Tasky Requirements

This requirements document is used to explain the main functionality behind Tasky. The exact looks and colors of a specific UI element can be seen on the mockups.

Note, that this serves as something to give you an overall impression of what the app should be able to do. Feel free to decide on your own 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 Tasky here:
https://www.figma.com/file/ATBztE2a0RObai1XZVeiVC/Tasky-app?node-id=0%3A1

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)

Authentication

- Users should be able to register and login

- To register, a user needs to provide their full name, email address and a password

- The full name must be between 2 and 50 characters

- The email must be a valid email

- The password must contain lowercase and uppercase letters as well as at least one digit

- The password needs to be at least 9 characters long

- When full name/email is valid, there should be a checkmark on the right side of the textfield

- For all password fields, the user should be able to toggle their visibility using an eye icon as per mockups

- Once successfully logged in, the user should stay logged in as long as the token is valid or they logout

Splash Screen

- The app should display a splash screen during which it will check if the user has an active session or needs to login

- If the user has an active session, navigate to the agenda screen, else to login screen

Agenda screen

- The agenda screen is the first screen that will open after a user logs in

- It will display the user's agenda for a given day

  - The agenda consists of events (any event you created or you're attendee in), tasks and reminders

  - All agenda items should be ordered ascending by time

  - There should be a time needle (as per mockups) to give the user an impression of the past and future agenda items

  - A click on the little circle of a task should toggle its isDone value

- The toolbar should consist of an option to select the date, which will be done using a DatePickerDialog (if you're using Compose, there's a good library for that: https://github.com/vanpra/compose-material-dialogs)

- On the right of the toolbar, there should be a profile icon consisting of the logged in user's full name initials

  - E.g. if the full name is Philipp Lackner, display the initials → PL

  - If the full name consists of one word, display the first 2 characters → PH

  - A click on that icon opens a drop down context menu with the option to logout

- At the top, display a list of the next 6 days the user can click on to select it and load the agenda for that day

- Each agenda item should display its title, description and time (For events, display both `from`

and `to` times)

- Each agenda item should have a context drop down menu to open, edit or delete it

- A click on open is equivalent to clicking on the agenda item. It will simply open it on the detail screen in non-edit mode

- A click on edit will open the item on the detail screen in edit mode

- A click on delete should show a confirmation dialog with the options `Cancel` and `OK` to let the user confirm the deletion

- A floating action button is used to add a new agenda item

  - It will open a drop down context menu to let the user select between adding an event, a task or a reminder

Event Detail Screen

- The event detail screen is used to create, edit and display events

- An event consists of the following data

  - Title

  - Description (optional)

  - Time range (from and to)

  - Reminder time - can be one of the following

    - 10 minutes before

    - 30 minutes before

    - 1 hour before

    - 6 hours before

    - 1 day before

  - Up to 10 photos (optional)

  - A list of attendees

- When an existing event is displayed on this screen, it's not editable by default

  - To make it editable, the user needs to click on an edit icon in the toolbar which will make all its fields clickable and editable

- When an event is editable, a "Save" button will display in place of the edit icon
- When an event is newly created, it starts in editable mode

- As per mockups, a click on the event title and description will lead to a separate screen where the user can change these values and confirm by clicking on "Save"

- A click on the date or time field will open a corresponding date/time dialog to edit it

- A click on add photo will open the users file chooser that displays the user's image files

- Clicking on an already added photo will open it up on the Photo Detail Screen where the user also has the option to delete it from the event

- The reminder time is chosen via a context drop down menu

- If you're the event creator, there will be a "Delete event" button at the bottom which will open a confirmation dialog to confirm the deletion before navigating back to the agenda screen

- A click on the add attendee button will open a dialog containing a text field where the user can enter an email of another user
  - Same as for the other email fields in the app, display a checkmark next to it if it's a valid email
  - The add button should only be clickable if the email is valid
  - If the user clicks on the add button and the API responds with an error relevant for the user (such as no user with that email exists), display that error as text below the email textfield

- Attendees
  - An added attendee's status will automatically be set to `going`
  - Once added, the attendee list item should display their initials in an avatar icon, their full name and the option to remove them from the event
  - If an attendee who is not the event owner clicks on the edit event button, the only thing they can edit is the reminder which will only apply for them and not other attendees
  - The attendees can be filtered by all/going/not going

- If you're an attendee, but not the event owner, instead of the "Delete event" button, it will display "Leave event" or "Join event" depending on your going status
- Event creators can't mark themselves as going/not going - they're always considered as going
- In the attendee list, display a "creator" badge next to the creator
- In offline mode, the user can't edit the event's photos

Task Detail Screen

- The task detail screen is used to create, edit and display a task
- A task is a reminder that can be marked as done/undone (like a todo item)
- A task consists of the following data
  - Title
  - Description
  - Time (Date + time)
  - Reminder time (same options as for events)
  - Whether it's done or not
- The mechanism to edit a task is the same as for events
- A task can not be marked as done on the task detail screen, that only works on the agenda screen (therefore it's undone by default)

Reminder Detail Screen

- The reminder detail screen is used to create, edit and display a reminder
- A reminder is the same as a task, just that you can't mark it as done/undone
- A reminder consists of the following data
  - Title

- Description
- Time (Date + time)
- Reminder time (same options as for events)
- The mechanism to edit a reminder is the same as for events/tasks

Notification reminders

- Once an agenda item has been saved, a reminder for it should be schedule at the selected reminder time
- When the reminder time is reached, a notification should show up with the agenda item's title and description
- A click on the notification should open it on the corresponding detail screen
- Reminders should be persisted on device restart
- There should be a mechanism in place that also schedules reminders for agenda items the user didn't add themselves
  - E.g. when you're added as attendee to an event without opening the app, a reminder should still be set for that event, even if the user might not be in the app
  - Another scenario: The user is logged into their account on device A and device B. The user adds a task on device A, then the notification should also show up on device B
  - The app should check at least every 30min if there are new agenda items and if so set the corresponding reminders
- When an agenda item is deleted, the corresponding reminders should be deleted as well
- A notification should only show if the reminder time is in the future
  - Example: The current time is 5pm. Event A lasts from 6pm-10pm and the reminder is set to 6h before. That notification shouldn't show up because 6pm - 6h = 12pm which is earlier than the current time.

Cache

- The app should work in offline mode once the user is logged in

- Therefore, the user should be able to add, edit and delete agenda items when not being connected to the internet

- Once an internet connection is established again, the items that have been added, updated and deleted in offline mode should be synced with the API