High-level Design (4%)

Describe which architectural pattern you would use to structure the system. Justify your answer.

We are choosing a Data-Centric Architecture for our To-Do app. This means we'll focus on organizing and handling task details and categories efficiently. By putting all our attention on managing task-related information, we can make sure tasks are clear, well-sorted, and easy to prioritize and categorize. This will help create a system that's strong and user-friendly, making task management smooth and organized for everyone using the app.
It streamlines task management enhances data security and facilitates integration with other systems. This approach allows for robust data protection, ensuring user information and task details are securely handled. It supports seamless integration with external services like calendars or email, enhancing the app's functionality and user experience. Additionally, this architecture can adapt to changing data models, making it flexible for future updates, and ensuring efficient maintenance.

Low-level Design (4%)

Discuss which design pattern family might be helpful for implementing this project. Justify your answer, providing a code or pseudocode representation and an informal class diagram.

For implementing this project, a Creational design pattern like the Builder would work well. The Builder pattern separates the construction of complex objects from their representation. In the context of a To-Do app, tasks might have various attributes (description, due date, priority, etc.), and the Builder pattern can facilitate the step-by-step creation of tasks with different configurations.

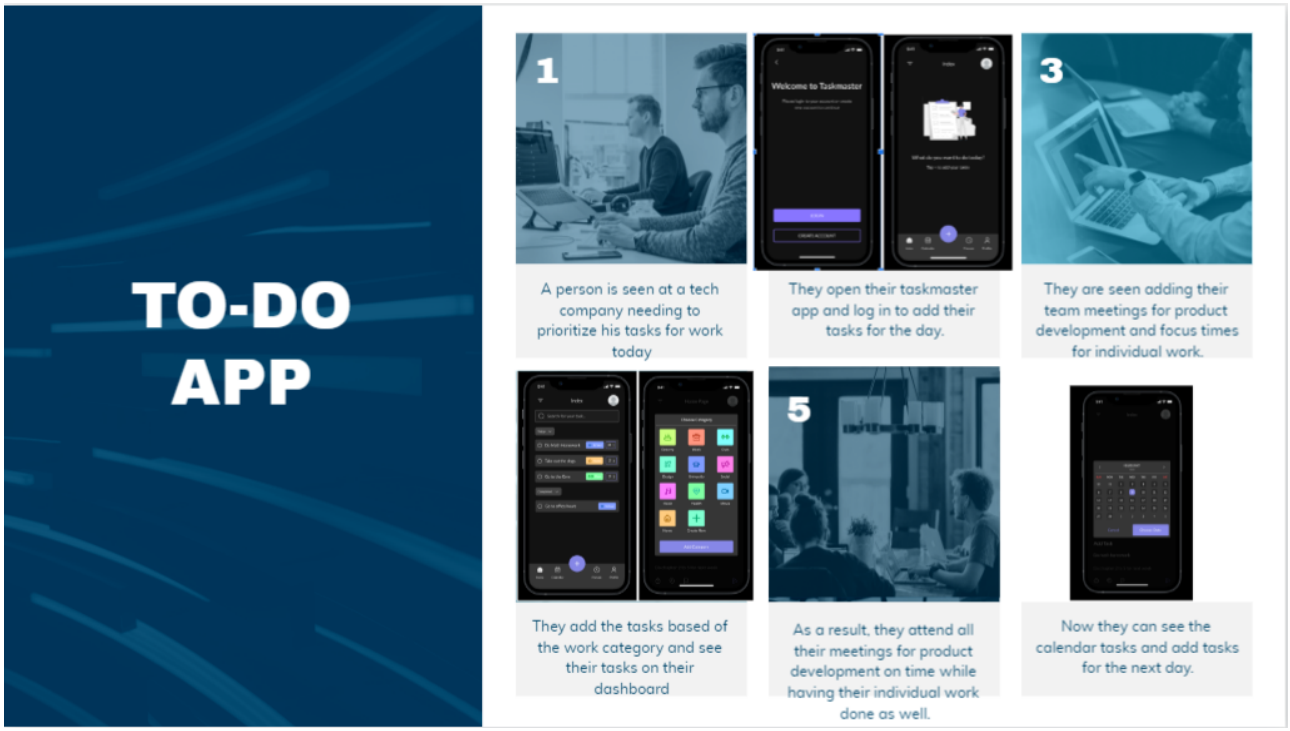
```
Task
    self.description = description
    self.due_date = due_date
    self.priority = priority


def set_description(self, description):
    self.task.description = description
    return self

def set_due_date(self, due_date):
    self.task.due_date = due_date
    return self
```

```
def set_priority(self, priority):
        self.task.priority = priority
        return self
```

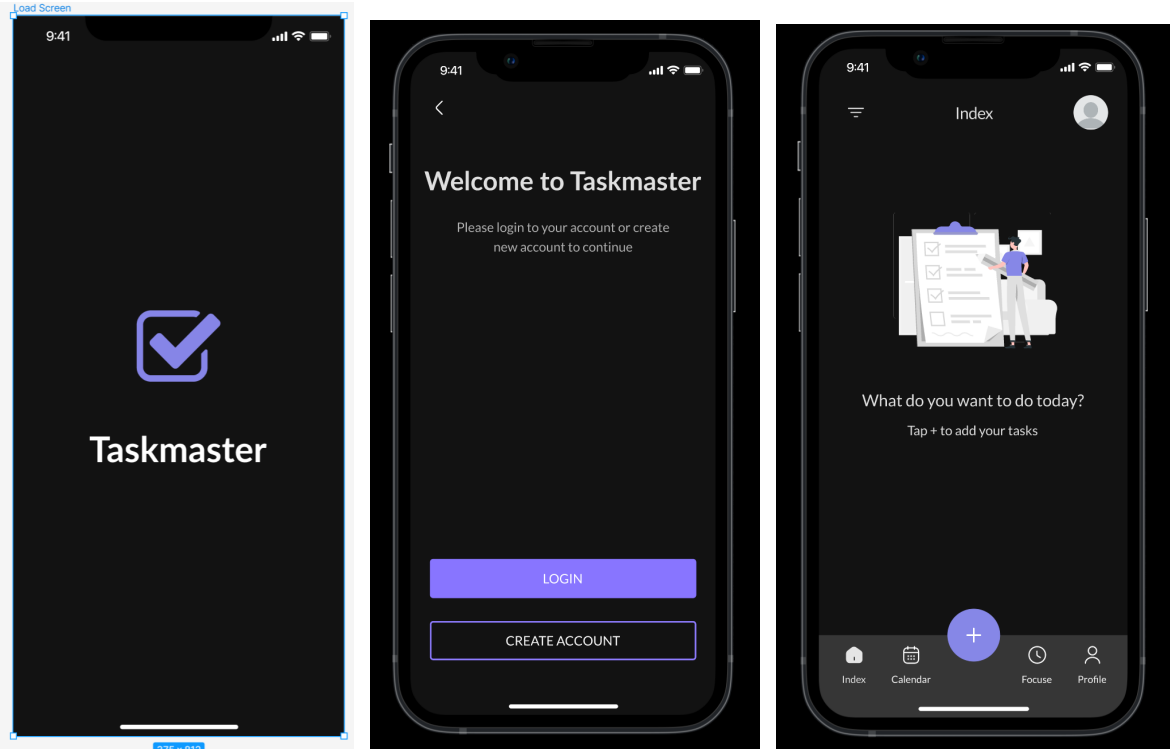
Design Sketch (4%)

Rationale:
In this situation, we can see how the design of our to-do app would act in a relevant situation. We see a person go by their day-to-day and interact with the interface of the to-do app to go through their day. We chose this design for our app as our top priority was organization and flexibility. This design allows our users to keep their things organized with color coding and flag options as well as flexibility as it gives our users an easy job to add the task as well as multiple different options on how they want to categorize/prioritize it.

Alex, a software engineer, sat at his desk with a steaming cup of coffee. Today, like every day, began by opening the Task Master app on his phone. Before diving into the usual flurry of coding and meetings, Alex needed to set the stage for a productive day. He started by reviewing the tasks he had listed the previous evening, a mix of coding assignments, bug fixes, and team collaboration tasks. With a few clicks, he prioritized them.



In part 2 you can see that Alex is opening the taskmaster and adding tasking for the day. This is how the app will look on his phone.



Our app features a simple black background, chosen because it's easy on the eyes and looks good, especially for those who use night mode on their phones. We designed the app to be easy for everyone to use, no matter their age. On the home screen, we put the most important stuff front and center. There's a clear message, “Tap + to add your tasks,” to guide you on how to get started. We know a calendar is key in a to-do app, so we placed a button for it at the bottom left for quick access. And to update your info, just tap the profile button on the bottom right. We made everything straightforward and user-friendly.

Project Check-In

Complete this survey to provide an update on your team progress on the project for this semester. Only one team member needs to complete this for the group.

Completed survey.

Process Deliverable (3%)

The submission for this deliverable will depend on the specific SE process model your team plans to use to complete the group project (as described in your project proposal). Example submissions for different processes include:

- ***Prototyping: submit a prototype of your system (it can be as formal/informal as needed)***
- ***Scrum: submit the notes (including each teammate) from your most recent scrum meeting***
- ***Kanban: submit a list of prioritized tasks from your task management system (and why they are prioritized)***
- ***Waterfall: submit supplementary planning documentation***
- ***Extreme programming: submit acceptance test criteria***
- ***Spiral: submit risk analysis***
- ***Code-and-fix: essay on why you used code-and-fix or up-to-date source code in your GitHub repository For other processes not listed above, the instructor will contact you with the exact submission requirements for this task.***

Prototype Figma Link:
https://www.figma.com/file/4P3nnGH0XAwUFPnR6c8U7K/TaskMaster?type=design&node-id=1%3A8499&mode=design&t=NH36JJq6Cs2roOD3-1

Screenshots:

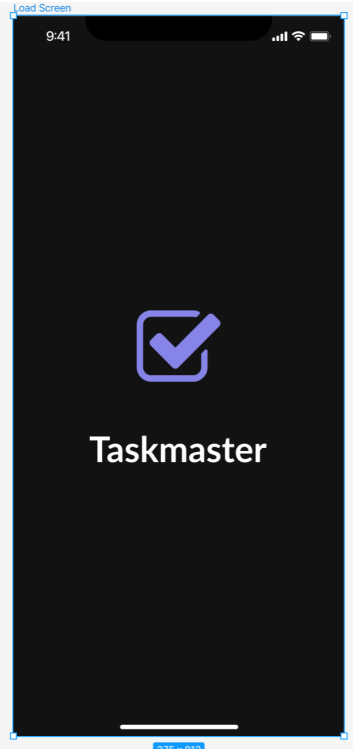


Figure 1: Load Screen

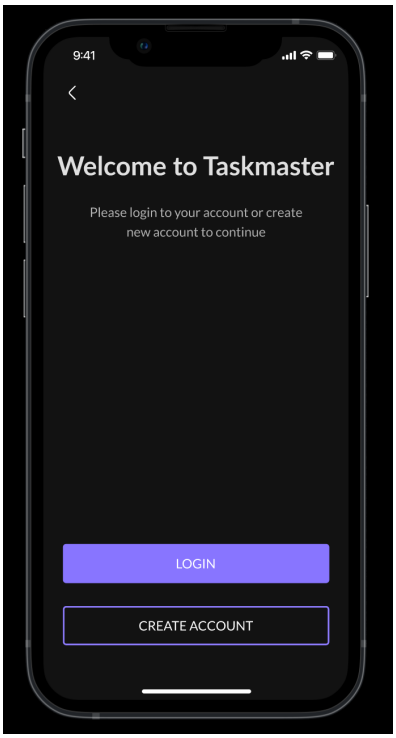


Figure 2: Start Screen

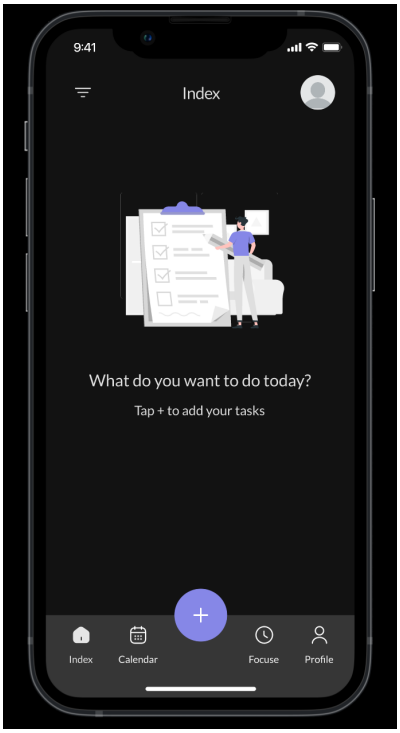


Figure 3: Home Screen

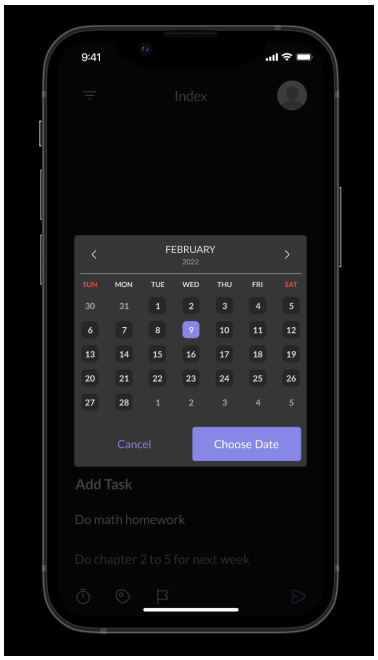


Figure 4: Task Date

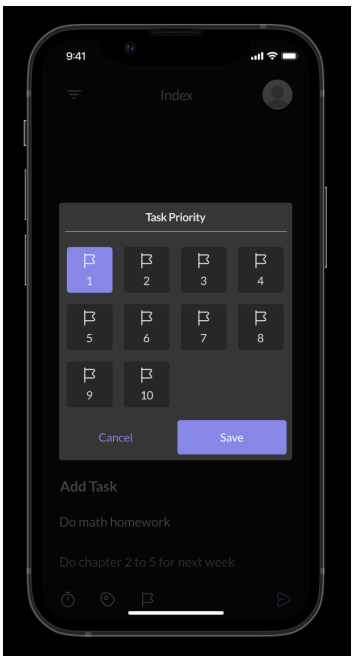


Figure 5: Task Priority

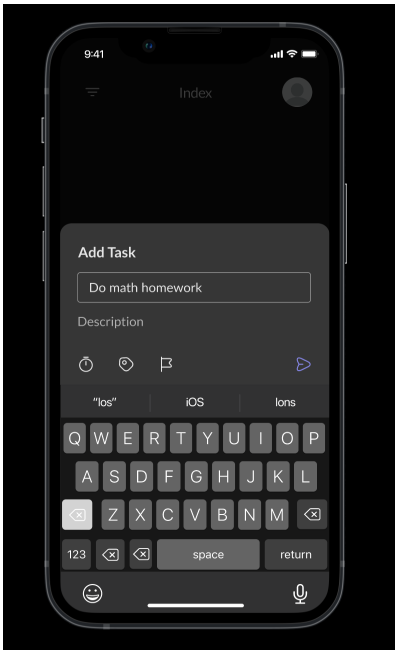


Figure 6: Home Screen with Tasks

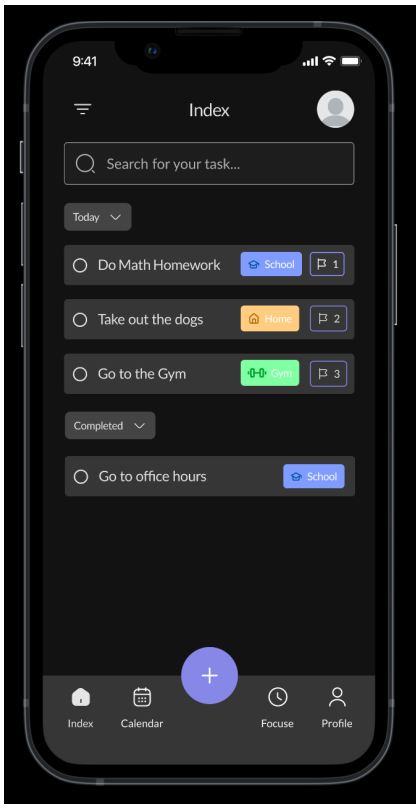


Figure 7: Home Menu with Tasks

Figure 8: Choose Categories

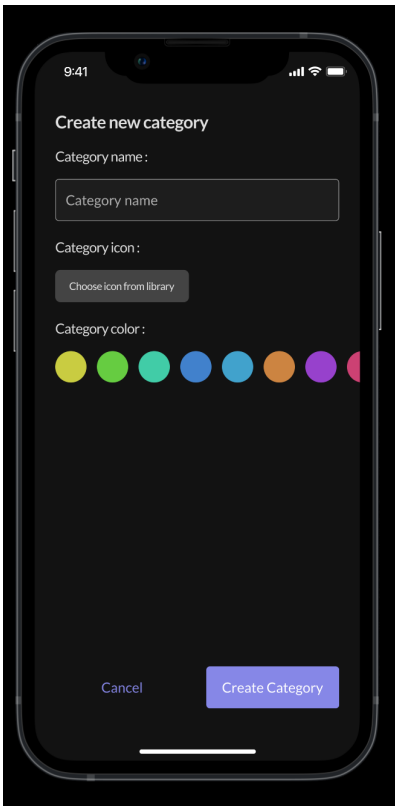


Figure 9: Create category