Take-Home Challenge for Flutter Mobile Developer

Welcome to the take-home challenge for the Flutter mobile developer position. We are excited to see your skills and experience in action. The challenge is to build a time tracking app for tasks. The app should have the following functional requirements:

1. A kanban board for tasks, where users can create, edit, and move tasks between different columns (e.g. "To Do", "In Progress", "Done").
2. A timer function that allows users to start and stop tracking the time spent on each task.
3. A history of completed tasks, including the time spent on each task and the date it was completed.
4. Users should be able to Comment on each task

For API services that you need during the challenge:

1. Use available services in https://developer.todoist.com/rest/v2/#overview
2. Authentication feature is not required in application so it's better to use `Test Token` in App Management Console
3. If the functionality that you want to implement is not provided by API services, use local solutions.

It will be important as well to follow these non-functional requirements as well:

1. Best practices of software development such as DRY (Don't Repeat Yourself), KISS (Keep It Simple, Stupid), and SOLID (Single responsibility, Open-closed,

Liskov substitution, Interface segregation, Dependency inversion) should be incorporated.

2. MVP(Minimum Viable Product) principle: This principle suggests that you should aim to create a minimum viable version of the product that can be released to users as soon as possible, and then iteratively add new features and improve the existing ones based on user feedback.
3. User-centered design: The app should be designed with the user's needs, goals, and preferences in mind. This includes making the app easy to use, visually appealing, and accessible to all users.
4. Performance optimization: The app should be optimized for performance, including fast loading times, smooth scrolling, and minimal use of memory and battery.
5. Code readability and maintainability: The code should be easy to read and understand, and should be organized in a way that makes it easy to maintain and update.
6. Test-Driven Development (TDD): Test-driven development is a software development process in which tests are written before the actual implementation of the feature. This allows you to catch and fix bugs early on and it helps to ensure the reliability and robustness of your code.
7. Continuous Integration and Continuous Deployment (CI/CD): Continuous integration and continuous deployment are practices that automate the process of building, testing, and deploying code changes. This allows for faster development and deployment cycles, as well as more frequent updates to the app.

Here are some anti-patterns that you should avoid when working on the take-home challenge for a Flutter mobile developer position:

1. Not following best practices for software development: Not following principles such as DRY, KISS, and SOLID, or not applying other best practices

such as performance optimization, user-centered design, and test-driven development can lead to a poorly-designed and difficult-to-maintain app.

2. Not considering the user's needs: Focusing too much on technical details and not enough on the user's needs, goals, and preferences can lead to an app that is difficult to use and not well-received by users.
3. Not testing the app: Failing to properly test the app can lead to bugs and other issues that can negatively affect the user's experience.
4. Not optimizing for performance: Not paying attention to performance optimization can lead to slow loading times, poor scrolling, and other issues that can negatively affect the user's experience.
5. Not considering security: Not implementing security best practices can lead to vulnerabilities in the app that can put user data at risk.
6. Over-engineering: Not considering the MVP principle and over-engineering the solution, making the app too complex and difficult to use.
7. Not following design standards: Not following the design standards and guidelines for the platform can lead to an app that looks and feels out of place, which can negatively affect the user's experience.

By avoiding these anti-patterns, you can ensure that your app is well-designed, easy to use, and well-received by users.

The time frame for a take-home challenge like this will depend on various factors, such as the complexity of the app, your experience level, and the availability of resources. However, as a rough estimate, you could expect to spend around 20-40 hours on the challenge.

It's important to keep in mind that the time frame for a take-home challenge is not so strict as it is for a project with a deadline, the main goal is to have a good quality of the solution and the best practices applied.

It's recommended to have a plan, a schedule of the different activities that you need to perform, this will help you to have a better control of the time you will spend on each task, also taking into account that you need to add some time for testing, debugging and finalizing the project.

It's important to note that this is just an estimate, and the actual time you spend on the challenge may be more or less depending on your experience and the specific requirements of the challenge.

Here are a few bonus features that you could add to the take-home challenge for a Flutter mobile developer position:

1. Analytics: Implement analytics to track how users are using the app and identify areas for improvement.
2. Notifications: Add push notifications to remind users of upcoming tasks or to let them know when a task has been completed.
3. Customizable themes: Allow users to customize the look and feel of the app by choosing from a selection of pre-defined color schemes.
4. Integration with third-party tools: Allow the app to integrate with other productivity tools such as calendars and task managers.
5. Offline functionality: Allow the app to work offline and automatically synchronize data when a connection is re-established.
6. Multi-language support: Add support for different languages, this will make the app more accessible to a global audience.

Keep in mind that these are just a few ideas, and there may be other features that you can add to the challenge depending on the requirements of the app. The important thing is that the features you add should be relevant and useful to the app's users.

We recommend using the latest version of Flutter for the challenge. Please submit the source code for the app in a **public repo** along with a brief documentation explaining how to run the app and **screenshots or a demo video** and any additional information you think is relevant.

Good luck and have fun!

You can look for inspiration on Dribbble, you can search for "Kanban board UI" and "time tracking app UI" or you can use any other design resources that you prefer, it is important that the design is easy to use and visually appealing.