NoteMark Milestone #2 Requirements

This requirements document explains **NoteMark's Milestone #2** requirements. For each milestone, a new requirements document and an updated Figma file will be created. You can find all current milestones in the members area.

The mockups show the exact look and colors of a specific UI element. The app has a single theme, but no light or dark theme.

Note that this serves to give you an overall impression of what the app should be able to do. Feel free to decide how you implement specific things (e.g., how you display a specific loading progress or how you specify error messages).

You can find the mockups for **NoteMark** here: https://www.figma.com/design/3eLOJSbYh6HVwIYVcqR5YE/NoteMark?node-id=6239-4099

Milestone #2 Goal

Let the Notes Begin: Make it easy, breezy, and beautiful to jot down a thought.

In this milestone, we bring NoteMark to life by introducing the core note-taking experience. The focus is on designing a scrollable, visually engaging Note List Screen with a staggered grid layout that feels fresh yet familiar. The content will also auto-save as they type.

Icons

All icons for the app can be Material design icons or taken from the mockups as SVG (in case an equivalent Material icon doesn’t exist)

Adaptive Layouts

All screens must look good and work well on phones and tablets, in both portrait and landscape. All UI requirements apply to each device, and unique differences will be explicitly mentioned.

Technical Requirements

▼ API to manage notes on server

- Please find a detailed API document for working with notes in the attachments section

▼ Offline-first behaviour

With offline-first apps, the source of truth shall always be the local database. It is important to keep your remote data source (API) in sync with the local database. Otherwise, the user can't rely on their data being safely stored remotely.

- All notes created, edited, and deleted should be saved locally (database) and remotely (API)
- Complete sync logic is not required for this milestone. You only need to update the database and make an API call each time a note is created, updated, or deleted.

> 💡 Make sure to make use of a CoroutineScope scoped to the Application's lifecycle when making the API calls to prevent them from being cancelled when the user navigates away from the current screen.

Sample code: Create Note

```kotlin
override suspend fun createNote(note: NoteItem): EmptyResult<DataError> {
  val localResult = localDataSource.upsertNoteItem(note)
  /*
    We may encounter an error (like no space on device!)
    In that case, creating the note failed and we notify the user
  */
  if (localResult !is Result.Success) {
    return localResult
  }

  /*
    We've saved the note to the local database.
    Let's protect the following code from being cancelled when the user
    navigates away from the current screen
  */
  return applicationScope.async {
    // Try to make an API call to create a note
    val remoteResult = remoteDataSource.postNote(note)
    if (remoteResult is Result.Error) {
      // Next milestone we'll handle what to do when the call fails 🙂
      return@async Result.Success(Unit)
    }

    return@async remoteResult
  }.await()
}
```

▼ Note List Screen

- Top app bar with app name far left
- Display a profile icon on the right of the toolbar using the user's initials:
  - The initials are derived from the username of the account
  - Single-word name: use first two characters (e.g. Philipp → **PH**)
  - Two word name: use first and last word initials (e.g. Philipp Lackner → **PL**)
  - More than two words name: use first and last word initials (e.g. First Middle Last → **FL**)
- Main content of screen is a staggered grid of notes
  - Placeholder text is shown when no notes exist
  - Two (2) columns when in portrait and tablet mode and three (3) columns when in landscape mode

    - Tap note to navigate to the *Note Detail Screen*
    - Long tap note to show a standard Material 3 confirmation dialog.
        - Dialog title: “Delete Note?”
        - Dialog body text: “Are you sure you want to delete this note? This action cannot be undone.”
        - Include two buttons: “Delete” and “Cancel”
        - Only delete note when the user taps the “Delete” button
        - When the user dismisses or taps the “Cancel” button, do not delete the note
    - Each note in the list consists of
        - The date created
            - If the note was not created this year, include the year also
            - “19 Apr” when created this year
            - “19 Apr 2024” when created last year
        - Title
        - Content preview is ellipsized if it is too long to fit (text is cut of and ends in 3 dots “…”)
        - Place a limit on the number of characters allowed in the preview text
            - Phone: 150 characters (including spaces)
            - Tablet: 250 characters (including spaces)
    - Each note in the list should take up as much space as the content needs. If the note content exceeds the character limit (150 for phones and 250 for tablets) then it should be truncated with an ellipsis.

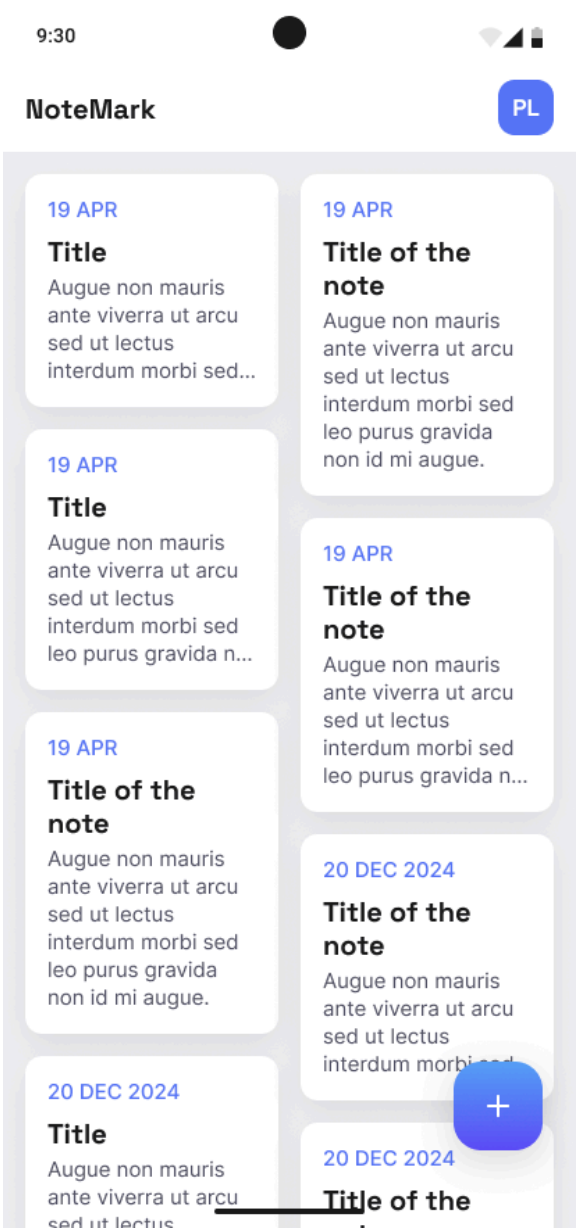
> 💡 This kind of layout can be achieved by using the `LazyVerticalStaggeredGrid` composable function.

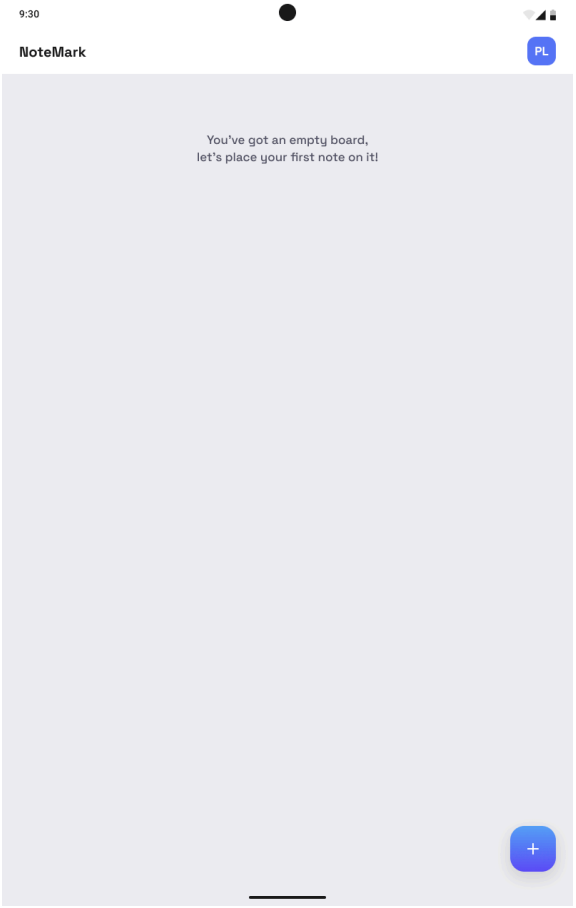
- FAB bottom right of screen

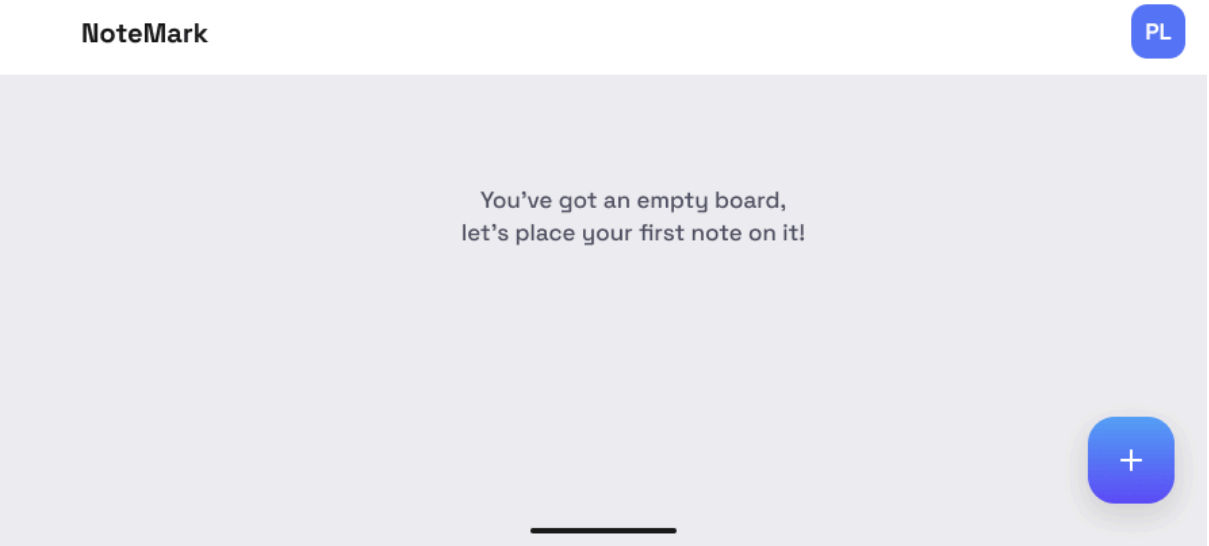
- Gradient background
- On tap
  - Create and save a new note to the database (don't forgot to call the API as well 🙂)
    - This new note will should have title "New Note" and its content value should be an empty string
  - Navigate to the Note Detail Screen with the newly created note

> **❓ Why create a new note immediately?**
>
> This approach prevents the need for a nullable ID. The Note Detail screen can always load an existing note using its ID. This makes the screen simpler because it doesn't have to check if it's "new" or "edit" — it always works the same way.

> **❓ What happens when the user cancels?**
>
> If the user opens a new note but does not type anything and goes back:
>
> - We can *check if the note is empty.*
> - If it's empty, we can *delete* it so that no blank notes stay in the list.
>
> This way, users don't see empty notes they didn't finish.
>
> 📝 ***Note that this is NOT a requirement to delete blank notes***


9:30
NoteMark
PL
19 APR
Title
Augue non mauris ante viverra ut arcu sed ut lectus interdum morbi sed...
19 APR
Title
Augue non mauris ante viverra ut arcu sed ut lectus interdum morbi sed leo purus gravida n...
19 APR
Title of the note
Augue non mauris ante viverra ut arcu sed ut lectus interdum morbi sed leo purus gravida non id mi augue.
20 DEC 2024
Title
Augue non mauris ante viverra ut arcu
19 APR
Title of the note
Augue non mauris ante viverra ut arcu sed ut lectus interdum morbi sed leo purus gravida non id mi augue.
19 APR
Title of the note
Augue non mauris ante viverra ut arcu sed ut lectus interdum morbi sed leo purus gravida n...
20 DEC 2024
Title of the note
Augue non mauris ante viverra ut arcu sed ut lectus
20 DEC 2024
Title of the

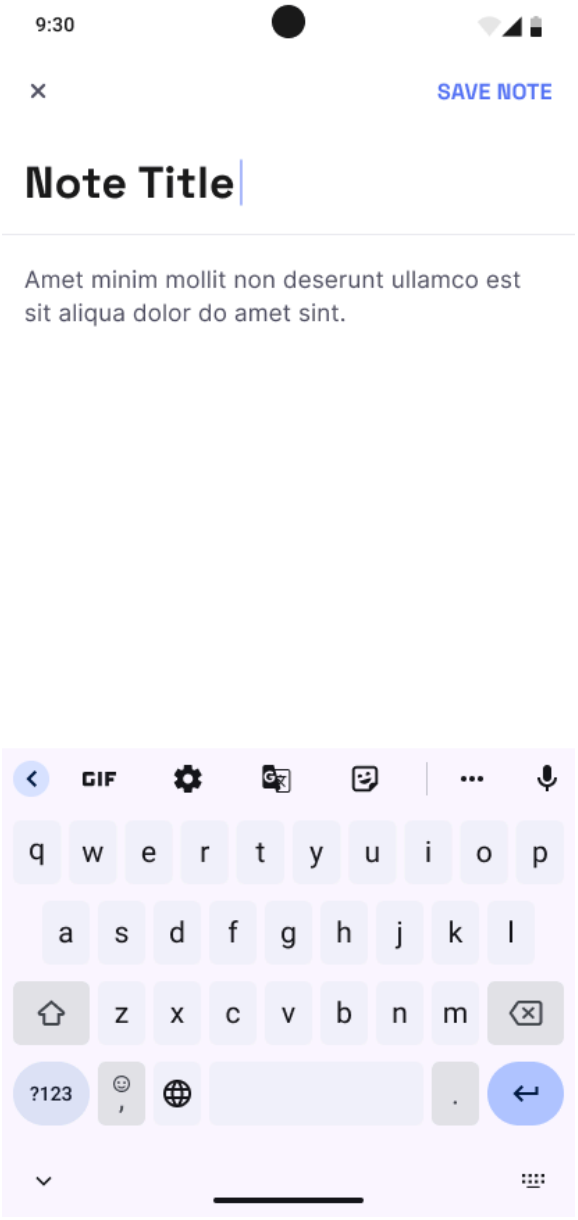

9:30
NoteMark
PL
You've got an empty board,
let's place your first note on it!

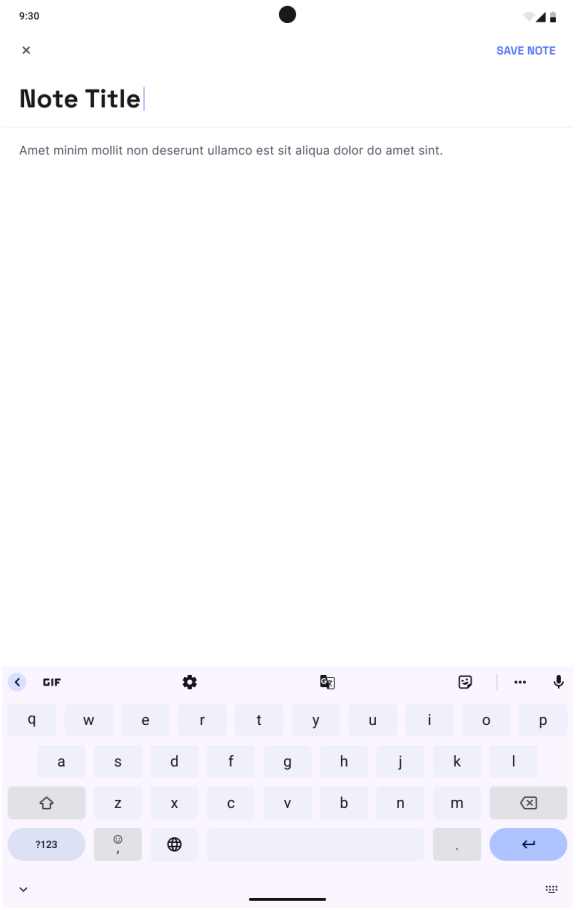


▼ Note Detail Screen

- This is a screen where existing notes can be updated.
  - The existing note's title and content text must be used to populate the two texfields on this screen
- Top bar with "x" icon far left and "Save Note" text far right
  - On "x" tap:
    - If the title or note content is different from the original note then show a confirmation dialog
      - Title: "Discard Changes?"
      - Body: "You have unsaved changes. If you discard now, all changes will be lost."
      - Confirmation button: "Discard"
      - Cancel button: "Keep Editing"
      - On confirm button tap, do not save the note and pop Note Detail Screen from the backstack
    - If no changes to the note was made, just pop Note Detail Screen from the backstack
  - "Save Note" text at top right corner of screen
    - On tap:

        - Update the note with the edits made by the user
        - Once the note has been saved, pop Note Detail Screen from the backstack
- Landscape mode
    - Place the topbar content in a sidebar to the left of the screen
- Main content is scrollable if content is larger than screen height
- Title text that is editable
    - Use the note's title as default value
    - Placeholder text is "Note Title"
    - Automatically gains focus when user navigates to this screen
- Content text that is editable
    - When note has content text, use it as default value
        - A note is considered to note have any content when the value is an empty string
    - Placeholder text is "Tap to enter note content"
- Keyboard should never overlap the text, it should instead push up the screen contents


9:30
SAVE NOTE
Note Title
Amet minim mollit non deserunt ullamco est sit aliqua dolor do amet sint.
GIF
q w e r t y u i o p
a s d f g h j k l
z x c v b n m
?123


9:30
SAVE NOTE
Note Title
Amet minim mollit non deserunt ullamco est sit aliqua dolor do amet sint.
GIF
q w e r t y u i o p
a s d f g h j k l
z x c v b n m
?123