Ausbaustufe 2 – SWK5

(Abgabe: 22. 12. 2021, 24:00 Uhr)

(A2.1) *AaaS.Core:* Implementieren Sie auf Basis der in Ausbaustufe 1 entwickelten Datenzugriffsschicht die gesamte Funktionalität der Komponente *AaaS.Core*. Überlegen Sie sich, wie die Klienten auf die (Geschäfts-)Logik zugreifen sollen. Fassen Sie die erforderlichen Funktionen zu logischen Gruppen zusammen und definieren Sie für jede Gruppe ein Interface. Alle REST-Services sollen (statisch) ausschließlich von diesen Geschäftslogik-Interfaces abhängig sein.

Entwerfen Sie die Klassen der Geschäftslogik mit besonderer Sorgfalt und legen Sie besonderen Wert auf eine flexible Software-Architektur. Sorgen Sie insbesondere dafür, dass Detektoren und Aktionen einfach ausgetauscht werden können. Implementieren Sie alle Funktionen, die nicht unmittelbar für die Umsetzung der REST-Schnittstelle oder des Web-Clients benötigt werden, in der Geschäftslogikkomponente von *AaaS.Core*.

Die zentralen Bestandteile von *AaaS.Core* stellen die Detektoren und die Aktionen dar. Überlegen Sie sich, wie Sie die korrekte Funktionsweise dieser Komponenten testen und demonstrieren können.

(A2.2) *AaaS.Api:* Diese Systemkomponente ist mit ASP.NET als REST-basiertes Web-Service zu realisieren. Die Anforderungen an die zu exportierende Funktionalität des Web-Service ergeben sich aus der Funktionalität von *AaaS.Web* bzw. *AaaS.ClientSDK*.

(A2.3) *AaaS.ClientSDK:* Bündeln Sie die Funktionalität zum Versenden von Telemetriedaten in einer .NET-Bibliothek. Implementieren Sie auf dieser Basis einen einfachen Client, mit dem Sie die volle Funktionalität des Softwaresystems zeigen können.

(A2.4) *Automatisierte Tests:* Erweitern Sie Ihre Test-Suite um Tests der Geschäftslogik. Die Tests sollen so weit wie möglich die gesamte Funktionalität der Anwendung abdecken.

Isolieren Sie in Ihren Tests die Geschäftslogik von der Persistenzschicht. Ersetzen Sie dazu Ihre DAOs durch Mock-Objekte. Setzen Sie dafür ein Mocking-Framework ein.

(A2.5) *Dokumentation:* Führen Sie die notwendigen Ergänzungen in der Dokumentation durch. Aus Ihrer Dokumentation soll die Architektur des Gesamtsystems hervorgehen, insbesondere die Struktur der Klassen und Interfaces und die Kopplung der Systemkomponenten sollte übersichtlich dargestellt werden. Hierzu eignen sich besonders gut UML-Klassen- und Paketdiagramme.