

**Faculty of Computers and Artificial Intelligence**

**Cairo University**

**CS251:** **Intro to Software Engineering**

**Assignment 2**

**Dr. Mohammed El Ramly**

**TechRadar**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **ID** | **Section** | **Email** |
| Mariam Badr Yahya Abd El-Naby | 20230391 | S17,18 | Mariambadr145@gmail.com |
| John Ayman Demian | 20230109 | S17,18 |  |
| Emad George Mattar Hanna | 20230244 | S23,24 |  |

**May 2025**

**20230391**

**Prompt engineering:**

Prompt engineering has become one of the essential skills when working with artificial intelligence, especially with large language models like ChatGPT. Simply put, it's about writing instructions or questions (prompts) in a smart and clear way so that the model gives you accurate and relevant responses. This skill enables anyone—even without a technical background—to interact with AI effectively.

A well-crafted prompt can make the model explain, summarize, translate, or even write code, all depending on how you guide it. To use AI efficiently, it's important to understand the different types of prompt engineering, including:

1. **Zero-shot Prompting**: You give a direct request without any examples—useful for simple tasks.
2. **One-shot Prompting**: You provide one example to clarify what you want.
3. **Few-shot Prompting**: You give multiple examples, helping the model better understand the context.
4. **Chain-of-thought Prompting**: You ask the model to think step-by-step, which is helpful in logical or math-based questions.
5. **Role-based Prompting**: You ask the model to respond as if it were in a specific role (like a doctor or teacher).

Over time, prompt engineering has become more than just a useful trick—it’s a new kind of “digital intelligence.” If you master it, you can steer AI with precision and get far better results. And while it does come with some challenges—like the model sometimes giving inaccurate answers—practice and experimentation help you improve quickly.

**In conclusion:**  
Prompt engineering is your key to mastering AI. The smarter your prompt, the better your results. As AI tools continue to evolve, this skill is no longer optional... it’s essential!

**20230109**

**Review of React Hook Form:**

React Hook Form (RHF) is a popular and fast library used to manage form state and data validation in React applications. Unlike traditional libraries that rely on controlled components and internal state management, RHF takes a different approach by using uncontrolled components and leveraging the native behavior of HTML forms. The result? Better performance and a cleaner, easier-to-use API for developers.

One of RHF’s biggest advantages is that it minimizes re-renders. Since it uses uncontrolled components under the hood, it doesn’t trigger a re-render on every keystroke like typical controlled forms do. This makes it very efficient—especially in large or complex forms. The library uses useRef and register() to connect inputs to the form state, making it both lightweight and fast.

RHF also offers strong and flexible validation support. Developers can define validation rules directly in the code or use libraries like Yup or Zod for more advanced validation. It also supports asynchronous validation, which is helpful for real-time checks with APIs. Error handling is simple because the errors object is readily available to show helpful messages to users.

Integration with UI libraries like Material-UI, Chakra UI, and Ant Design is very smooth. Even though these libraries use controlled components, RHF provides a Controller component that makes integration easy—so you can use RHF with almost any UI framework.

The developer experience with RHF is excellent. The API is concise and well-documented, and TypeScript support is strong, giving you accurate typing for form values and validation rules. This makes development easier and reduces the chances of runtime bugs. There’s also the RHF DevTools extension that helps you visualize form state during development and makes debugging easier.

In short, React Hook Form is a powerful, fast, and developer-friendly library for managing forms in React. It balances performance with flexibility, provides excellent validation features, and works well with both native inputs and third-party UI libraries. That’s why it’s a top choice for modern React apps and makes working with forms much easier.

**20230244**

**.NET Core:**

.NET Core is a free, open-source development platform made by Microsoft. It allows developers to build many types of applications such as web apps, desktop apps, mobile apps, APIs, cloud services, and more. It is a cross-platform tool, which means you can run your code on Windows, Linux, and macOS without changing it.

One of the things I liked most about .NET Core is its speed and performance. It runs very fast compared to older frameworks, and it's great for building web applications and APIs. Also, it's lightweight and doesn’t require installing big packages to get started. You can build and run a small web server in just a few lines of code.

.NET Core uses the C# programming language, which is easy to learn and has modern features. It also works with popular tools like Visual Studio or Visual Studio Code. When I tried it, I found it easy to create a small web API that returned user data using just a few commands in the terminal and a basic controller file.

Another great point is that .NET Core is part of a bigger platform called .NET 6/7/8, which unifies all app development types. Microsoft and the developer community actively update it, which means it's improving all the time.

In short, I think .NET Core is a powerful and friendly tool for developers. Whether you're new to coding or already experienced, it helps you create strong applications fast. I would recommend it to anyone interested in modern software development, especially if they want to work in companies using Microsoft technologies.