**Chapter 1: Overview**

**Design Philosophy**

When you learn about a framework, it’s important to know not only what it does but what principles it follows. Here are the guiding principles of the Spring Framework:

- Provide choice at every level. Spring lets you defer design decisions as late as possible. For example, you can switch persistence providers through configuration without changing your code. The same is true for many other infrastructure concerns and integration with third-party APIs.

- Accommodate diverse perspectives. Spring embraces flexibility and is not opinionated about how things should be done. It supports a wide range of application needs with different perspectives.

- Maintain strong backward compatibility. Spring’s evolution has been carefully managed to force a few breaking changes between versions. Spring supports a carefully chosen range of JDK versions and third-party libraries to facilitate maintenance of applications and libraries that depend on Spring.

- Care about API design. The Spring team puts a lot of thought and time into making APIs that are intuitive and that hold up across many versions and many years.

- Set high standards for code quality. The Spring Framework puts a strong emphasis on meaningful, current, and accurate javadoc. It is one of very few projects that can claim clean code structure with no circular dependencies between packages.

**Chapter 2. Core Technologies**A diagram of a software framework

Description automatically generated with medium confidence