|  |  |  |
| --- | --- | --- |
|  |  |  |
| МИНОБРНАУКИ РОССИИ | | |
| Федеральное государственное бюджетное образовательное учреждение  высшего образования  **«МИРЭА – Российский технологический университет»**  **РТУ МИРЭА** | | |

Институт Информационных технологий

Кафедра Математического обеспечения и стандартизации информационных технологий

**Отчет по практической работе №3**

по дисциплине «Разработка мобильных приложений»

|  |  |
| --- | --- |
| **Выполнил:**  студент группыИКБО-28-22 | Некрасов Г.А. |
| **Проверил:**  к.э.н доцент | Степанов П.В. |

Москва 2024 г.

СОДЕРЖАНИЕ

[ПРАКТИЧЕСКОЕ ЗАДАНИЕ 3](#_Toc184915617)

[Код практического задания 4](#_Toc184915618)

[Тестирование практического задания 10](#_Toc184915619)

[Вывод 12](#_Toc184915620)

# ПРАКТИЧЕСКОЕ ЗАДАНИЕ

Используя Android Architecture Components преобразовать структуру проекта из практической работы №2.

# Код практического задания

В ходе выполнения работы, мы написали код для реализации Android Architecture Components в фрагментах (см. Листинги 1-11)

Листинг 1. UserDao.kt

|  |
| --- |
| package com.example.pract3.dao  import androidx.room.Dao import androidx.room.Insert import androidx.room.Query import com.example.pract3.models.User  @Dao interface UserDao {  @Insert  suspend fun insert(user: User)   @Query("SELECT \* FROM users WHERE id = :id")  suspend fun getUser(id: Int): User } |

Листинг 2. User.kt

|  |
| --- |
| package com.example.pract3.models  import androidx.room.Entity import androidx.room.PrimaryKey  @Entity(tableName = "users") data class User(  @PrimaryKey(autoGenerate = true) val id: Int,  val name: String,  val dob: String,  val email: String ) |

Листинг 3. UserRepository.kt

|  |
| --- |
| package com.example.pract3.repository  import androidx.lifecycle.LiveData import androidx.lifecycle.MutableLiveData import androidx.lifecycle.liveData import com.example.pract3.dao.UserDao import com.example.pract3.models.User  class UserRepository(private val userDao: UserDao) {   val userLiveData = MutableLiveData<User>()   suspend fun insert(user: User) {  userDao.insert(user)  }   fun getUser(id: Int): LiveData<User> {  return liveData {  val user = userDao.getUser(id)  emit(user)  }  } } |

Листинг 4. LoginViewModel.kt

|  |
| --- |
| package com.example.pract3.vm  import android.app.Application import androidx.lifecycle.AndroidViewModel import androidx.lifecycle.LiveData import com.example.pract3.AppDatabase import com.example.pract3.models.User import com.example.pract3.repository.UserRepository  class LoginViewModel(application: Application) : AndroidViewModel(application) {   private val repository: UserRepository  val userLiveData: LiveData<User>   init {  val userDao = AppDatabase.getDatabase(application).userDao()  repository = UserRepository(userDao)  userLiveData = repository.userLiveData  }   fun login(email: String, password: String) {  // Логика авторизации  } } |

Листинг 5. ProfileViewModel.kt

|  |
| --- |
| package com.example.pract3.vm  import android.app.Application import androidx.lifecycle.AndroidViewModel import androidx.lifecycle.LiveData import com.example.pract3.AppDatabase import com.example.pract3.models.User import com.example.pract3.repository.UserRepository  class ProfileViewModel(application: Application) : AndroidViewModel(application) {   private val repository: UserRepository  val userLiveData: LiveData<User>   init {  val userDao = AppDatabase.getDatabase(application).userDao()  repository = UserRepository(userDao)  userLiveData = repository.getUser(1)  } } |

Листинг 6. SettingsViewModel.kt

|  |
| --- |
| package com.example.pract3.vm  import android.app.Application import androidx.lifecycle.AndroidViewModel import androidx.lifecycle.LiveData import androidx.lifecycle.MutableLiveData  class SettingsViewModel(application: Application) : AndroidViewModel(application) {   private val \_notificationsEnabled = MutableLiveData<Boolean>()  val notificationsEnabled: LiveData<Boolean> get() = \_notificationsEnabled   fun setNotificationsEnabled(enabled: Boolean) {  \_notificationsEnabled.value = enabled  } } |

Листинг 7. AppDatabase.kt

|  |
| --- |
| package com.example.pract3  import android.content.Context import androidx.room.Database import androidx.room.Room import androidx.room.RoomDatabase import com.example.pract3.dao.UserDao import com.example.pract3.models.User  @Database(entities = [User::class], version = 1) abstract class AppDatabase : RoomDatabase() {  abstract fun userDao(): UserDao   companion object {  @Volatile  private var INSTANCE: AppDatabase? = null   fun getDatabase(context: Context): AppDatabase {  return INSTANCE ?: synchronized(this) {  val instance = Room.databaseBuilder(  context.applicationContext,  AppDatabase::class.java,  "app\_database"  ).build()  INSTANCE = instance  instance  }  }  } } |

Листинг 8. LoginFragment.kt

|  |
| --- |
| package com.example.pract3  import android.os.Bundle import androidx.fragment.app.Fragment import android.view.View import android.widget.Button import androidx.lifecycle.ViewModelProvider import androidx.navigation.fragment.findNavController import com.example.pract3.vm.LoginViewModel   class LoginFragment : Fragment(R.layout.fragment\_login) {   private lateinit var viewModel: LoginViewModel   override fun onViewCreated(view: View, savedInstanceState: Bundle?) {  super.onViewCreated(view, savedInstanceState)   viewModel = ViewModelProvider(this).get(LoginViewModel::class.java)   val loginButton = view.findViewById<Button>(R.id.loginButton)  val registerButton = view.findViewById<Button>(R.id.registerButton)   loginButton.setOnClickListener {  findNavController().navigate(R.id.action\_loginFragment\_to\_profileFragment)  // Логика авторизации  }   registerButton.setOnClickListener {  // Логика регистрации  }  } } |

Листинг 9. MainActivity.kt

|  |
| --- |
| package com.example.pract3  import android.os.Bundle import androidx.appcompat.app.AppCompatActivity import androidx.navigation.fragment.NavHostFragment import androidx.navigation.ui.NavigationUI import com.example.pract3.R import com.google.android.material.bottomnavigation.BottomNavigationView  class MainActivity : AppCompatActivity() {  override fun onCreate(savedInstanceState: Bundle?) {  super.onCreate(savedInstanceState)  setContentView(R.layout.activity\_main)   // Находим NavHostFragment  val navHostFragment = supportFragmentManager.findFragmentById(R.id.nav\_host\_fragment) as NavHostFragment  // Получаем NavController из NavHostFragment  val navController = navHostFragment.navController   // Связываем BottomNavigationView с NavController  val bottomNavigationView = findViewById<BottomNavigationView>(R.id.bottomNavigationView)  NavigationUI.setupWithNavController(bottomNavigationView, navController)   // Управляем логикой навигации через NavigationUI  bottomNavigationView.setOnItemSelectedListener { item ->  when (item.itemId) {  R.id.nav\_login -> {  navController.navigate(R.id.loginFragment)  true  }  R.id.nav\_profile -> {  navController.navigate(R.id.profileFragment)  true  }  R.id.nav\_settings -> {  navController.navigate(R.id.settingsFragment)  true  }  else -> false  }  }  } } |

Листинг 10. ProfileFragment.kt

|  |
| --- |
| package com.example.pract3  import android.os.Bundle import androidx.fragment.app.Fragment import android.view.View import android.widget.TextView import androidx.lifecycle.ViewModelProvider import com.example.pract3.vm.ProfileViewModel  class ProfileFragment : Fragment(R.layout.fragment\_profile) {   private lateinit var viewModel: ProfileViewModel   override fun onViewCreated(view: View, savedInstanceState: Bundle?) {  super.onViewCreated(view, savedInstanceState)   viewModel = ViewModelProvider(this).get(ProfileViewModel::class.java)   val nameTextView = view.findViewById<TextView>(R.id.nameTextView)  val emailTextView = view.findViewById<TextView>(R.id.emailTextView)  val dobTextView = view.findViewById<TextView>(R.id.birthDateTextView)   // Убедитесь, что данные пользователя не `null` перед отображением  viewModel.userLiveData.observe(viewLifecycleOwner) { user ->  user?.let {  nameTextView.text = it.name  emailTextView.text = it.email  dobTextView.text = it.dob  } ?: run {  // Данные пользователя еще не загружены  nameTextView.text = "Loading..."  emailTextView.text = "Loading..."  dobTextView.text = "Loading..."  }  }  } } |

Листинг 11. SettingsFragment.kt

|  |
| --- |
| package com.example.pract3  import android.os.Bundle import androidx.fragment.app.Fragment import android.view.View import android.widget.CheckBox import androidx.lifecycle.ViewModelProvider import com.example.pract3.vm.SettingsViewModel  class SettingsFragment : Fragment(R.layout.fragment\_settings) {   private lateinit var viewModel: SettingsViewModel   override fun onViewCreated(view: View, savedInstanceState: Bundle?) {  super.onViewCreated(view, savedInstanceState)   viewModel = ViewModelProvider(this).get(SettingsViewModel::class.java)   val checkBox = view.findViewById<CheckBox>(R.id.notificationsCheckBox)   viewModel.run {  notificationsEnabled.observe(viewLifecycleOwner) { enabled ->  checkBox.isChecked = enabled  }  }   checkBox.setOnCheckedChangeListener { \_, isChecked ->  viewModel.setNotificationsEnabled(isChecked)  }  } } |

# Тестирование практического задания

Выполним проверку работы нашего кода (см. Рисунки 1-3)

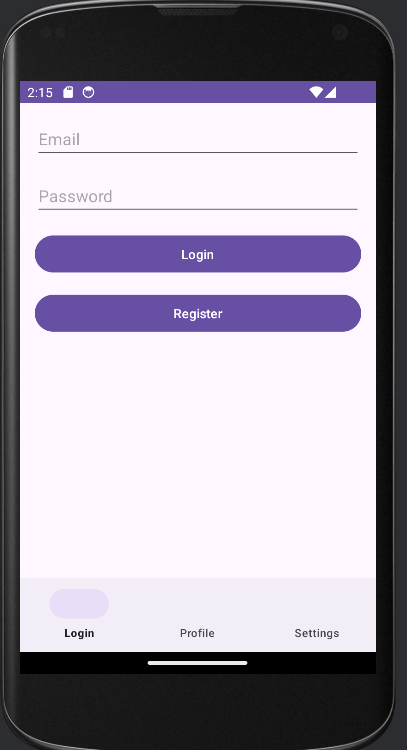


Рисунок 1. Фрагмент Login

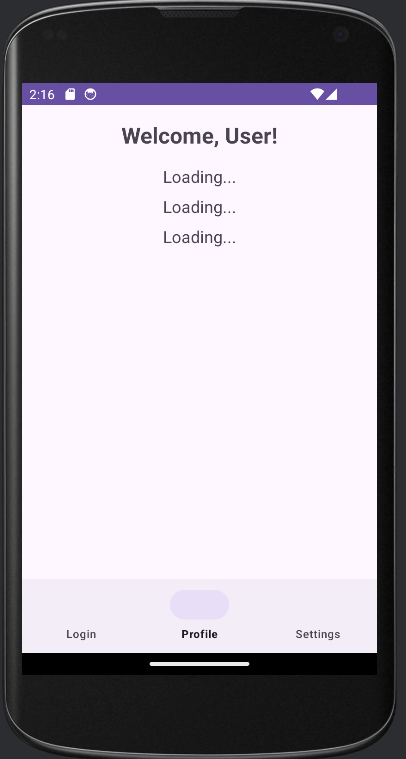


Рисунок 2. Фрагмент Profile

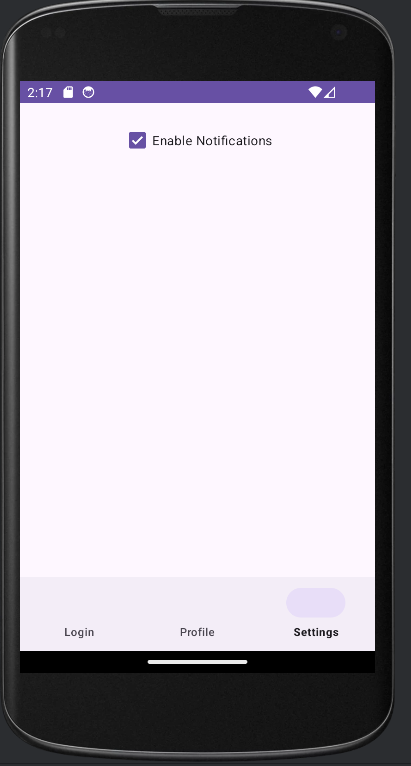


Рисунок 3. Фрагмент Settings

# Вывод

В процессе выполнения работы мы узнали, как работает структура Android Architecture Components, а также её реализовали