Die Aufgabe

EPU - Backoffice

FACHHOCHSCHULE
TECHNIKUM WIEN

Aufgabenstellung

- EPU - Backoffice
- Ein EPU möchte seine Administrativen Tätigkeiten in einer Software abbilden.

Zu verwalten sind

- Kunden
- Kontakte
- Angebote
- Projekte
- Ausgangsrechnungen
- Eingangsrechnungen
- Bankkonto (Buchungszeilen)
- Zeiterfassung (Aggregiert)

Anforderungen

- Folgende Anforderungen wurden nach einem Beratungsgespräch festgelegt:

- Ein Kunde kann mehrere Angebote erhalten, ein Angebot wird immer nur einem Kunden gelegt
- Ein Angebot hat eine Angebotssumme, Dauer, Datum sowie die Chance der Umsetzung
- Mehrere Angebote können in ein Projekt münden, ein Angebot kann jedoch nur einem Projekt zugeordnet werden
- Am Ende und/oder während des Projektes können Ausgangsrechnungen gestellt werden. Diese sind zu Drucken

Anforderungen (II)

- Ausgangsrechnungen haben Rechnungszeilen
- Jede Rechnungszeile ist mit einem Angebot zu verknüpfen
- Jede Ausgangsrechnung ist mit genau einem Kunden verknüpft
- Jede Eingangsrechnung wird mittels Scanner in ein elektronisches Archiv importiert
- Jede Eingangsrechnung ist mit einem Kontakt verknüpft

Anforderungen (III)

- Jede Rechnung (Ein- und Ausgang) ist mit ein oder mehreren Buchungszeilen am Bankkonto verknüpft
- Manche Buchungszeilen sind mit ein oder mehreren Rechnungen (Ein- und Ausgang) verknüpft
- Jede Buchungszeile hat ein oder mehrere Kategorie (Einnahme, Ausgabe, Steuer, SVA, etc.)
- Jede Buchungszeile kann gesplittet werden (Ust und Rechnungssumme trennen)
- Jedem Projekt werden in aggregierter Form Zeiten zugebucht

Anforderungen (IV)

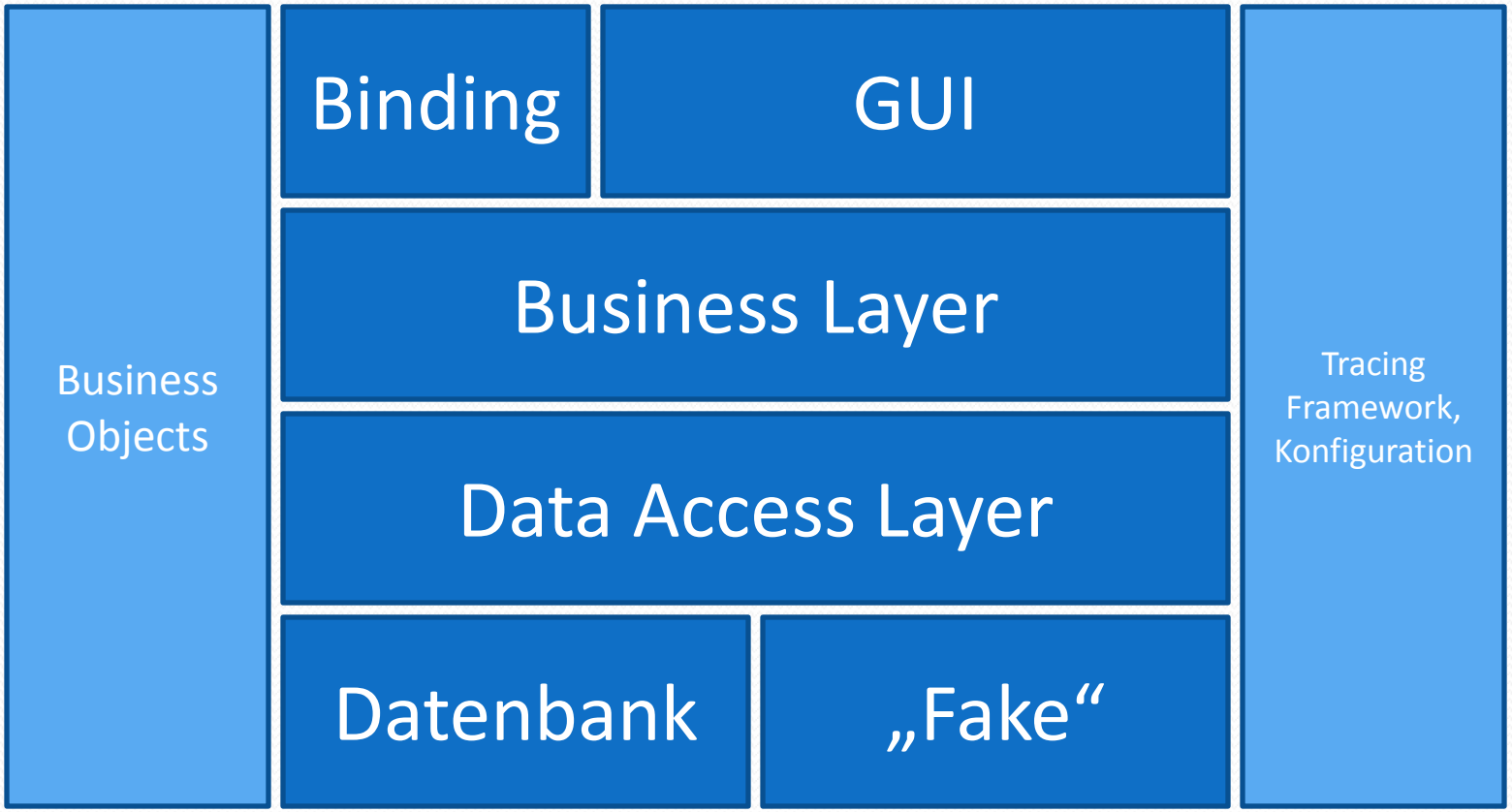
- Auswertungen

- Wie hoch ist der prognostizierte Jahresumsatz aufgrund der Angebote
- Wie hoch sind die aktuellen Einnahmen/Ausgaben nach Monat
- Wie viele Rechnungen sind offen
- Wie viele Projekte sind lt. Angebot offen
- Wie hoch ist der Stundensatz je Projekt und gesamt

Anforderungen (V)

- Grafische Oberfläche
  - WinForms & Swing supported, andere auf eigene Verantwortung
- Saubere Trennung der Layer
  - GUI
  - Business Layer
  - Data Access Layer
- Dynamischer DAL
  - einen „Fake“/Mock DAL
  - Ein „echter“ DAL gegen eine Datenbank Ihrer Wahl
- Databinding Framework
- Tracing Framework
- Konfiguration

Architektur, Layer



Architektur

- Klassische 2-Tier Architektur
- Datenbank implementier die komplette Business Logic
- Business Logic am Client
  - übernimmt die Steuerung der Datenbankabfragen (Stored Procedure calls)
  - Kümmert sich um‘s Polling (asynchron)
  - Steuert die UI
  - Führt Vorprüfungen durch (z.B. Keine Aktionen wenn man nicht an der Reihe ist)

Weitere Anforderungen

- Unit Tests (JUnit/NUnit)
- CodeCoverage Analyse
  - Report ist Abzugeben
- CodeStyle Analyse (CheckStyle, FxCop oder Gendarme)
  - Report ist Abzugeben

Bewertung

- In Summe 20 Punkte
- In der letzten Übung: „Präsentation“
  - Lauffähiger Code (.exe, .jar)
  - SourceCode

Bewertung

Bewertung	Punkte	Max. Punkte
Anfroderungen		
Kunden- und Kontakteverwaltung		1
Angebotsverwaltung		1
Projektverwaltung		1
Rechnungsverwaltung (inkl. PDF)		1
Bankkonto		1
Anbindung an Zeiterfassung		1
Angebotsreport [Prognose] (als PDF)		1
Ein- Ausgaben Report (als PDF)		1
Rechnungsreport (als PDF)		1
Nichtfunktionale Anforderungen		
Saubere Trennung der Layer		2
Dynamischer DAL		2
Databinding Framework		1
Tracing Framework		1
Konfiguration		1
50 Unittests		4
Abzüge		
Readme fehlt		1
Sonstiges 1		
Sonstiges 2		
Summen	**0**	**20**
Gruppennote	**5**	

Readme.txt

- Ausgedruckt 1 ca. A4 Seite lang
- Wenn sie fehlt -> 1 Punkt Abzug
- Inhalt
  - Benutzerhandbuch – Wie wird die Applikation verwendet
  - Lösungsbeschreibung – Wie wurde die Aufgabe gelöst
  - Worauf bin ich stolz
  - Was würde ich das nächste mal anders machen
  - Reports (CodeCoverage, etc)

Aber es gibt noch keine Datenbank!?

- Macht nichts, die kann man simulieren!
- Einfach im „Fake“ DAL Objekte zurück geben
- bzw. kann man im „Fake“ DAL Objekte auch im Hauptspeicher verwalten

Gruppenarbeit

- Max. 1-2 Leute