

Projecte Android en Base de dades:

Gestor de Biblioteca Personal



Professor: Jordi Vega


Assignatura: M8 Programació multimèdia i dispositius mòbils

Nom: Harvey

Cognoms: John Glover

INDEX:

Requisits:	3
Explicació de la temàtica	3
Detalls de l'aplicació	4
Estructura de l'app en tres fragments principals	4
Disseny i estil	4
Captures de pantalla i descripció	5
Fitxers afegits i modificats	6
Llibreries i dependències	7
build.gradle.kts (GestorBiblioteca)	7
build.gradle.kts (App)	8
DATA	10
LibraryDao.kt	10
LibraryDatabase.kt	11
LibraryItem.kt	11
UI	12
DetailFragment.kt	12
EditFragment.kt	14
LibraryAdapter.kt	16
MainFragment.kt	17
VIEWMODEL	18
LibraryViewModel.kt	18
MainActivity.kt	19
DRAWABLE	20
LAYOUT	21
activity_main.xml	21
fragment_detail.xml	22
fragment_edit.xml	26
fragment_main.xml	30
item_library.xml	31
VALUES	32
colors.xml	32
strings.xml	33
themes.xml	33
nav_graph.xml	34
Proves i comprovacions	36
Captures del funcionament del CRUD	36
Verificació amb Database Inspector d'Android Studio	37

 INSTITUT DE L'EBRE TORTOSA	M8 Programació multimèdia i dispositius mòbils	DAM 2n
--	--	------------------

Requisits:

- Base de dades en Android Room i sqllite.
- CRUD (s'han d'utilitzar data entities, interfícies DAO).
- Fragments (per a les diferents pàgines)
- RecyclerViews.
- Arquitectura NVVM
- Generar .apk

Explicació de la temàtica

L'aplicació "Gestor de Biblioteca Personal" està dissenyada per ajudar els usuaris a organitzar i gestionar la seva col·lecció personal de llibres, pel·lícules i sèries. Aquesta eina permet centralitzar la informació en un sol lloc, fent que sigui més fàcil accedir-hi, actualitzar-la i revisar-la en qualsevol moment.

El propòsit principal és proporcionar una solució senzilla i intuïtiva per als usuaris que vulguin tenir un registre complet de les seves col·leccions. Això inclou característiques com afegir nous elements, editar els existents i eliminar els que ja no siguin necessaris. La funcionalitat de cerca i la visualització detallada també contribueixen a una experiència fluida i eficaç.



Detalls de l'aplicació

Estructura de l'app en tres fragments principals

1. Llista Principal

- Mostra tots els elements de la col·lecció en un **RecyclerView**.
- Cada element es representa amb el títol, el tipus (llibre, sèrie o pel·lícula) i una descripció breu.
- Inclou un botó per afegir nous elements i un sistema de navegació cap a la pantalla de detall o edició.

2. Detall de l'Element

- Mostra la informació completa d'un element seleccionat, com el títol, el tipus i la descripció.
- Disposa de dos botons: "Editar" per modificar l'element i "Esborrar" per eliminar-lo.

3. Edició/Creació d'Elements

- Aquesta pantalla permet afegir un nou element o editar un existent.
- El formulari inclou camps de text per al títol, el tipus (amb un desplegable) i la descripció.
- Un botó de guardar valida la informació i la desa a la base de dades.


Disseny i estil

● Estètica general

- Disseny modern i minimalista, amb una paleta de colors personalitzada que inclou tons clars i foscos.
- L'usuari pot canviar el tema entre mode clar i mode fosc des de la configuració del sistema.

● Fragments

- La llista principal té un disseny amb marges generosos i espais entre elements, donant una aparença neta i organitzada.
- El detall de l'element té un layout centrat, amb els textos alineats i botons grans per facilitar la interacció.
- El formulari d'edició inclou camps de text clars i ítems seleccionables, amb validació en temps real per evitar errors.

 INSTITUT DE L'EBRE TORTOSA	M8 Programació multimèdia i dispositius mòbils	DAM 2n
--	--	------------------

Captures de pantalla i descripció

1. Pantalla de Llista Principal

- La captura mostra una llista scrollable amb diversos elements, cadascun amb el títol i el tipus. A la part superior hi ha un botó flotant per afegir un nou element.

2. Pantalla de Detall de l'Element

- Mostra un exemple d'element seleccionat, amb el títol en gran i una descripció detallada. Els botons d'editar i "esborrar" són visibles a la part inferior.

3. Pantalla d'Edició/Creació

- Inclou un formulari complet amb camps editables per afegir o modificar la informació. La validació del formulari es mostra amb icones i missatges d'error.

Fitxers afegits i modificats

```
alumnat@harvey:~/Code/M7-8-9/Android/GestorBiblioteca/app/src/main/java/com/harvey/gestorbiblioteca$ tree
.
├── data
│   ├── LibraryDao.kt
│   ├── LibraryDatabase.kt
│   ├── LibraryItem.kt
│   └── LibraryRepository.kt
├── MainActivity.kt
├── ui
│   ├── DetailFragment.kt
│   ├── EditFragment.kt
│   ├── LibraryAdapter.kt
│   ├── MainFragment.kt
│   └── theme
│       ├── Color.kt
│       ├── Theme.kt
│       └── Type.kt
└── viewmodel
    └── LibraryViewModel.kt
```

```
alumnat@harvey:~/Code/M7-8-9/Android/GestorBiblioteca/app/src/main/res$ tree
.
├── drawable
│   ├── ic_add.xml
│   ├── ic_delete.xml
│   ├── ic_edit.xml
│   ├── ic_info.xml
│   ├── ic_launcher_background.xml
│   ├── ic_launcher_foreground.xml
│   └── ic_save.xml
├── layout
│   ├── activity_main.xml
│   ├── fragment_detail.xml
│   ├── fragment_edit.xml
│   ├── fragment_main.xml
│   └── item_library.xml
├── navigation
│   └── nav_graph.xml
├── values
│   ├── colors.xml
│   ├── strings.xml
│   └── themes.xml
└── xml
    ├── backup_rules.xml
    └── data_extraction_rules.xml
```

Llibreries i dependències

build.gradle.kts (GestorBiblioteca)

Aquest és un fitxer de construcció de nivell superior per a un projecte Android. Aquí és on configurem les opcions que seran comunes a tots els subprojectes o mòduls del teu projecte.

1. **plugins:** Defineix els plugins que s'utilitzaran en el projecte. En aquest cas, s'estan definint tres plugins:
 - android.application: Plugin per a aplicacions Android, però no està aplicat (apply false).
 - kotlin.android: Plugin per a suportar Kotlin en Android, però no està aplicat (apply false).
 - kotlin.compose: Plugin per a suportar Kotlin Compose, que sí està aplicat (apply true).
2. **buildscript:** Defineix les dependències i repositoris necessaris per a la configuració del projecte.
 - **repositories:** Defineix els repositoris Maven utilitzats per resoldre les dependències. google() afegeix el repositori Google Maven, i mavenCentral() afegeix el repositori Maven Central.
 - **dependencies:** Defineix les dependències necessàries per al projecte de construcció. En aquest cas, s'afegeix el plugin de navegació segura (libs.navigation.safe.args.gradle.plugin).

```
build.gradle.kts (GestorBiblioteca) x
1 // Arxiu de construcció de nivell superior on podeu afegir opcions de configuració comunes a tots els subprojectes/mòduls.
2 plugins {
3     alias(libs.plugins.android.application) apply false // Alias per al plugin d'aplicació Android, aplicat com a fals
4     alias(libs.plugins.kotlin.android) apply false // Alias per al plugin Kotlin per Android, aplicat com a fals
5     alias(libs.plugins.kotlin.compose) apply true // Alias per al plugin Kotlin Compose, aplicat com a cert
6 }
7
8 buildscript {
9     repositories {
10         google() // Assegurar-se que el repositori Google Maven estigui disponible
11         mavenCentral() // Assegurar-se que Maven Central estigui disponible
12     }
13     dependencies {
14         classpath(libs.navigation.safe.args.gradle.plugin) // Dependència per al plugin de navegació segura
15     }
16 }
```

build.gradle.kts (App)

Aquest fitxer de construcció per a Android configura els plugins, dependències, i altres opcions per al projecte. Utilitza Kotlin i Compose, amb suport per a navegació segura i vinculació de vistes, entre altres.

```

1  import org.jetbrains.kotlin.storage.CacheResetOnProcessCanceled.enabled
2
3  plugins {
4      alias(libs.plugins.android.application)
5      alias(libs.plugins.kotlin.android)
6      alias(libs.plugins.kotlin.compose) apply true
7      id ("androidx.navigation.safeargs.kotlin")
8      id ("kotlin-kapt")
9  }

```

```

buildFeatures {
    compose = true
    viewBinding = true
}
composeOptions {
    kotlinCompilerExtensionVersion = "1.5.0"
}

```

```

dependencies {
    // Room and Navigation
    implementation(libs.androidx.room.runtime)
    implementation(libs.androidx.navigation.fragment.ktx)
    implementation(libs.androidx.navigation.ui.ktx)
    implementation(libs.androidx.navigation.compose)

    // Compose dependencies
    implementation(libs.ui)
    implementation(libs.material3)
    implementation(libs.ui.tooling.preview)
    implementation(libs.androidx.activity.compose.v131)
    implementation(libs.androidx.navigation.compose.v240alpha10)

    // Room compiler // noinspection KaptUsageInsteadOfKsp
    kapt(libs.androidx.room.compiler)
    implementation(libs.androidx.room.ktx)
}

```



```
// Lifecycle and LiveData
implementation(libs.androidx.lifecycle.runtime.ktx)
implementation(libs.androidx.lifecycle.viewmodel.ktx)
implementation(libs.androidx.lifecycle.livedata.ktx)

// RecyclerView and Core
implementation(libs.androidx.recyclerview)
implementation(libs.androidx.core.ktx)

// Compose UI and Material
implementation(platform(libs.androidx.compose.bom))
implementation(libs.androidx.ui)
implementation(libs.androidx.ui.graphics)
implementation(libs.material)
implementation(libs.androidx.ui.tooling.preview)
implementation(libs.androidx.material3)
implementation(libs.androidx.cardview)

// Testing
testImplementation(libs.junit)
androidTestImplementation(libs.androidx.junit)
androidTestImplementation(libs.androidx.espresso.core)
androidTestImplementation(platform(libs.androidx.compose.bom))
androidTestImplementation(libs.androidx.ui.test.junit4)
debugImplementation(libs.androidx.ui.tooling)
debugImplementation(libs.androidx.ui.test.manifest)
```

Sincronitzo el fitxer abans de continuar

Gradle files have changed since last project sync. A project sync may be necessary... [Sync Now](#)

DATA

LibraryDao.kt

Aquest codi defineix una interfície LibraryDao per a accedir a la base de dades Room. Inclou funcions per obtenir tots els elements, obtenir un element per ID, inserir, actualitzar i eliminar elements de la taula library_items.

```
LibraryDao.kt x
1 package com.harvey.gestorbiblioteca.data
2
3 import androidx.lifecycle.LiveData
4 import androidx.room.*
5
6 @Dao
7 interface LibraryDao {
8     @Query("SELECT * FROM library_items ORDER BY title ASC")
9     fun getAllItems(): LiveData<List<LibraryItem>> // Obtenir tots els elements ordenats per títol en ordre ascendent
10
11     @Query("SELECT * FROM library_items WHERE id = :itemId")
12     fun getItemById(itemId: Int): LiveData<LibraryItem> // Obtenir un element per ID en temps real
13
14     @Query("SELECT * FROM library_items WHERE id = :itemId")
15     suspend fun getItemByIdSync(itemId: Int): LibraryItem? // Obtenir un element per ID de manera sincrònica
16
17     @Insert(onConflict = OnConflictStrategy.REPLACE)
18     suspend fun insertItem(item: LibraryItem) // Inserir un element, reemplaçant-lo si ja existeix
19
20     @Update
21     suspend fun updateItem(item: LibraryItem) // Actualitzar un element existent
22
23     @Delete
24     suspend fun deleteItem(item: LibraryItem) // Eliminar un element existent
25 }
```

LibraryDatabase.kt

Aquest codi defineix la base de dades LibraryDatabase utilitzant Room. Crea una instància única de la base de dades i ofereix accés a les funcions del DAO (Data Access Object).

```
LibraryDatabase.kt x
1 package com.harvey.gestorbiblioteca.data
2
3 import android.content.Context
4 import androidx.room.Database
5 import androidx.room.Room
6 import androidx.room.RoomDatabase
7
8 @Database(entities = [LibraryItem::class], version = 1, exportSchema = false)
9 // Defineix la base de dades Room amb l'entitat LibraryItem
10 @abstract class LibraryDatabase : RoomDatabase() {
11     abstract fun libraryDao(): LibraryDao // Declaració de la funció abstracta per obtenir l'objecte DAO
12
13     companion object {
14         @Volatile
15         private var INSTANCE: LibraryDatabase? = null // Instància única de la base de dades
16
17         fun getDatabase(context: Context): LibraryDatabase {
18             return INSTANCE ?: synchronized(lock, this) { // Obté la instància de la base de dades de manera sincronitzada
19                 val instance = Room.databaseBuilder(
20                     context.applicationContext,
21                     LibraryDatabase::class.java,
22                     name: "library_database"
23                 ).build()
24                 INSTANCE = instance
25                 instance
26             }
27         }
28     }
29 }
```

LibraryItem.kt

Aquest codi defineix una entitat de la base de dades Room anomenada LibraryItem, amb una taula library_items. Cada article de la biblioteca té un identificador autogenerat (id), un títol (title), una descripció (description), un autor (author), un any de publicació (year) i un gènere (genre).

```
LibraryItem.kt x
1 package com.harvey.gestorbiblioteca.data
2
3 import androidx.room.Entity
4 import androidx.room.PrimaryKey
5
6 @Entity(tableName = "library_items") // Defineix una entitat de la base de dades amb el nom de taula "library_items"
7 data class LibraryItem(
8     @PrimaryKey(autoGenerate = true) val id: Int = 0, // Clau primària autogenerada
9     val title: String, // Títol de l'article de la biblioteca
10     val description: String, // Descripció de l'article
11     val author: String, // Autor de l'article
12     val year: Int, // Any de publicació
13     val genre: String // Gènere de l'article
14 )
```

UI

DetailFragment.kt

Aquest codi defineix un fragment de detall (DetailFragment) utilitzant ViewModel i navegació segura. Mostra els detalls d'un element de la biblioteca i permet eliminar-lo o editar-lo

```
DetailFragment.kt x
1 package com.harvey.gestorbiblioteca.ui
2
3 > import ...
14
15 class DetailFragment : Fragment() {
16     private var _binding: FragmentDetailBinding? = null
17     private val binding get() = _binding!! // Vinculació de vistes
18     private val viewModel: LibraryViewModel by viewModels() // Instància del ViewModel
19     private val args: DetailFragmentArgs by navArgs() // Arguments de navegació
20
21     override fun onCreateView(
22         inflater: LayoutInflater, container: ViewGroup?,
23         savedInstanceState: Bundle?
24     ): View {
25         _binding = FragmentDetailBinding.inflate(inflater, container, attachToParent: false)
26         // Infla el layout per a aquest fragment
27         return binding.root // Retorna la vista arrel
28     }
29
30     override fun onViewCreated(view: View, savedInstanceState: Bundle?) {
31         super.onViewCreated(view, savedInstanceState)
32
33         val itemId = args.itemId // Obté l'ID de l'element des dels arguments
34
35         viewModel.getItemById(itemId).observe(viewLifecycleOwner) { item ->
36             item?.let {
37                 binding.textViewTitle.text = it.title
38                 binding.textViewAuthor.text = it.author
39                 binding.textViewDescription.text = it.description
40                 binding.textViewYear.text = it.year.toString()
41                 binding.textViewGenre.text = it.genre
42             }
43         }
44     }
45 }
```

```
binding.buttonDelete.setOnClickListener {
    viewModel.getItemById(itemId).observe(viewLifecycleOwner) { item ->
        item?.let {
            viewModel.delete(it)
            Toast.makeText(requireContext(), text: "Llibre eliminat", Toast.LENGTH_SHORT).show()
            findNavController().navigateUp() // Navega cap amunt (retorna a l'anterior)
        }
    }
}

binding.buttonEdit.setOnClickListener {
    // Navega cap a EditFragment passant l'itemId per editar-lo
    val action = DetailFragmentDirections.actionDetailFragmentToEditFragment(itemId = itemId)
    findNavController().navigate(action)
}

override fun onDestroyView() {
    super.onDestroyView()
    _binding = null // Netegem la vinculació quan es destrueix la vista
}
}
```

EditFragment.kt

Aquest codi defineix un fragment d'edició (EditFragment) que permet carregar un element existent per editar-lo o crear-ne un de nou i desar-lo. Utilitza ViewModel i navegació segura amb Safe Args.

```

1 package com.harvey.gestorbiblioteca.ui
2
3 > import ...
4
15
16 class EditFragment : Fragment() {
17
18     private var _binding: FragmentEditBinding? = null
19     private val binding get() = _binding!! // Vinculació de vistes
20     private val viewModel: LibraryViewModel by viewModels() // Instància del ViewModel
21     private val args: EditFragmentArgs by navArgs() // s'utilitzen els safe args
22     private var currentItem: LibraryItem? = null // Element actual en edició
23
24     override fun onCreateView(
25         inflater: LayoutInflater, container: ViewGroup?,
26         savedInstanceState: Bundle?
27     ): View {
28         _binding = FragmentEditBinding.inflate(inflater, container, attachToParent: false)
29         // Infla el layout per a aquest fragment
30         return binding.root // Retorna la vista arrel
31     }
32
33     override fun onViewCreated(view: View, savedInstanceState: Bundle?) {
34         super.onViewCreated(view, savedInstanceState)
35
36         val itemId = args.itemId // Obté l'ID de l'element des dels arguments
37         if (itemId != -1) {
38             // Mode edició: carregar l'element existent
39             viewModel.getItemById(itemId).observe(viewLifecycleOwner) { item ->
40                 item?.let {
41                     currentItem = it
42                     binding.editTextTitle.setText(it.title)
43                     binding.editTextAuthor.setText(it.author)
44                     binding.editTextDescription.setText(it.description)
45                     binding.editTextYear.setText(it.year.toString())
46                     binding.editTextGenre.setText(it.genre)
47                 }
48             }
49         }
50     }
51 }

```

```
binding.buttonSave.setOnClickListener {
    val title = binding.editTextTitle.text.toString().trim()
    val author = binding.editTextAuthor.text.toString().trim()
    val description = binding.editTextDescription.text.toString().trim()
    val year = binding.editTextYear.text.toString().toIntOrNull() ?: 0
    val genre = binding.editTextGenre.text.toString().trim()

    // Validació bàsica
    if (title.isEmpty() || author.isEmpty()) {
        Toast.makeText(requireContext(), text: "El títol i l'autor són obligatoris", Toast.LENGTH_SHORT).show()
        return@setOnClickListener
    }
}
```

```
        if (itemId != -1 && currentItem != null) {
            // Actualització de l'element existent
            val updatedItem = currentItem!!.copy(
                title = title,
                author = author,
                description = description,
                year = year,
                genre = genre
            )
            viewModel.update(updatedItem)
            Toast.makeText(requireContext(), text: "Llibre actualitzat", Toast.LENGTH_SHORT).show()
        } else {
            // Inserció d'un nou element
            val newItem = LibraryItem(
                title = title,
                author = author,
                description = description,
                year = year,
                genre = genre
            )
            viewModel.insert(newItem)
            Toast.makeText(requireContext(), text: "Llibre desat", Toast.LENGTH_SHORT).show()
        }

        findNavController().popBackStack() // Torna enrere a la pila de navegació
    }
}

override fun onDestroyView() {
    super.onDestroyView()
    _binding = null // Netegem la vinculació quan es destrueix la vista
}
}
```

LibraryAdapter.kt

Aquest codi defineix un adaptador (LibraryAdapter) per a una RecyclerView en una aplicació Android. Gestiona la llista d'elements de la biblioteca i els mostra utilitzant View Binding. Permet també gestionar els clics sobre els elements de la llista.

```
LibraryAdapter.kt x
1
2
3 import android.view.LayoutInflater
4 import android.view.ViewGroup
5 import androidx.recyclerview.widget.RecyclerView
6 import com.harvey.gestorbiblioteca.data.LibraryItem
7 import com.harvey.gestorbiblioteca.databinding.ItemLibraryBinding
8
9 class LibraryAdapter(private val onItemClick: (LibraryItem) -> Unit) :
10     RecyclerView.Adapter<LibraryAdapter.LibraryViewHolder>() {
11
12     private var items: List<LibraryItem> = emptyList() // Llista d'elements de la biblioteca
13
14     fun submitList(newItems: List<LibraryItem>) {
15         items = newItems
16         notifyDataSetChanged() // Notifica que la llista ha canviat
17     }
18
19     override fun onCreateViewHolder(parent: ViewGroup, viewType: Int): LibraryViewHolder {
20         val binding = ItemLibraryBinding.inflate(LayoutInflater.from(parent.context), parent, attachToParent: false)
21         return LibraryViewHolder(binding) // Crea un nou ViewHolder
22     }
23
24     override fun onBindViewHolder(holder: LibraryViewHolder, position: Int) {
25         holder.bind(items[position]) // Lliga l'element a la vista
26     }
27
28     override fun getItemCount(): Int = items.size // Retorna el nombre d'elements
29
30     inner class LibraryViewHolder(private val binding: ItemLibraryBinding) :
31         RecyclerView.ViewHolder(binding.root) {
32         fun bind(item: LibraryItem) {
33             binding.textViewTitle.text = item.title
34             binding.textViewAuthor.text = item.author
35             itemView.setOnClickListener { onItemClick(item) } // Gestiona el clic de l'element
36         }
37     }
38 }
```


MainFragment.kt

Aquest codi defineix un fragment principal (MainFragment) que configura una RecyclerView amb un adaptador per mostrar una llista d'elements de la biblioteca i permet navegar a detalls i edició d'elements utilitzant ViewModel i navegació segura.

```
MainFragment.kt x
1 package com.harvey.gestorbiblioteca.ui
2
3 > import ...
14
15 class MainFragment : Fragment() {
16     private lateinit var binding: FragmentMainBinding // Vinculació de vistes
17     private val viewModel: LibraryViewModel by viewModels() // Instància del ViewModel
18
19     override fun onCreateView(
20         inflater: LayoutInflater, container: ViewGroup?,
21         savedInstanceState: Bundle?
22     ): View {
23         binding = FragmentMainBinding.inflate(inflater, container, attachToParent: false)
24         // Infla el layout per a aquest fragment
25         // Configura l'adaptador de la RecyclerView
26         val adapter = LibraryAdapter { item ->
27             val bundle = Bundle().apply {
28                 putInt("itemId", item.id)
29             }
30             findNavController().navigate(R.id.action_mainFragment_to_detailFragment, bundle)
31             // Navega cap a DetailFragment passant l'itemId
32         }
33         binding.recyclerView.adapter = adapter
34
35         // Configura el LayoutManager perquè la RecyclerView pugui disposar els seus elements
36         binding.recyclerView.layoutManager = LinearLayoutManager(requireContext())
37
38         // Observa els canvis i actualitza la llista de l'adaptador
39         viewModel.allItems.observe(viewLifecycleOwner) { items ->
40             adapter.submitList(items)
41         }
42
43         // Navega a la pantalla d'afegir/editar quan es fa clic al FAB
44         binding.fabAdd.setOnClickListener {
45             findNavController().navigate(R.id.action_mainFragment_to_editFragment)
46         }
47
48         return binding.root // Retorna la vista arrel
49     }
50 }
```

VIEWMODEL

LibraryViewModel.kt

Aquest codi defineix un `LibraryViewModel` per gestionar la lògica de dades d'una aplicació Android. Proporciona funcions per obtenir, inserir, actualitzar i eliminar elements de la biblioteca, utilitzant la base de dades Room i el DAO.

```
LibraryViewModel.kt x
1 package com.harvey.gestorbiblioteca.viewmodel
2
3 import android.app.Application
4 import androidx.lifecycle.*
5 import com.harvey.gestorbiblioteca.data.LibraryDatabase
6 import com.harvey.gestorbiblioteca.data.LibraryItem
7 import kotlinx.coroutines.launch
8
9 class LibraryViewModel(application: Application) : AndroidViewModel(application) {
10     private val dao = LibraryDatabase.getDatabase(application).libraryDao() // Obté l'objecte DAO de la base de dades
11     val allItems: LiveData<List<LibraryItem>> = dao.getAllItems() // LiveData amb tots els elements de la biblioteca
12
13     fun getItemById(itemId: Int): LiveData<LibraryItem> {
14         return dao.getItemById(itemId) // Obté un element per ID en temps real
15     }
16
17     fun insert(item: LibraryItem) = viewModelScope.launch {
18         dao.insertItem(item) // Insereix un nou element
19     }
20
21     fun update(item: LibraryItem) = viewModelScope.launch {
22         dao.updateItem(item) // Actualitza un element existent
23     }
24
25     fun delete(item: LibraryItem) = viewModelScope.launch {
26         dao.deleteItem(item) // Elimina un element existent
27     }
28 }
```

MainActivity.kt

Aquest codi defineix l'activitat principal (MainActivity) que configura la navegació entre fragments i mostra un diàleg amb informació del creador. Utilitza View Binding i MaterialAlertDialogBuilder per gestionar les vistes i els diàlegs.

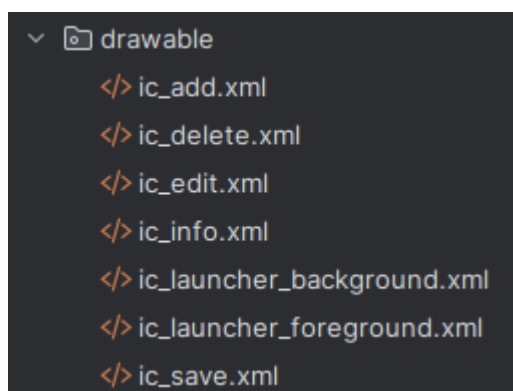
```

1 package com.harvey.gestorbiblioteca
2
3 > import ...
4
5
6
7
8
9
10 class MainActivity : AppCompatActivity() {
11     private lateinit var binding: ActivityMainBinding // Vinculació de vistes
12
13     override fun onCreate(savedInstanceState: Bundle?) {
14         super.onCreate(savedInstanceState)
15         binding = ActivityMainBinding.inflate(layoutInflater)
16         setContentView(binding.root) // Configura el contingut de la vista
17         // Recupera el NavHostFragment i obté el seu NavController.
18         val navHostFragment =
19             supportFragmentManager.findFragmentById(R.id.nav_host_fragment) as NavHostFragment
20         val navController = navHostFragment.navController
21         setupActionBarWithNavController(navController) // Configura la barra d'accions amb NavController
22
23         // Configura el botó per mostrar la informació del creador.
24         binding.btnCreatorInfo.setOnClickListener { showCreatorInfoDialog() }
25     }
26
27     private fun showCreatorInfoDialog() {
28         MaterialAlertDialogBuilder(context, this)
29             .setTitle("Fet per Harvey John Glover 2025")
30             .setMessage("Nom: Harvey John Glover\n" +
31                 "Correu: hglover@iesebre.com\n" +
32                 "Mòdul: MP08\n" +
33                 "Professor: Jordi Vega")
34             .setPositiveButton(text: "Tancan", listener: null)
35             .show() // Mostra un diàleg amb la informació del creador
36     }
37
38     override fun onSupportNavigateUp(): Boolean {
39         val navHostFragment =
40             supportFragmentManager.findFragmentById(R.id.nav_host_fragment) as NavHostFragment
41         val navController = navHostFragment.navController
42         return navController.navigateUp() || super.onSupportNavigateUp() // Gestiona la navegació cap amunt
43     }
44 }

```

DRAWABLE

Els diferents fitxers que tinc (add, delete, edit, info i save) defineixen diferents icones amb formes variades. Cada fitxer vectorial conté la definició gràfica d'una icona específica, i que la seva estructura XML és consistent en especificar les formes i colors per a la representació gràfica.



Exemple de fitxer (add)

```
</> ic_add.xml x
1  <vector xmlns:android="http://schemas.android.com/apk/res/android"
2      android:width="24dp"
3      android:height="24dp"
4      android:viewportWidth="960"
5      android:viewportHeight="960">
6      <path
7          android:pathData="M440,520L200,520v-80h240v-240h80v240h240v80L520,520v240h-80v-240Z"
8          android:fillColor="#FFFFFF"/>
9  </vector>
```

LAYOUT

activity_main.xml

Aquest codi XML defineix el disseny principal de la meua aplicació Android. Conté un botó que mostra la informació del creador i un FragmentContainerView que gestiona la navegació entre fragments amb NavHostFragment, utilitzant ConstraintLayout per a la disposició de les vistes.

```
<?xml version="1.0" encoding="utf-8" />
<androidx.constraintlayout.widget.ConstraintLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent">

    <Button
        android:id="@+id/btn_creator_info"
        android:layout_width="50dp"
        android:layout_height="50dp"
        android:layout_marginTop="25dp"
        android:background="#000000"
        android:backgroundTint="#000000"
        android:textColor="@android:color/white"
        app:icon="@drawable/ic_info"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintTop_toTopOf="@id/nav_host_fragment" />

    <androidx.fragment.app.FragmentContainerView
        android:id="@+id/nav_host_fragment"
        android:name="androidx.navigation.fragment.NavHostFragment"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        app:defaultNavHost="true"
        app:navGraph="@navigation/nav_graph"
        tools:layout_editor_absoluteX="0dp"
        tools:layout_editor_absoluteY="-16dp" />

</androidx.constraintlayout.widget.ConstraintLayout>
```

fragment_detail.xml

Aquest codi XML defineix una targeta (CardView) amb un disseny de contingut utilitzant ConstraintLayout. A la targeta, hi ha diversos components com TextViews per mostrar el títol, la descripció, l'autor, l'any i el gènere d'un element de la biblioteca. També inclou botons per editar i eliminar l'element, estilitzats amb icones i colors personalitzats.

```
</> fragment_detail.xml ×
1  <?xml version="1.0" encoding="utf-8"?>
2  <androidx.cardview.widget.CardView
3      xmlns:android="http://schemas.android.com/apk/res/android"
4      xmlns:app="http://schemas.android.com/apk/res-auto"
5      android:layout_width="match_parent"
6      android:layout_height="wrap_content"
7      app:cardElevation="8dp"
8      app:cardCornerRadius="12dp"
9      android:layout_marginRight="15dp"
10     android:layout_marginLeft="15dp"
11     android:layout_marginTop="40dp"
12     app:cardBackgroundColor="@color/white">
13
14     <androidx.constraintlayout.widget.ConstraintLayout
15         android:layout_width="match_parent"
16         android:layout_height="wrap_content"
17         android:padding="15dp">
18
19         <!-- Títol -->
20         <TextView
21             android:id="@+id/textViewTitle"
22             android:layout_width="0dp"
23             android:layout_height="wrap_content"
24             android:text="Títol"
25             android:textAppearance="?attr/textAppearanceHeadline6"
26             android:textColor="@color/black"
27             android:layout_marginBottom="10dp"
28             app:layout_constraintTop_toTopOf="parent"
29             app:layout_constraintStart_toStartOf="parent"
30             app:layout_constraintEnd_toEndOf="parent" />
```

```

<!-- Descripció -->
<TextView
    android:id="@+id/textViewDescription"
    android:layout_width="0dp"
    android:layout_height="wrap_content"
    android:text="Descripció"
    android:textAppearance="?attr/textAppearanceBody1"
    android:textColor="@color/black"
    app:layout_constraintTop_toBottomOf="@id/textViewTitle"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    android:layout_marginTop="8dp" />

<!-- Separador -->
<View
    android:id="@+id/separator"
    android:layout_width="match_parent"
    android:layout_height="1dp"
    android:background="#D3D3D3"
    app:layout_constraintTop_toBottomOf="@id/textViewDescription"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    android:layout_marginTop="8dp" />

<!-- Autor -->
<TextView
    android:id="@+id/textViewAuthor"
    android:layout_width="0dp"
    android:layout_height="wrap_content"
    android:text="Autor"
    android:textAppearance="?attr/textAppearanceSubtitle1"
    android:textColor="@color/black"
    app:layout_constraintTop_toBottomOf="@id/separator"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    android:layout_marginTop="15dp" />

```

```
<!-- Any -->
<TextView
    android:id="@+id/textViewYear"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Any"
    android:textAppearance="?attr/textAppearanceBody1"
    android:textColor="@color/black"
    app:layout_constraintTop_toBottomOf="@id/textViewAuthor"
    app:layout_constraintStart_toStartOf="parent"
    android:layout_marginTop="8dp" />

<!-- Gènere -->
<TextView
    android:id="@+id/textViewGenre"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Gènere"
    android:textAppearance="?attr/textAppearanceBody1"
    android:textColor="@color/black"
    app:layout_constraintTop_toBottomOf="@id/textViewYear"
    app:layout_constraintStart_toStartOf="parent"
    android:layout_marginTop="8dp" />
```



```
<!-- Botó d'editar -->
<com.google.android.material.button.MaterialButton
    android:id="@+id/buttonEdit"
    android:layout_width="0dp"
    android:layout_height="wrap_content"
    android:text="Editar"
    app:cornerRadius="8dp"
    app:icon="@drawable/ic_edit"
    app:iconGravity="textStart"
    app:iconPadding="8dp"
    app:layout_constraintTop_toBottomOf="@id/textViewGenre"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintEnd_toStartOf="@id/buttonDelete"
    android:layout_marginTop="10dp"
    android:layout_marginEnd="8dp"
    app:backgroundTint="@color/primaryColor"/>

<!-- Botó d'eliminar -->
<com.google.android.material.button.MaterialButton
    android:id="@+id/buttonDelete"
    android:layout_width="0dp"
    android:layout_height="wrap_content"
    android:text="Eliminar"
    app:cornerRadius="8dp"
    app:icon="@drawable/ic_delete"
    app:iconGravity="textStart"
    app:iconPadding="8dp"
    app:layout_constraintTop_toBottomOf="@id/textViewGenre"
    app:layout_constraintStart_toEndOf="@id/buttonEdit"
    app:layout_constraintEnd_toEndOf="parent"
    android:layout_marginTop="10dp"
    android:layout_marginStart="8dp"
    app:backgroundTint="@color/errorColor"/>

</androidx.constraintlayout.widget.ConstraintLayout>
</androidx.cardview.widget.CardView>
```

fragment_edit.xml

Aquest codi XML defineix una vista de desplaçament (ScrollView) que conté un layout de restricció (ConstraintLayout). A dins, hi ha diversos camps d'edició de text (EditText) per introduir el títol, l'autor, la descripció, l'any i el gènere d'un element de la biblioteca. També inclou un botó per guardar la informació introduïda, estilitzat amb una icona i un color de fons personalitzat.

```
</> fragment_edit.xml ×
1  <?xml version="1.0" encoding="utf-8"?>
2  <ScrollView
3      xmlns:android="http://schemas.android.com/apk/res/android"
4      xmlns:app="http://schemas.android.com/apk/res-auto"
5      android:layout_width="match_parent"
6      android:layout_height="match_parent"
7      android:fillViewport="true"
8      android:background="@color/background_color">
9
10     <androidx.constraintlayout.widget.ConstraintLayout
11         android:layout_width="match_parent"
12         android:layout_height="wrap_content"
13         android:padding="16dp"
14         android:layout_marginRight="10dp"
15         android:layout_marginLeft="10dp"
16         android:layout_marginTop="40dp"
17         android:layout_marginBottom="40dp">
18
19         <!-- Títol -->
20         <EditText
21             android:id="@+id/editTextTitle"
22             android:layout_width="0dp"
23             android:layout_height="wrap_content"
24             android:hint="Títol"
25             android:inputType="textPersonName"
26             android:textSize="18sp"
27             android:padding="12dp"
28             app:layout_constraintTop_toTopOf="parent"
29             app:layout_constraintStart_toStartOf="parent"
30             app:layout_constraintEnd_toEndOf="parent" />
```

```
<!-- Autor -->
<EditText
    android:id="@+id/editTextAuthor"
    android:layout_width="0dp"
    android:layout_height="wrap_content"
    android:hint="Autor"
    android:inputType="textPersonName"
    android:textSize="18sp"
    android:padding="12dp"
    app:layout_constraintTop_toBottomOf="@id/editTextTitle"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    android:layout_marginTop="16dp"/>

<!-- Descripció -->
<EditText
    android:id="@+id/editTextDescription"
    android:layout_width="0dp"
    android:layout_height="wrap_content"
    android:hint="Descripció"
    android:inputType="textMultiLine"
    android:textSize="18sp"
    android:padding="12dp"
    app:layout_constraintTop_toBottomOf="@id/editTextAuthor"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    android:layout_marginTop="16dp"/>
```

```
<!-- Any -->
<EditText
    android:id="@+id/editTextYear"
    android:layout_width="0dp"
    android:layout_height="wrap_content"
    android:hint="Any"
    android:inputType="number"
    android:textSize="18sp"
    android:padding="12dp"
    app:layout_constraintTop_toBottomOf="@id/editTextDescription"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    android:layout_marginTop="16dp"/>

<!-- Gènere -->
<EditText
    android:id="@+id/editTextGenre"
    android:layout_width="0dp"
    android:layout_height="wrap_content"
    android:hint="Gènere"
    android:inputType="text"
    android:textSize="18sp"
    android:padding="12dp"
    app:layout_constraintTop_toBottomOf="@id/editTextYear"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    android:layout_marginTop="16dp"/>
```

```
<!-- Botó de guardar -->
<com.google.android.material.button.MaterialButton
    android:id="@+id/buttonSave"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Guardar"
    app:cornerRadius="8dp"
    app:icon="@drawable/ic_save"
    app:iconGravity="textStart"
    app:iconPadding="8dp"
    app:layout_constraintTop_toBottomOf="@id/editTextGenre"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    android:layout_marginTop="24dp"
    android:paddingLeft="24dp"
    android:paddingRight="24dp"
    app:backgroundTint="@color/primaryColor" />

</androidx.constraintlayout.widget.ConstraintLayout>
</ScrollView>
```

fragment_main.xml

Aquest codi XML defineix un ConstraintLayout que conté una RecyclerView per mostrar una llista d'elements i un FloatingActionButton per afegir nous elements. La RecyclerView s'adapta a la mida del contenidor i el botó flotant està posicionat a la part inferior dreta de la pantalla.

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:layout_marginRight="15dp"
    android:layout_marginLeft="15dp"
    android:layout_marginTop="45dp"
    android:layout_marginBottom="45dp">

    <androidx.recyclerview.widget.RecyclerView
        android:id="@+id/recyclerView"
        android:layout_width="0dp"
        android:layout_height="0dp"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

    <com.google.android.material.floatingactionbutton.FloatingActionButton
        android:id="@+id/fabAdd"
        android:layout_width="56dp"
        android:layout_height="56dp"
        android:layout_margin="20dp"
        android:layout_marginEnd="16dp"
        android:layout_marginBottom="128dp"
        android:contentDescription="Afegir element"
        app:backgroundTint="@color/primaryColor"
        app:elevation="8dp"
        app:fabSize="normal"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:srcCompat="@drawable/ic_add" />

</androidx.constraintlayout.widget.ConstraintLayout>
```

item_library.xml

Aquest codi XML defineix una CardView que conté un layout de restricció (ConstraintLayout). Dins la targeta, hi ha dos TextView per mostrar el títol i l'autor d'un element de la biblioteca. La targeta té una elevació i unes vores arrodonides, així com un color de fons personalitzat.

```
<?xml version="1.0" encoding="utf-8" />
<androidx.cardview.widget.CardView
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    app:cardElevation="8dp"
    app:cardCornerRadius="12dp"
    android:layout_marginBottom="20dp"
    app:cardBackgroundColor="@color/list_item_background">

    <androidx.constraintlayout.widget.ConstraintLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:padding="15dp"
        android:layout_marginBottom="10dp">

        <TextView
            android:id="@+id/textViewTitle"
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:text="Títol"
            android:textAppearance="?attr/textAppearanceHeadline6"
            android:textColor="@color/list_item_text"
            app:layout_constraintTop_toTopOf="parent"
            app:layout_constraintStart_toStartOf="parent"
            app:layout_constraintEnd_toEndOf="parent" />

        <TextView
            android:id="@+id/textViewAuthor"
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:text="Autor"
            android:textAppearance="?attr/textAppearanceSubtitle1"
            android:textColor="@color/secondaryTextColor"
            app:layout_constraintTop_toBottomOf="@id/textViewTitle"
            app:layout_constraintStart_toStartOf="parent"
            app:layout_constraintEnd_toEndOf="parent"
            android:layout_marginTop="8dp" />

    </androidx.constraintlayout.widget.ConstraintLayout>
</androidx.cardview.widget.CardView>
```

VALUES

colors.xml

Aquest codi XML defineix una sèrie de colors utilitzats en la meua aplicació Android. Els colors inclouen diferents tons de morat, turquesa, negre, blanc, i altres colors personalitzats per a elements específics com el color primari, secundari, de fons, de text principal i secundari, entre d'altres.

```
</> colors.xml x
1  <?xml version="1.0" encoding="utf-8"?>
2  <resources>
3      <color name="purple_200">#FFBB86FC</color>
4      <color name="purple_500">#FF6200EE</color>
5      <color name="purple_700">#FF3700B3</color>
6      <color name="teal_200">#FF03DAC5</color>
7      <color name="teal_700">#FF018786</color>
8      <color name="black">#FF000000</color>
9      <color name="white">#FFFFFFFF</color>
10     <color name="primaryColor">#6200EE</color>
11     <color name="secondaryColor">#03DAC6</color>
12     <color name="errorColor">#B00020</color>
13     <color name="backgroundColor">#F5F5F5</color>
14     <color name="primaryTextColor">#212121</color>
15     <color name="secondaryTextColor">#757575</color>
16     <color name="list_item_background">#e8def7</color>
17     <color name="list_item_text">#000000</color>
18 </resources>
```


strings.xml

Aquest codi XML defineix dues cadenes de text (strings) utilitzades en la meua aplicació Android. La cadena `app_name` estableix el nom de l'aplicació com "GestorBiblioteca" i la cadena `add_item` proporciona el text per afegir un nou element.

```
</> strings.xml ×  
1  <resources>  
2      <string name="app_name">GestorBiblioteca</string>  
3      <string name="add_item">Afegir element</string>  
4  </resources>
```

themes.xml

Aquest codi XML defineix un tema per a la meua aplicació Android anomenat "Theme.GestorBiblioteca". Hereta de "Theme.MaterialComponents.Light.DarkActionBar" i especifica els colors primaris, primari fosc i d'accent utilitzant els recursos de color definits en el fitxer de colors.

```
</> themes.xml ×  
1  <?xml version="1.0" encoding="utf-8"?>  
2  <resources>  
3      <style name="Theme.GestorBiblioteca" parent="Theme.MaterialComponents.Light.DarkActionBar">  
4          <item name="colorPrimary">@color/purple_500</item>  
5          <item name="colorPrimaryDark">@color/purple_700</item>  
6          <item name="colorAccent">@color/teal_200</item>  
7      </style>  
8  </resources>
```

nav_graph.xml

Aquest codi XML defineix la navegació entre fragments en una aplicació Android utilitzant un gràfic de navegació (navigation). Configura els diferents fragments (MainFragment, DetailFragment, EditFragment) i les accions de navegació entre ells, incloent arguments per passar dades entre fragments.

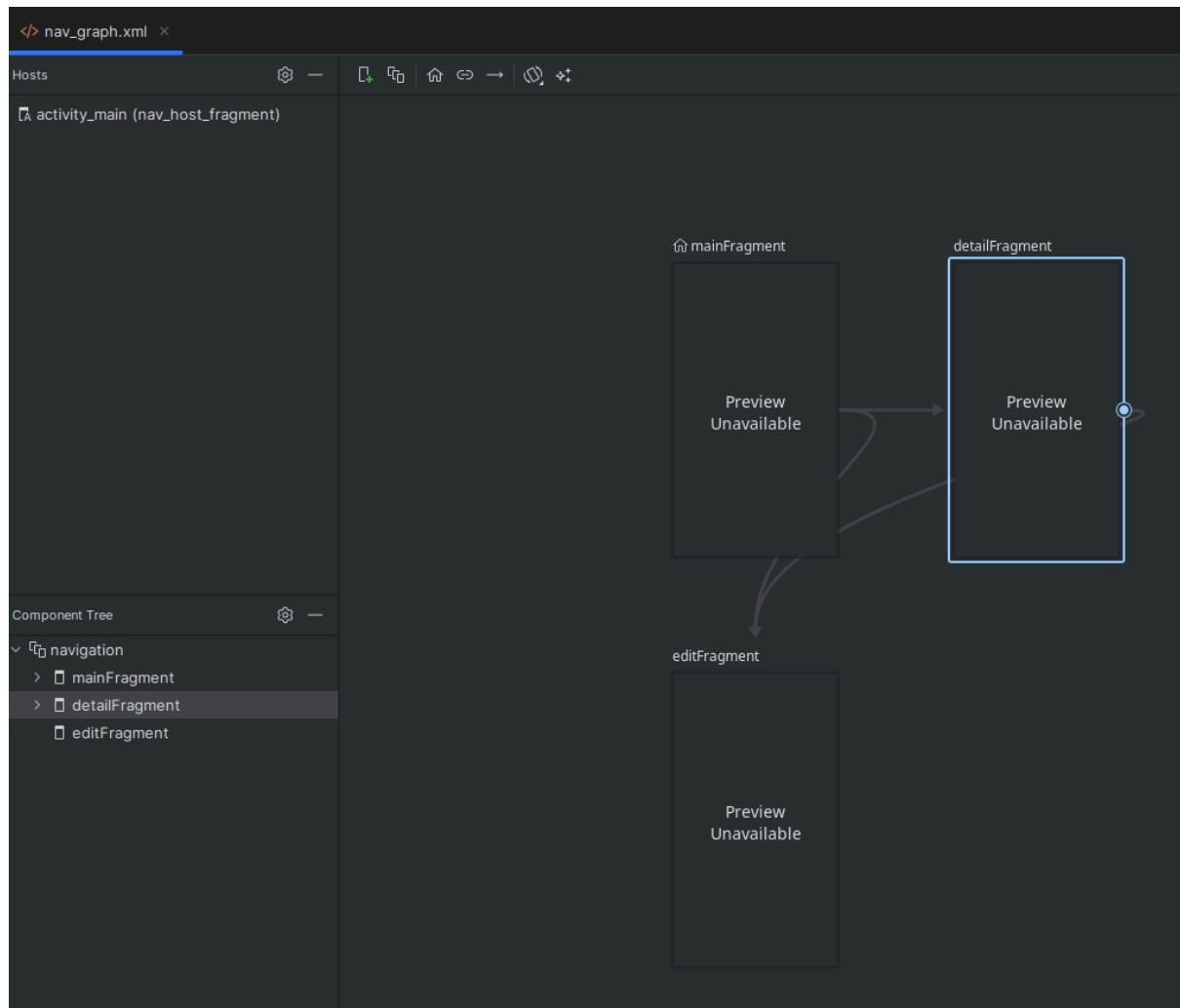
```
<?xml version="1.0" encoding="utf-8"?>
<navigation
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    app:startDestination="@id/mainFragment">

    <fragment
        android:id="@+id/mainFragment"
        android:name="com.harvey.gestorbiblioteca.ui.MainFragment"
        android:label="Llista de llibres">
        <action
            android:id="@+id/action_mainFragment_to_detailFragment"
            app:destination="@id/detailFragment" />
        <action
            android:id="@+id/action_mainFragment_to_editFragment"
            app:destination="@id/editFragment" />
    </fragment>

    <fragment
        android:id="@+id/detailFragment"
        android:name="com.harvey.gestorbiblioteca.ui.DetailFragment"
        android:label="Detall del llibre">
        <argument
            android:name="itemId"
            app:argType="integer" />
        <action
            android:id="@+id/action_detailFragment_to_editFragment"
            app:destination="@id/editFragment" />
    </fragment>

    <fragment
        android:id="@+id/editFragment"
        android:name="com.harvey.gestorbiblioteca.ui.EditFragment"
        android:label="Editar/Crear llibre">
        <argument
            android:name="itemId"
            app:argType="integer"
            android:defaultValue="-1" />
    </fragment>
</navigation>
```

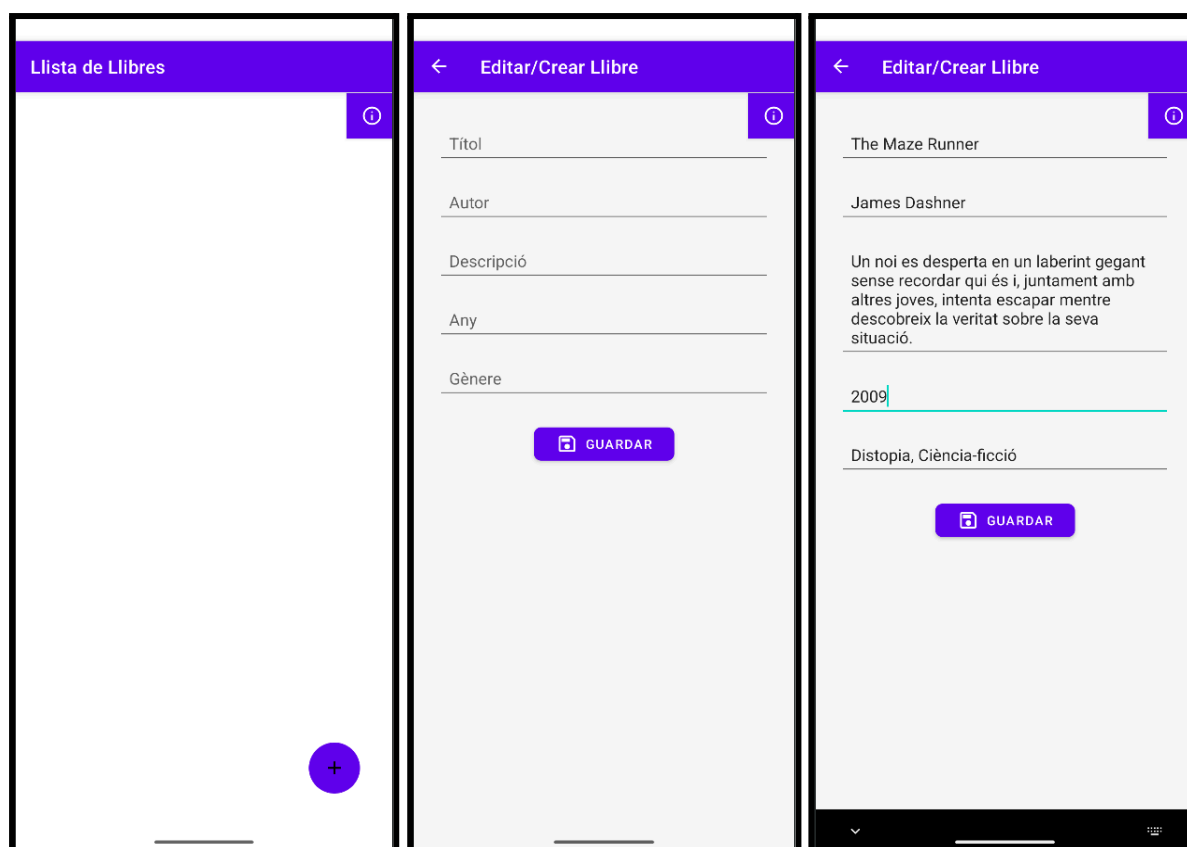
Vista Fragments



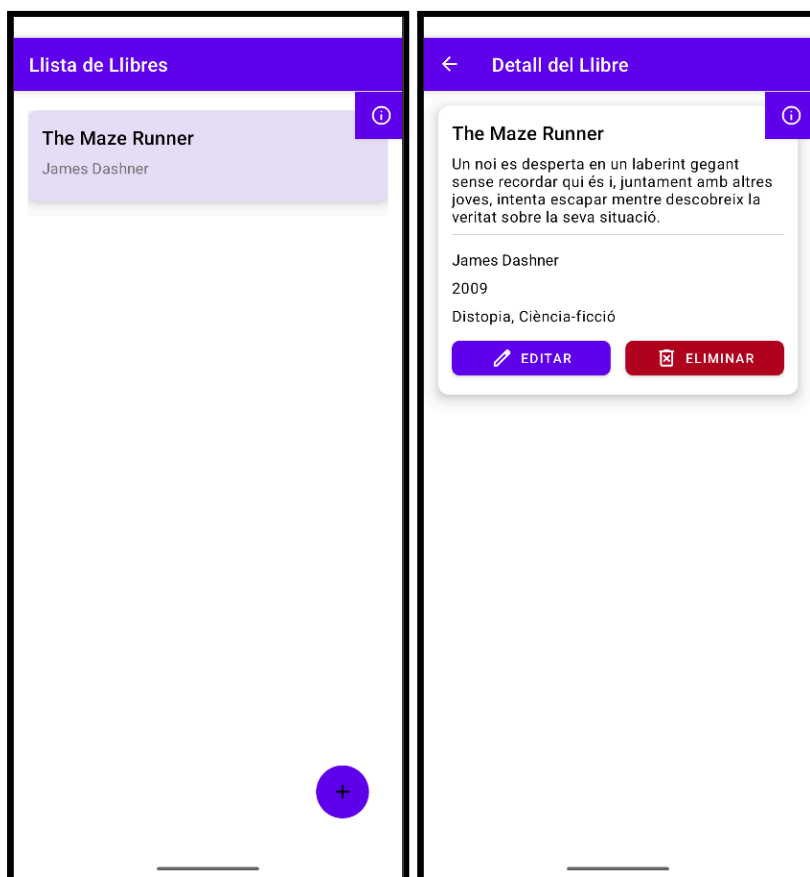
Proves i comprovacions

Captures del funcionament del CRUD

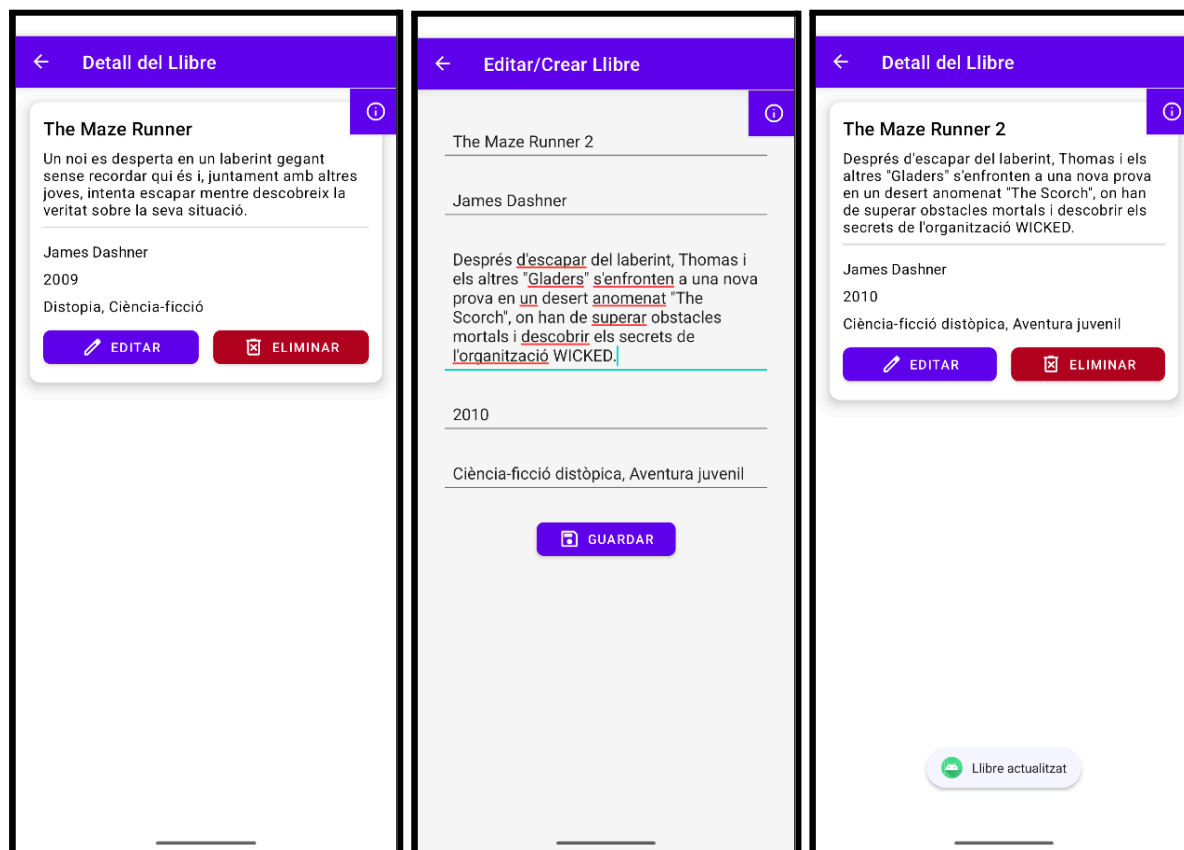
CREATE: Començare obrint l'app, apareix el menú per omplir les opcions i una vegada plenades, clico el botó de guardar.



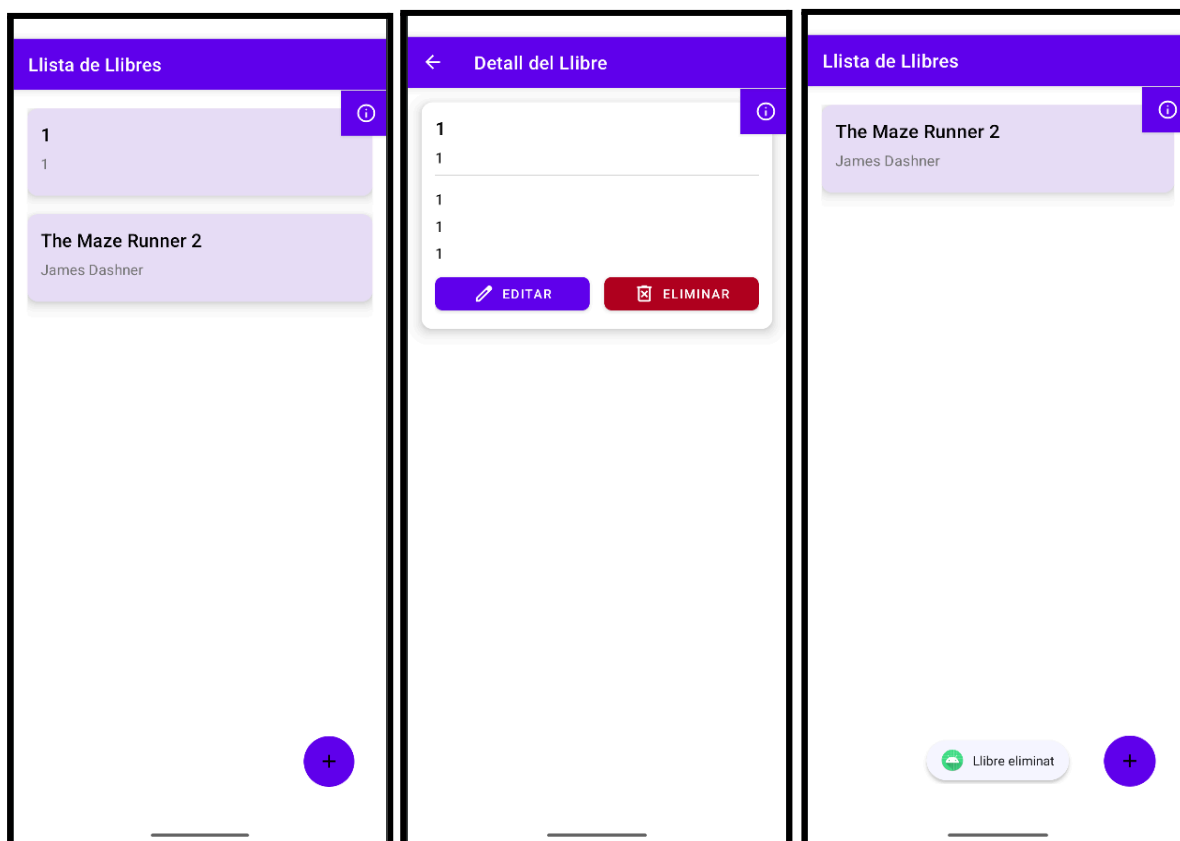
READ: Una vegada creat em mostra el llibre a la llista principal, clico sobre el llibre, i em mostra els detalls amb l'opció de editar o eliminar.




EDIT: Una vegada creat em mostra l'opció de editar, que si pulso sobre podre canviar valors del llibre. Una vegada canviats, guardo i m'apareix un missatge de Llibre actualitzat, veure que s'han canviat.

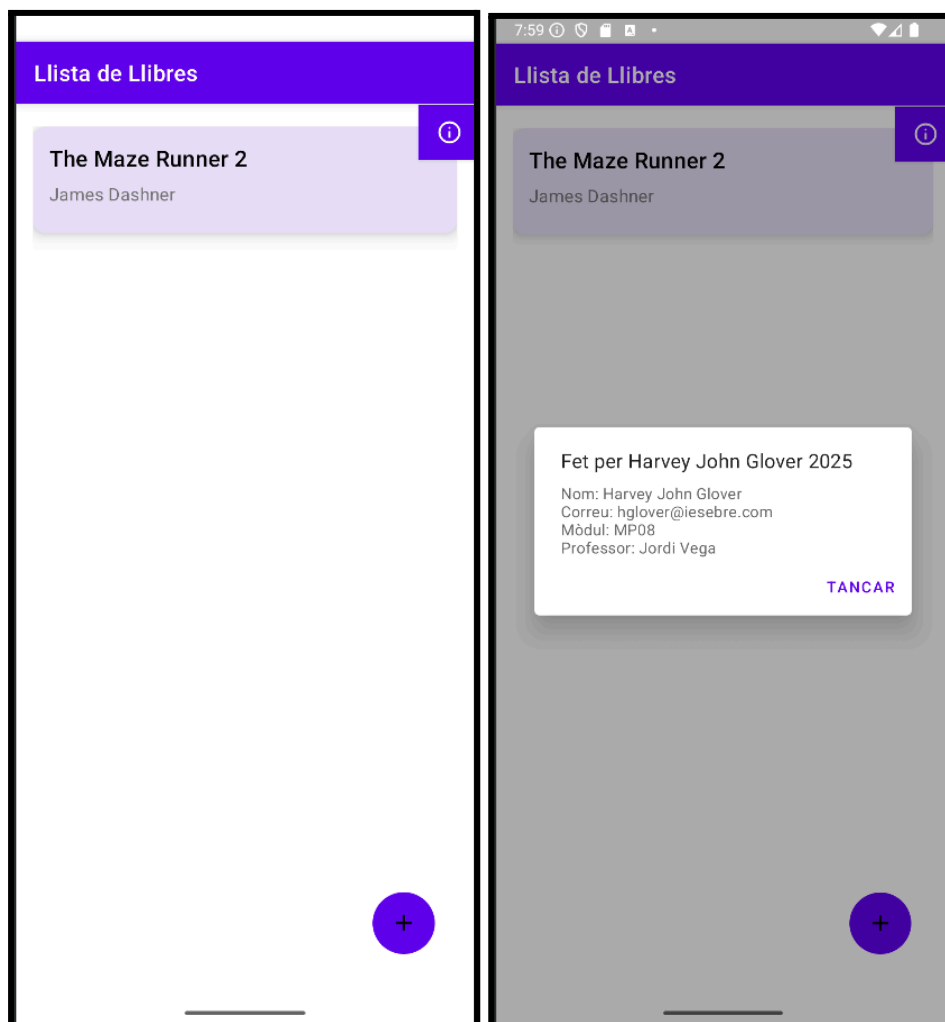


DELETE: Per eliminar, primer creo un llibre de prova anomenat "1". Quan entro als detalls m'apareix un botor d'eliminar, el clico i quan surto al menú veure que desapareix.



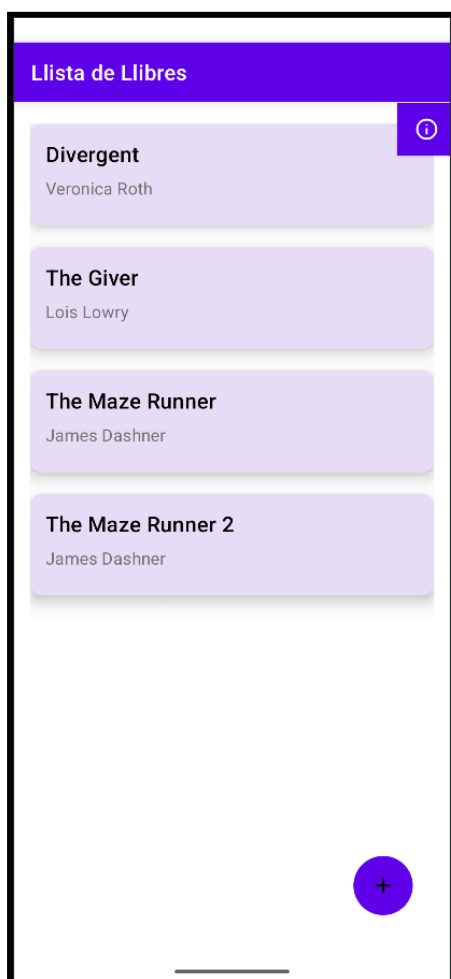
 INSTITUT DE L'EBRE TORTOSA	M8 Programació multimèdia i dispositius mòbils	DAM 2n
--	--	------------------

INFO: En totes les pàgines hi ha un icono dalt a la dreta que serveix per tenir info extra de l'app. Si clico sobre la icona, veure que em mostra un popup amb la info.



Verificació amb Database Inspector d'Android Studio

Pleno la llista amb 4 llibres com a exemples, i si miro el Inspector, veure que apareixen els 4 amb tota la info que tenen.



App Inspection

Medium Phone API 35 > com.harvey.gestorbiblioteca

Database Inspector | Network Inspector | Background Task Inspector

Databases: library_items

Live updates: ☐

	id	title	description	author	year	genre
1	30	The Maze Runner 2	Després d'escap	James Dashner	2010	Ciència-ficció
2	32	Divergent	En una societat c	Veronica Roth	2011	Distopia, Cièn
3	33	The Giver	En una societat e	Lois Lowry	1993	Distopia, Cièn
4	34	The Maze Runner	Un noi es desper	James Dashner	2009	Distopia, Cièn