Summary

In this chapter, we have covered some important aspects of implementing globalization in ASP.NET 3.5. We saw that although it is easy and simple, there are a few important points and best practices to bear in mind when globalizing our ASP.NET web applications:

- Do not rely on the web browser's settings. Provide a link on the application (may be in the header) so that the users can select their choice of language.
- Use `.resx` files to separate out presentation-related data in the GUI. Resource fallback is the approach used by ASP.NET when it is unable to find the resource file for a particular culture. It will first go to the neutral resource file, and then to the default or fallback resource file.
- Strike a balance between global and local resource files based on the application's needs.
- We can extend the Resource-Provider-Model in ASP.NET to store localized content in database tables instead of the `.resx` files. This gives us the flexibility to modify and update localized content easily at runtime without disturbing the main application.

In the nine chapters of this book so far, we saw and understood how to architect and design our ASP.NET web applications based on our actual project needs. We also learnt about the various architectural options, and when to use a particular approach over the other considering the scope, complexity and long term goals of our projects.

To summarize, a good architecture should take into consideration the following list of parameters:

- scalability
- reusability
- interpretability
- flexibility
- maintainability
- security
- robustness
- readability

Each of the above parameter plays a decisive role in forming the core architecture of our projects. In this book, we have studied only a few approaches but there can be potentially infinite ways to program and develop applications, and it's impossible to learn and study them all. There are many ways in which we can structure and develop our applications, and no particular architecture or design is a perfect solution. Moreover, an architecture designed for a particular scenario might not work well in other cases. So the best strategy is to understand the current as well as the future business needs of the project and decide on which architecture will best suit the long-term and as well as short-term goals.