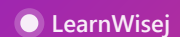


KI-Coding-Übung

Erstellen Sie das Beispielprojekt SupportDesk dieses Moduls mit ChatGPT oder Claude aus einem fertigen Prompt, der das Modell auf die Spezifikation des Moduls verweist – und prüfen, starten und erweitern Sie dann, was Sie erhalten, auch im Vergleich mit unserem eigenen Referenz-Build auf GitHub.



Der Assistent liefert die schnellen Hände. Der Architekt bleiben Sie.

Ein Assistent erstellt das Beispielprojekt dieses Moduls in wenigen Minuten. Was er nicht kann, ist zu entscheiden, ob das Ergebnis der richtige Entwurf für Ihre Anwendung ist – diese Entscheidung bleibt bei Ihnen. Das Urteilsvermögen bringen Sie mit: Instruieren Sie ihn sorgfältig und prüfen Sie dann, was zurückkommt, anhand dessen, was Ihnen Modul 7 gerade vermittelt hat.

Worum Sie bitten

Machen Sie die Support Desk Data Console sicher für mehrere Benutzer: Ergänzen Sie RowVersion im Edit-Modell und in seinem verborgenen Zustand, setzen Sie OriginalValue, bevor Sie eine bestehende Entität speichern, reproduzieren Sie einen Konflikt, indem Sie dasselbe Ticket in zwei Browser-Sessions bearbeiten, fangen Sie DbUpdateConcurrencyException ab, erstellen Sie eine Konfliktliste mit vorgeschlagenen Werten und Datenbankwerten, ergänzen Sie gemäß Richtlinie die Pfade Reload und Overwrite, erzeugen Sie ein idempotentes SQL-Migrationsskript mit Deployment-Hinweisen und umgebungsspezifischen Connection-Strings, lassen Sie das Logging sensibler Daten außerhalb der Entwicklung ausgeschaltet und schließen Sie dann das Review des Abschlussprojekts ab.

Das Modul verlangt diese Ergebnisse, und der Prompt unten fordert genau diese vom Assistenten an:

- ✓ RowVersion im Edit-Modell mitgeführt und vor dem Speichern als OriginalValue gesetzt
- ✓ Reproduzierbarer Nebenläufigkeitskonflikt, als DbUpdateConcurrencyException abgefangen
- ✓ Konfliktdialog, der vorgeschlagene Werte und Datenbankwerte auflistet, mit den Pfaden Reload und Overwrite
- ✓ Idempotentes SQL-Migrationsskript mit Deployment-Hinweisen und umgebungsspezifischen Connection-Strings
- ✓ Logging sensibler Daten außerhalb der Entwicklung deaktiviert und Review des Abschlussprojekts abgeschlossen

Warum es so aufgebaut ist

Zwei Bearbeiter öffnen das Ticket T-1042 in zwei Browser-Sessions; der eine schließt es, der andere erhöht seine Priorität, und welches Speichern als zweites ankommt, entscheidet, was sich die Datenbank merkt. Zeilen zu sperren ist in einer Webanwendung keine Option; die Support Desk Data Console muss also

beide Benutzer arbeiten lassen, die Kollision beim Speichern erkennen, sie erklären und einen Menschen entscheiden lassen. Dieses Modul behandelt außerdem Transaktionen, Konfiguration, Migrationen, Logging, Tests und das Abschlussprojekt.

Diese Begriffe muss der Assistent richtig verwenden. Geht die Antwort nachlässig mit ihnen um, hat er das Modul nicht verstanden:

Begriff	Was er hier bedeutet
optimistische Nebenläufigkeit	Benutzer bearbeiten frei; statt Zeilen zu sperren, erkennt die Anwendung beim Speichern, dass jemand anderes die Zeile seit dem Laden geändert hat.
RowVersion-Token	Ein von der Datenbank erzeugtes Concurrency-Token; EF Core setzt seinen Originalwert in die WHERE-Klausel des UPDATE, sodass eine geänderte Zeile zu null aktualisierten Zeilen führt.
OriginalValue	Die Version, die der Editor geladen hat, vor SaveChangesAsync auf der getrackten Entität wiederhergestellt, damit EF Core mit dem vergleicht, was der Benutzer gesehen hat.
DbUpdateConcurrencyException	Wird ausgelöst, wenn eine erwartete Zeile nicht betroffen war; ihre Entries enthalten die vorgeschlagenen Werte, und GetDatabaseValuesAsync holt, was jetzt in der Datenbank steht.
Konfliktdialog	Ein Wisej.NET-Formular, das Feld, Ihren Wert, Datenbankwert und Originalwert auflistet und Reload, Overwrite (sofern die Richtlinie es erlaubt) oder Merge anbietet.
Ausführungsstrategie	Der Wiederholungsmechanismus von EF Core für vorübergehende Fehler; eine explizite Transaktion muss darin laufen, damit eine Wiederholung die gesamte Unit of Work erneut ausführen kann.
idempotentes Migrationsskript	SQL aus dotnet ef migrations script --idempotent, das vor jedem Schritt die Migrationshistorie prüft und daher in jeder Umgebung sicher ist.

Selbst bauen – wählen Sie Ihren Assistenten

Beide Buttons öffnen einen neuen Tab, in dem **derselbe fertige Prompt** bereits eingetragen ist. Der Prompt verweist das Modell auf die Spezifikation dieses Moduls – eine einzelne Markdown-Datei mit der Lektion, der Lab-Aufgabe und den geforderten Ergebnissen –, sodass es mit dem echten Material arbeitet, statt zu raten, worum es in Modul 7 geht. Sie können [module-07.md](#) selbst öffnen, um genau zu sehen, was es lesen wird.



ChatGPT verwenden

Öffnet [chatgpt.com](#) mit dem Prompt dieses Moduls. Funktioniert mit GPT-5 oder



Claude verwenden

Öffnet [claude.ai](#) mit demselben Prompt. Wenn Sie Claude Code nutzen, können Sie

Codex; jeder Tarif mit Websuche ruft die Spezifikation ab.

ihn stattdessen in einem leeren Ordner in ein Terminal einfügen.



Unverändert abgeschickt, erhalten Sie ein Projekt, kein Code-Fragment

Der Prompt verlangt am Ende die gesamte Projektmappe als einzelne `.zip`-Datei mit Anleitung zum Kompilieren und Starten. Laden Sie sie herunter, öffnen Sie die Projektmappe in Visual Studio, drücken Sie F5, und SupportDesk startet auf `localhost`. Sie benötigen das .NET SDK und das NuGet-Paket `Wisej-4`; alles andere soll der Assistent laut Prompt selbst schreiben.

Ich arbeite gerade Modul 7 – „Nebenläufigkeit, Transaktionen, Deployment, Diagnose und Abschlussprojekt“ – des LearnWisej-Kurses „Data Binding mit EF Core“ durch und möchte das Beispielprojekt des Moduls gemeinsam mit dir erstellen. Bitte antworte auf Deutsch.

Lies zuerst die Modulspezifikation. Sie ist eine einzelne Markdown-Datei (auf Englisch) mit der vollständigen Lektion, der Lab-Aufgabe und den geforderten Ergebnissen:

<https://learnwisej.com/assets/courses/ef-core-binding/specs/module-07.md>

Erstelle dann ein vollständiges Wisej.NET-4-Projekt, das ich in Visual Studio öffnen kann:

1. Projekt „SupportDesk“, C#, net8.0 oder neuer, NuGet-Paket Wisej-4 4.1.x, Kestrel-Host, der `app.UseWisej()` aufruft.
2. Setze um, was das Lab der Spezifikation verlangt: Machen Sie die Support Desk Data Console sicher für mehrere Benutzer: Ergänzen Sie `RowVersion` im Edit-Modell und in seinem verborgenen Zustand, setzen Sie `OriginalValue`, bevor Sie eine bestehende Entität speichern, reproduzieren Sie einen Konflikt, indem Sie dasselbe Ticket in zwei Browser-Sessions bearbeiten, fangen Sie `DbUpdateConcurrencyException` ab, erstellen Sie eine Konfliktliste mit vorgeschlagenen Werten und Datenbankwerten, ergänzen Sie gemäß Richtlinie die Pfade `Reload` und `Overwrite`, erzeugen Sie ein idempotentes SQL-Migrationsskript mit Deployment-Hinweisen und umgebungsspezifischen Connection-Strings, lassen Sie das Logging sensibler Daten außerhalb der Entwicklung ausgeschaltet und schließen Sie dann das Review des Abschlussprojekts ab.
3. Liefere diese Ergebnisse und sag mir, wo sich jedes davon in der Projektmappe befindet:
 - `RowVersion` im Edit-Modell mitgeführt und vor dem Speichern als `OriginalValue` gesetzt
 - Reproduzierbarer Nebenläufigkeitskonflikt, als `DbUpdateConcurrencyException` abgefangen
 - Konfliktdialog, der vorgeschlagene Werte und Datenbankwerte auflistet, mit den Pfaden `Reload` und `Overwrite`
 - Idempotentes SQL-Migrationsskript mit Deployment-Hinweisen und umgebungsspezifischen Connection-Strings
 - Logging sensibler Daten außerhalb der Entwicklung deaktiviert und Review des Abschlussprojekts abgeschlossen
4. Halte dich an die Spezifikation, wo immer sie eine Wisej.NET-Klasse, -Eigenschaft oder ein -Event nennt – verwende genau diese. Wo sie eine Wahl offenlässt, sag mir, welche API du gewählt hast und warum.
5. Der Server bleibt maßgeblich: Zustand und Validierung leben in .NET, der Browser stellt dar und meldet zurück. Verlagere nie eine Entscheidung nach JavaScript, nur weil es dort einfacher ist.
6. Mach das Ergebnis vorführbar: Ich muss es starten und das Verhalten sehen können, um das es in diesem Modul geht – auch, was passiert, wenn etwas schiefgeht.

Bevor du Code schreibst, fasse in eigenen Worten zusammen, was du bauen wirst und wo jedes Teil liegen wird, damit ich den Entwurf korrigieren kann, solange das noch wenig kostet.

Wenn der Code fertig ist, packe die gesamte Projektmappe in eine einzelne .zip-Datei zum Herunterladen und nenne mir die genauen Schritte, um sie zu kompilieren und auszuführen.

Beantworte zum Schluss diese Fragen aus der Spezifikation und nutze dabei deine eigene Implementierung als Beleg:

- Was würde der Test mit zwei Sessions zeigen, wenn Sie die Zeile mit `OriginalValue` weglassen, und warum sähe das wie ein Erfolg aus, während die Änderung des ersten Bearbeiters stillschweigend verloren geht?
- Welche Auflösungspfade würden Sie einem normalen Agent anbieten und welche einem

Supervisor, und was muss nach jedem Pfad mit dem Grid und der RowVersion im Edit-Modell geschehen?

– Nach einem Deployment starten drei Produktivinstanzen. Welche davon wendet in Ihrem Entwurf die Migration an und wann, und was würde schiefgehen, wenn Migrate() im Startup jeder Instanz lief?

Frag mich nach allem, was unklar ist, bevor du anfängst.

Prüfen Sie dann, was zurückkommt

Code, der kompiliert, ist noch lange kein richtiger Code. Die spannenden Fragen sind Designfragen: Ist das wirklich das Wisej.NET-Control, das die Aufgabe verlangt? Folgt der Code dem Lebenszyklus, den das Framework vorgibt? Liegt der maßgebliche Zustand auf dem Server – oder wurde er im Browser abgelegt, weil das bequemer war? Lesen Sie das Ergebnis als Reviewer, nicht als Kunde.

Die Checkliste liefert Ihnen dieses Modul. Beantworten Sie jeden Punkt anhand des Codes, den Sie gerade erhalten haben:

- ✓ Was würde der Test mit zwei Sessions zeigen, wenn Sie die Zeile mit OriginalValue weglassen, und warum sähe das wie ein Erfolg aus, während die Änderung des ersten Bearbeiters stillschweigend verloren geht?
- ✓ Welche Auflösungspfade würden Sie einem normalen Agent anbieten und welche einem Supervisor, und was muss nach jedem Pfad mit dem Grid und der RowVersion im Edit-Modell geschehen?
- ✓ Nach einem Deployment starten drei Produktivinstanzen. Welche davon wendet in Ihrem Entwurf die Migration an und wann, und was würde schiefgehen, wenn Migrate() im Startup jeder Instanz lief?

Alles, was Sie korrigieren mussten, sollten Sie festhalten. Eine kurze Notiz zu den Konventionen und Entscheidungen, auf die Sie sich festgelegt haben – eingefügt in die erste Nachricht Ihrer nächsten Sitzung –, ist mehr wert als jeder noch so clevere Prompt.



Dieses Beispiel haben wir schon gebaut – vergleichen Sie Ihres mit unserem

Die Referenzimplementierung zu Modul 7 liegt im öffentlichen Beispiel-Repository. Bauen Sie zuerst Ihre eigene Version und öffnen Sie dann unsere: Die Unterschiede sind der interessante Teil.



[Beispiel zu Modul 7 auf GitHub öffnen](#)

Oder klonen Sie alle Beispiele mit `git clone https://github.com/iceteagroup/LearnWisej-Samples` und öffnen Sie [Data Binding with EF Core Course/Module 7](#).



Unser Beispiel ist ein Ausgangspunkt, kein fertiges Produkt

Es ist genauso entstanden wie Ihres – per Prompt und anschließend korrigiert – und endet bewusst dort, wo das Modul endet. Rechnen Sie mit Platzhaltern, wo Produktivcode hingehört, mit knapper Fehlerbehandlung und ohne Tests. Verstehen Sie das als Einladung: Nutzen Sie, was Sie in diesem Modul gelernt haben, und dazu den Assistenten, um es weiterzuentwickeln. Code zu verbessern, den Sie nicht selbst geschrieben haben, ist die Fähigkeit, die dieser Kurs eigentlich vermittelt.

Drei Prompts für den nächsten Schritt

Richten Sie Ihren Assistenten auf das Projekt, das Sie gerade gebaut haben – oder auf unseres –, und probieren Sie diese Prompts aus. Jeder ist eine Übung zum Prüfen und Erweitern, kein Neuschreiben.

- ✓ **Beweisen Sie, dass es wirklich funktioniert.** *"Geh dieses Projekt Ergebnis für Ergebnis anhand der Modulspezifikation unter <https://learnwisej.com/assets/courses/ef-core-binding/specs/module-07.md> durch und zeig mir den Beleg, dass jedes wirklich erledigt ist – den Codepfad, worauf ich klicken muss und was ich sehen sollte. Wo ein Ergebnis nur teilweise erfüllt ist, sag das klar, statt es schönzureden. Ergänze dann, was fehlt."*
- ✓ **Prüfen Sie es wie einen Pull-Request.** *"Führ mich durch diesen SupportDesk-Code, als würdest du den Pull-Request eines Kollegen prüfen und hättest ihn nicht selbst geschrieben. Wo lebt der Zustand, was passiert bei einer einzelnen Benutzeraktion von Anfang bis Ende, was würdest du zur Überarbeitung zurückgeben, und welche Teile würdest du nicht guten Gewissens einem Kunden vorlegen? Sei konkret und zitiere die Zeilen, die du meinst – fasse mir nicht den Entwurf zusammen."*
- ✓ **Tauschen Sie ein Teil aus, ohne die Schnittstellen zu verschieben.** *"Wähle den Teil dieses SupportDesk-Projekts, der am offensichtlichsten nur ein Platzhalter für etwas Echtes ist – eine Fake-Datenquelle, ein In-Memory-Speicher, ein vereinfachter Service – und ersetze ihn durch eine praxisnahe Implementierung. Der Vertrag auf beiden Seiten darf sich nicht ändern: dieselben Typen, dieselben Aufrufe, dasselbe Verhalten im Fehlerfall. Sag mir, was du durch die echte Implementierung behandeln musstest, was der Platzhalter nicht behandelt hat."*



Nehmen Sie es mit ins Lab

Das praktische Lab am Ende dieses Moduls verlangt dieselben Ergebnisse, die Sie gerade vom Assistenten haben erstellen lassen. Nutzen Sie, was Sie hier haben, als Gerüst für Ihre Antwort und als Beleg dafür – nicht als die Antwort selbst. Bewertet wird im Lab Ihre Argumentation, und dieser Teil muss von Ihnen stammen.