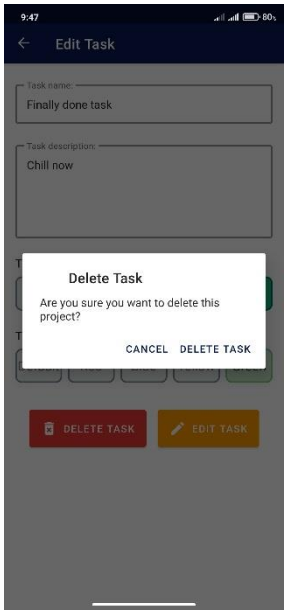

NEW TASK


NEW TASK


NEW TASK

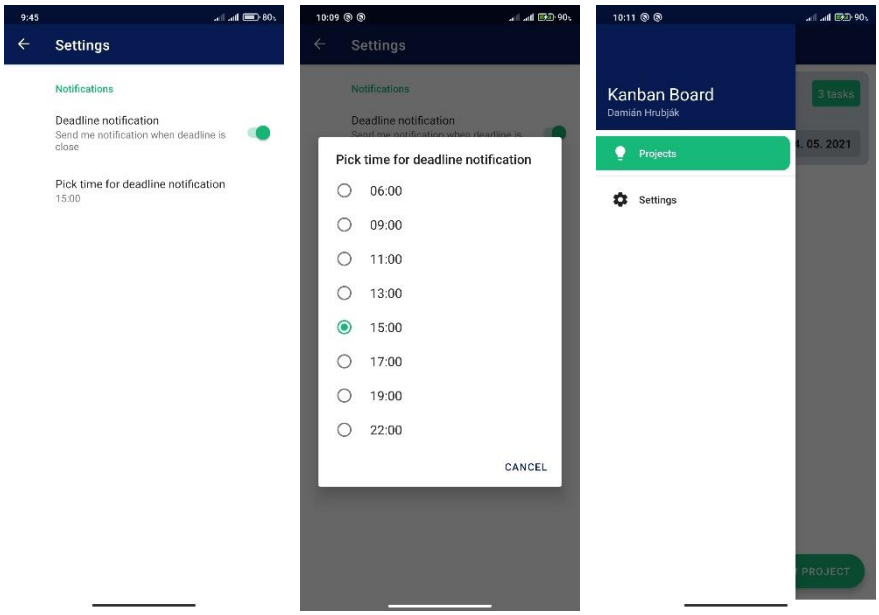

NEW TASK


Edit Task
Task name:
Finally done task
Task description:
Chill now
Task type:
To Do
Doing
Done
Task color:
Default
Red
Blue
Yellow
Green
DELETE TASK
EDIT TASK

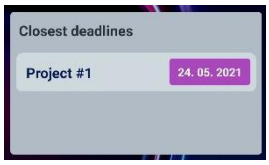

Edit Task
Task name:
Finally done task
Task description:
Chill now
Delete Task
Are you sure you want to delete this project?
CANCEL
DELETE TASK
DELETE TASK
EDIT TASK


New Task
Task name:
Task description:
Task type:
To Do
Doing
Done
Task color:
Default
Red
Blue
Yellow
Green
ADD TASK

Okná nastavenia a navigačná zásuvka



Widget a notifikácia



Popis implementácie

- Všetky texty, obrázky, backgroundy sú resources
- Aplikácia je napísaná v anglickom jazyku

```
<resources>
    <string name="app_name">Kanban Board</string>
    <string name="creator">Damián Hrubják</string>
    <string name="navigation_drawer_open">Open navigation drawer</string>
    <string name="navigation_drawer_close">Close navigation drawer</string>
    <string name="nav_header_desc">Navigation header</string>
    <string name="about_author">About author</string>
```

- Použil som 2 aktivity, hlavná aplikácia reaguje na callbacky, aby mohol fragment komunikovať s fragmentom

MainActivity

```
/**
 * Method for obtaining task from database
 *
 * @param projectID - ID of the project to which the tasks are assigned to
 * @param type - type of tasks (TO_DO, DOING, DONE)
 * @return if there are any task, return MutableList of Task, otherwise return empty list
 */
override fun getTaskListFromDatabase(projectID: Long, type: TaskType): MutableList<Task>? {
    return if(projectID != 0L) database.getTasksAssignedToProjectByProjectIDAndType(
        projectID,
        type
    ) else mutableListOf()
}
```

TabFragment

```
/**
 * Define interface of callbacks
 *
 */
interface Callbacks {
    fun addTaskToViewModel(task: Task, type: TaskType)
    fun getTaskListFromDatabase(projectID: Long, type: TaskType): MutableList<Task>?
}
```

- Použil som 7 fragmentov
- Aplikácia uchováva dáta pri otočení obrázovky
- Snažil som sa vytvoriť čo najlepšie UX a UI

- Aby sa mi zobrazovali notifikácie, musel som použiť BroadCast Receiver a AlarmManager, notifikácie môžete zapnúť, alebo vypnúť, taktiež je možné prispôsobiť čas príchodu notifikácie

```kotlin
/**
 * This method schedules alarm for notification
 * Creates intent of ReminderBroadcast
 *
 * @param projectName
 * @param afterDeadline
 * @param milliseconds
 */
private fun setAlarm(projectName: String, afterDeadline: Boolean, milliseconds: Long) {
    val context = requireContext()
    val intent = Intent(context, ReminderBroadcast::class.java)
    intent.apply { this: Intent
        action = "send.notification.extras"
        putExtra( name: "projectName", projectName)
        putExtra( name: "afterDeadline", afterDeadline)
    }


    val pendingIntent = PendingIntent.getBroadcast(context, requestCode: 0, intent, flags: 0)
    val alarmManager = context.getSystemService(ALARM_SERVICE) as AlarmManager

    alarmManager.setRepeating(
        AlarmManager.RTC_WAKEUP,
        milliseconds,
        AlarmManager.INTERVAL_DAY,
        pendingIntent
    )
}
```

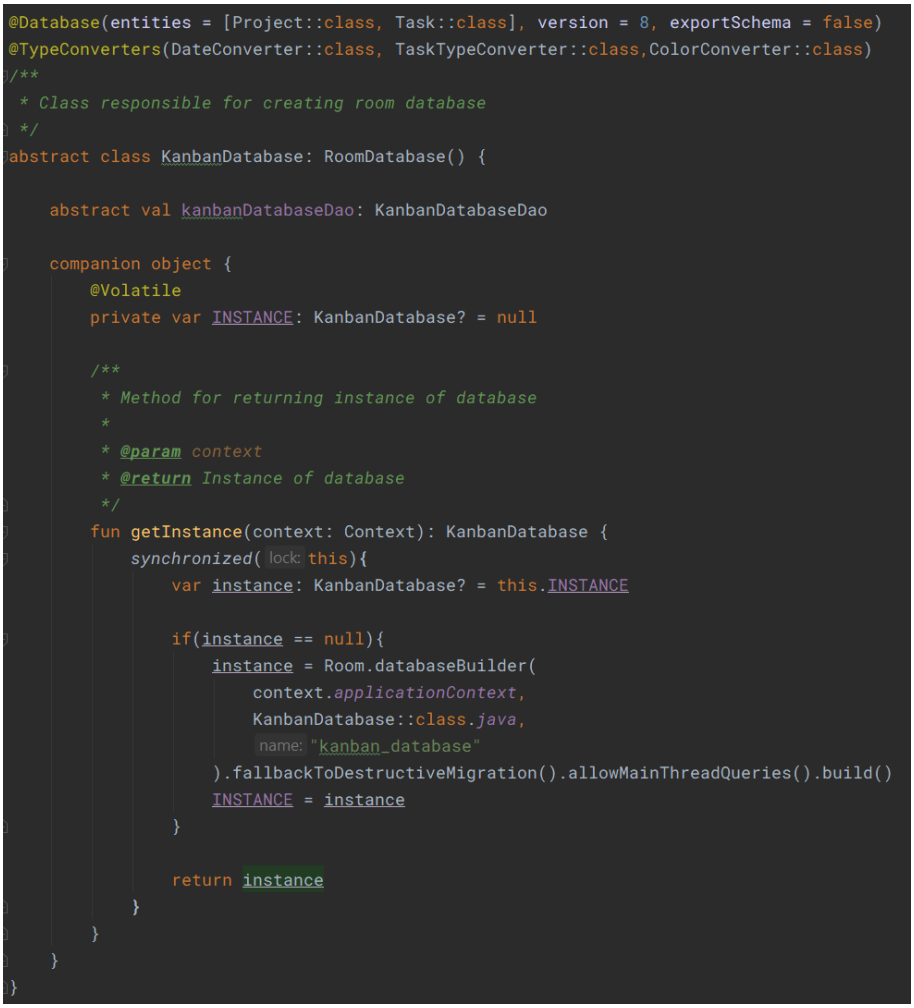
- Použil som databinding a Android navigáciu
- Vytvoril som widget, aby používateľ mal prehľad o najbližších projektov
- Na zobrazenie všetkých prvkov zoznamu som použil RecyclerView s adaptérom a ViewHolder
- Aby boli úlohy v troch sekciách, musel som použiť ViewPager2

```kotlin
/**
 * Adapter class for view pager
 *
 * @constructor
 * TODO
 *
 * @param fa FragmentActivity
 */
private inner class TasksFragmentStateAdapter(fa: FragmentActivity) : FragmentStateAdapter(fa) {
    override fun createFragment(position: Int): Fragment {
        //Create argument bundle with task list type
        val tasklistFragmentArguments = Bundle().apply { this: Bundle
            putInt("tasks_type", position)
            putLong("project_ID", projectID)
        }

        //Attach the argument bundle to new fragment instance and return the fragment
        return TabFragment().apply { this: TabFragment
            arguments = tasklistFragmentArguments
        }
    }

    override fun getItemCount(): Int = 3 // three categories
}
```

- Na uchovávanie údajov som použil Room Database, ktorá pozostáva z dvoch tabuliek a vzťahom medzi nimi. Databáza mi umožňuje ukladať, vyberať, upravovať a vymazať dáta



```
@Database(entities = [Project::class, Task::class], version = 8, exportSchema = false)
@TypeConverters(DateConverter::class, TaskTypeConverter::class,ColorConverter::class)
/**
 * Class responsible for creating room database
 */
abstract class KanbanDatabase: RoomDatabase() {

    abstract val kanbanDatabaseDao: KanbanDatabaseDao

    companion object {
        @Volatile
        private var INSTANCE: KanbanDatabase? = null

        /**
         * Method for returning instance of database
         *
         * @param context
         * @return Instance of database
         */
        fun getInstance(context: Context): KanbanDatabase {
            synchronized( lock: this){
                var instance: KanbanDatabase? = this.INSTANCE

                if(instance == null){
                    instance = Room.databaseBuilder(
                        context.applicationContext,
                        KanbanDatabase::class.java,
                         name: "kanban_database"
                    ).fallbackToDestructiveMigration().allowMainThreadQueries().build()
                    INSTANCE = instance
                }

                return instance
            }
        }
    }
}
```