Android Dev

Sai Kalyana Raman(U24553100)
Shatrugna Rao Korukanti(U43517028)

Overview

Our project is to build an utility android app. It is similar to the present day Notion app. We aim to provide tools in assisting document generation and also integrate with google drive storage medium.

Related Work

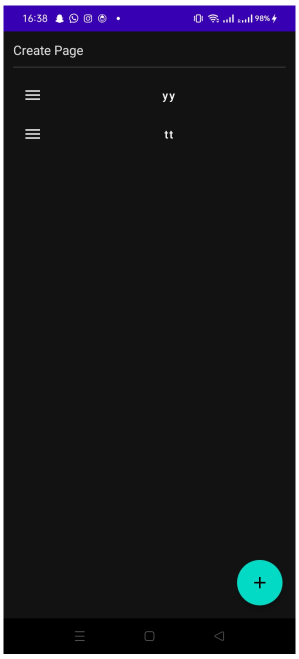
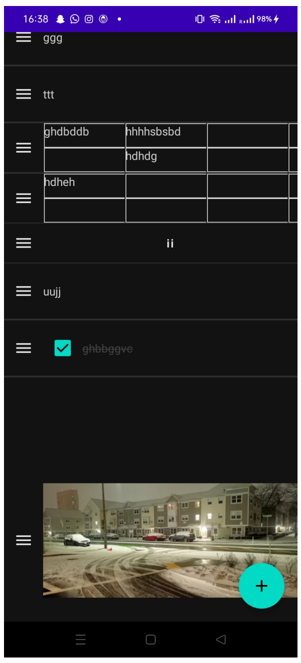
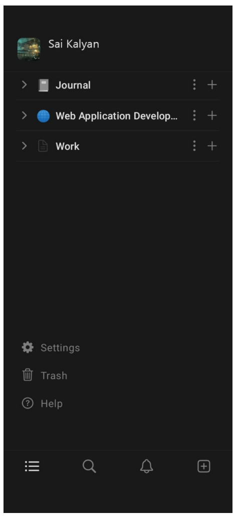
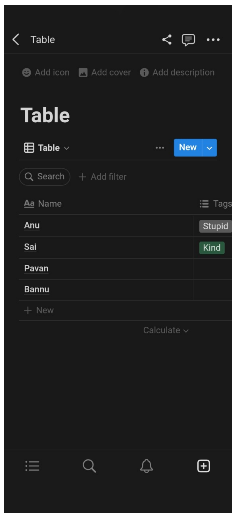
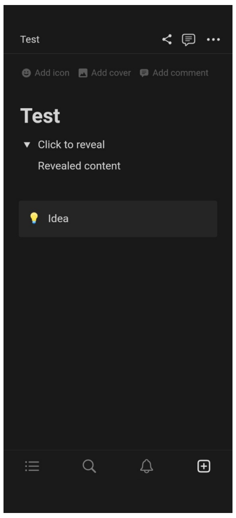
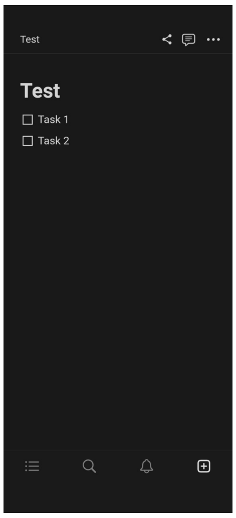
It is simlar to Notion app, with dynamic creation of components over an infinite long screen. It mimics the operation of writing a diary and storing notes of a user. There could be a downgrade in the visual elements due to material design constraint in android jetpack compose.

GitHub Link

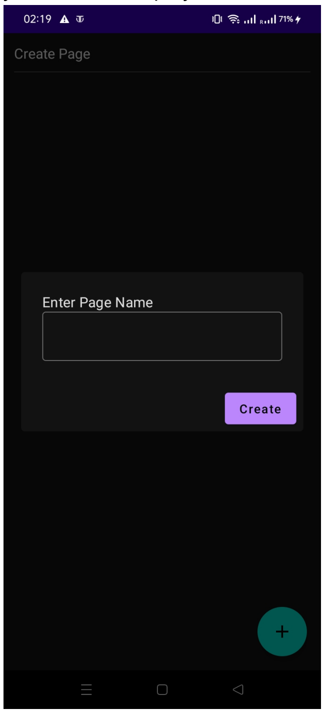
https://github.com/CS683/project-androiddev

Requirement Analysis and Testing

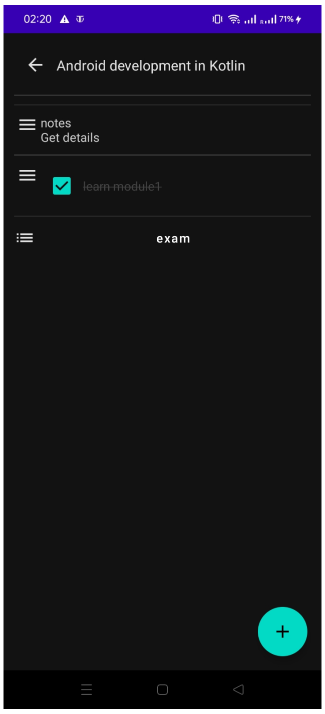
Title	Project details (Achieved)	Project details (Desired)
Description	Our project is to build an utility android app similar to Notion app. We aim to provide tools in assisting document generation and also integrate with google drive storage medium.	Additionally, we planned to integrate backup using google drive api.

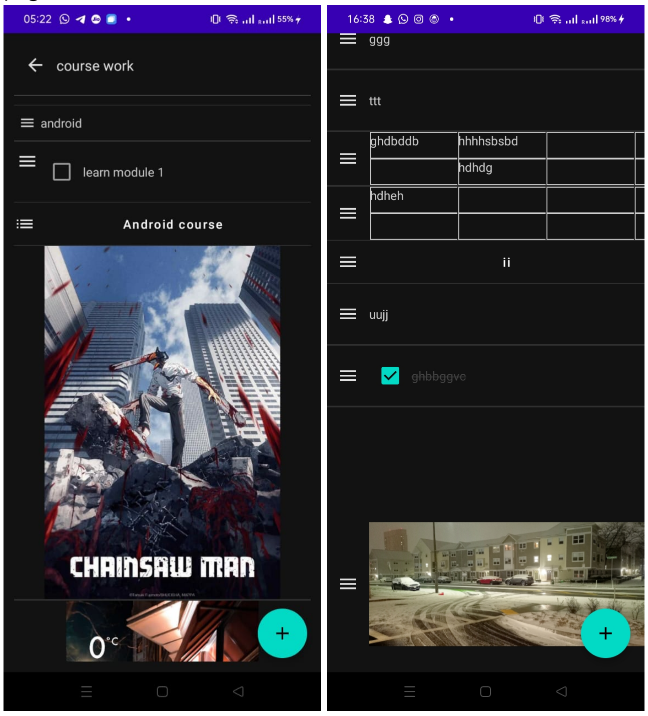
Title	Project details (Achieved)	Project details (Desired)
UI	**sample_main**  **sample_page**  ggg ttt ghdbddb hhhhsbsbd hdhdg hdheh ii uujj ~~ghbbggve~~	Mockup that was expected to be built **sample_main**  Sai Kalyan Journal Web Application Develop... Work Settings Trash Help **sample_page**  Table Add icon Add cover Add description Table Table New Search Add filter Name Tags Anu Stupid Sai Kind Pavan Bannu New Calculate  Test Add icon Add cover Add comment Test Click to reveal Revealed content Idea 
Acceptance Tests	User testing all the implemented features described in the report.	None
Test Results	All the implemented are working with no issue. Only the image component takes a lot of time to add image to UI screen because of the unoptimised code of resizing huge images.	None

Design and Implementation

- UI design and implementation
  - Jetpack Compose UI elements.
    - Jetpack Compose is a modern UI toolkit recently launched by Google which is used for building native Android UI.
    - Declarative: It is fully declarative so that you can describe your UI components by calling some predefined functions.
    - All the UI and states are declared inside composable functions.
    - The most basic UI elements are the Row, Column which are used for layout, since everything in the jetpack compose can be visually viewed as grid.
    - The UI and logic are entirely written in kotlin which allows the user to not create unnecessary XML UI files.
    - Drawbacks include inflexible ui styling, and in this project the ui cannot be viewed until the code is compiled and run on an emulator or device.
  - Composable functions can have Simple UI code like Row{Text(modifier = Modifier.padding(3.dp)){"Click"}} which replaces predefined static xml design.
  - The navigation component is handled using NavHost element in which you can declare path and call @composable function you want to display over the screen.
  - 

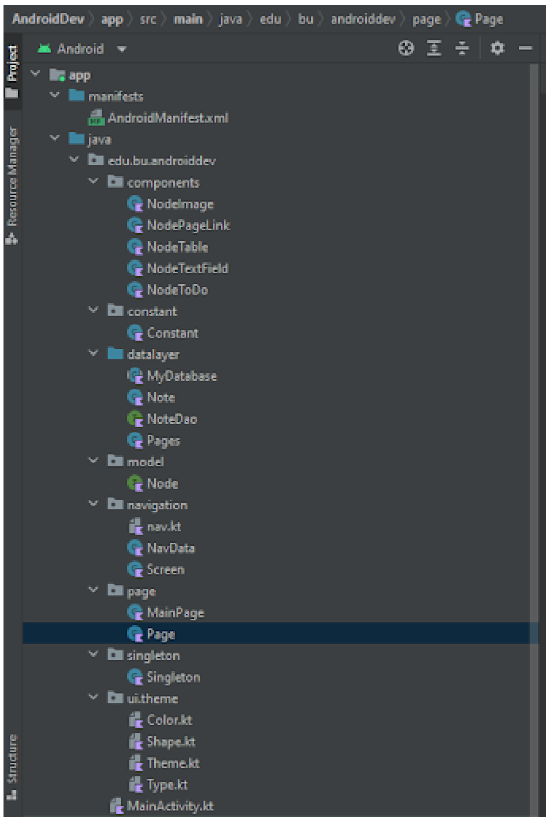
- Code Structure
  - There is a node interface that is implemented to all the components classes that are created.
  - The node interface list is created to put all the components in a single list for easy accessibility.
  - The code has multiple packages with proper name to define the purpose of it in the project.
  - The components are broken down into simple text data for it to be stored in db and can also be easily interpreted while dynamically generating the components UI.
- Activities
  - There are 2 main jetpack UI pages MainPage, Page.
  - MainPage contains link to navigate to other pages.

- 

- The Page can contain multiple components like textField, toDo checkbox, image component, table and link to navigate other pages.
- 

* Third party APIs
  - usage of Jackson Core for json serialization and deserialization.
* Database schema, data storage
  - The data of the pages and info were initially stored in the data store shared preferences.
  - The data storage have been moved to room.
    - There are 2 tables, one which contains the pages and the other table contains all the components that have been created in all the pages.
    - The data of components for a page is retrieved using a query to get the components that match to the selected page id.

Project Structure



```
AndroidDev 〉 app 〉 src 〉 main 〉 java 〉 edu 〉 bu 〉 androiddev 〉 page 〉 Page

app
  manifests
    AndroidManifest.xml
  java
    edu.bu.androiddev
      components
        NodeImage
        NodePageLink
        NodeTable
        NodeTextField
        NodeToDo
      constant
        Constant
      datalayer
        MyDatabase
        Note
        NoteDao
        Pages
      model
        Node
      navigation
        nav.kt
        NavData
        Screen
      page
        MainPage
        Page
      singleton
        Singleton
      ui.theme
        Color.kt
        Shape.kt
        Theme.kt
        Type.kt
      MainActivity.kt
```

Packages	Description	Files
Components	contains kotlin classes with composable render function and state variables. They implement Node interface. It handles deserialization and serialization of room DB	- NodeImage - responsible for adding image - NodePageLink - responsible for redirectingNodeTable - responsible for table - NodeTextField - responsible for basictextfield - NodeToDo - responsible for checkbox textfield
Constant	contain constants variables	- constant - final static variables
Datalayer	contains code for instantiation of Room DB and related classes	- MyDatabase - initiates room db object - Note - Data model class to store pages - Pages - Data model class to store components data - NoteDao - A Dao layer which contains operations for both Note and Pages data models.
Model	contain model classes or interfaces	- Node - Interface for the components. Contains functions for serialization and deserialization.
Navigation	contains navigation codes	- Nav - initialises navcontroller - navData - stores info of the url path
Page	contains ui which spans across the screen and are mapped to the navigation	- MainPage - First viewable screen to contain only redirect component - Page - composable function which can hold all the components.

Timeline

Iteration	Application Requirements			Android Components and Features	member 1 contribution/tasks(Sai)	member 2 contribution/tasks(Shatrugna)
	Essential	**Desirable**	**Optional**			
1	App Layout UI tools development	Complex UI development such as table and kanban board	Hyperlinks and image ui integration.	Jetpack Compose	Design and implement app layout, Functional Implementation of components	Navigation implementation, data storage of the UI components. Testing of the app and performing minor fixes.
	Essential	**Desirable**	**Optional**			
2	Serialization of UI data and integrating gdrive api.	Integrating voice and video chat using webrtc protocol.	serialization of data in the form of markdown markup language.	Google Drive Kotlin APIs'	Room database api implementation, moving from shared preferences data store to room.	Image component and integration of room api with current design of the application.
	Essential	**Desirable**	**Optional**			
3	Table Component, delete functionality and bug fixes.	Google drive api integration.	Dragging and repositioning of UI components.	Fixing issues and improving performance	Bug Fixes. Saving data on navigating between pages and image size constraint fix and also UI fix.	Table Component added and recursive page links fix. Delete functionality added to components.

Future Scope

- Would like to improve on UI
- Drag capability
- Firebase Integration
- Notification System
- Complex components like kanban board

References

Notion app