****

**Programmieren II - Finale Aufgabe**

Entwicklung und Umsetzung einer Banking-App

**Projekt Team:** Aaron Lacks, Luca Trumpfheller, Jens Krug, Julius Elk, Gideon Bieryt, Luc Forster

**Donzent:** Jessica Goertz

Bearbeitungszeitraum                21.05.2021 - 21.06.2021

Kurs                            TINF20IT2

**Aufgabenstellung**

Schreiben Sie eine Bankanwendung „BankingApp“:

Bank Kunde:

• Login/Logout

• Benutzerdaten pflegen (Name, Geburtsdatum, Adresse, E-Mail, Telefonnummer, ...) - d.h. anlegen, ändern, löschen

• Konto anlegen (z.B. Girokonto, Festgeldkonto, Kreditkarte, Depot, ...), Konto ändern, Konto auflösen, Zahlungseingang, Zahlungsausgang, Überweisung auf anderes Konto, Prüfungen wie Konto nicht gedeckt, Dispo anzeigen.

Bänker:

• Login/Logout

• Report über alle zugehörigen Kunden, alle Konten, Report über alle Kunden mit einem negativen Kontostand

• Freigabe von neuen Konten, bestimmen ob und wie weit der Kunde ins Minus darf

• Das Programm muss eine UI-Schnittstelle haben mit Java AWT/ Swing.

• Das Programm soll Intuitive Bedienbar sein.

• Das Programm muss seine Daten dauerhaft abspeichern (z.B. Persistenz mit JDBC). D.h. wenn neue Benutzer angelegt wurden und neue Konten – dann müssen diese Daten nach Neustart im Programm verfügbar sein.

• Überlegen Sie sich ein passendes, realistische Klassen-/Objektmodell (z.B. Konto, Girokonto, Festgeldkonto, Kreditkarte, Depot, ...)

• Leiten Sie sinnvoll ab, verteilen Sie die Attribute und Methoden so, dass die Objektorientierung einen Sinn ergibt

• Es muss eine sinnvolle Vererbungshierarchie erkennbar sein. Arbeiten Sie ggf. mit abstrakten Klassen

• Verwenden Sie passende Attribute (z.B. BIC, IBAN, eMail, Name, ...) und funktionsreiche Methoden

• Setzen Sie ggf. Polymorphie ein

• Denken Sie an Datenkapselung

• Denken Sie an die Fehlerbehandlung (Eingabe, Konvertierungen, Überläufe, ...)

• Formatieren Sie  im Team konsistent, Wählen Sie sinnvolle Bezeichner (entscheiden Sie sich für eine Sprache)

Team-Setup und Projektplanung

• Überlegen Sie wie Sie an das Projekt herangehen wollen

• Überlegen Sie welche Tätigkeiten/Rollen im Projekt gebraucht werden

• Überlegen Sie sich ein Vorgehensmodell und einen Zeitplan

• Legen Sie fest wer was bis wann macht

• Beginnen Sie mit dem Datenmodell

• Abstrahieren Sie ein Klassenmodell

• Attribute, Vererbung, Methoden

• Dokumentieren Sie die Ergebnisse

Abgabeinhalte:

• Als Zip-Datei mit Namen „Banking App <Gruppen-Id>.zip“

• Jeglichen Source Code und komplett lauffähiges Eclipse-Projekt

• Packen Sie das Projekt in ausführbare „jar“-Files. Die „jar“-Files sollen auf der Konsole ausführbar sein : java -jar BankingApp ...

• Komplette Dokumentation inkl. Anleitung, wie das Programm zu installieren, auszuführen und zu bedienen ist (z.B. Pfade, Login-Daten wie Benutzer, PW, ..., Demodaten, ...), Design Dokumente, Informationen zu externer Software wie z.B. Datenbanken, Web-Server usw.

• Klassendiagramm

• Daran denken initial auch ein System mit Daten zu liefern.

• Video als Demonstration aller Funktionalität KEINEN SOURCE CODE. Demonstration von fehlerhafter Eingabe durch den Benutzer

**Team-Setup und Projektplanung**

Aufteilung in 2er und 3er Teams

Geplantes Vorgehen

1. Aufgabenstellung analysieren
2. Requirements festhalten
3. Software planen
4. Implementierung
5. Testen / Anpassen

Allgemein to do:

Visio besorgen oder Alternative

Nächster Termin Freitag der 28. 05 15:15 Uhr Planung abschließen

Samstag 29.05 Start der Implementation

**Use-Case Diagramm**

To do:

übertragen in Visio sobald vorhanden     (Aaron)

**Komponentendiagramm**

To do: in Visio erstellen                (Gruppe)

**Klassendiagramm**

To do: in Visio erstellen                (Gruppe)

**Software**

Ergibt sich aus Klassendiagramm

**Datenbank**

To do:

ER Modell erstellen - Visio             (Gruppe, Entwurf bis Freitag: Luc)

1. Login, 2. Benutzerdaten und Konten

3.(Backup)

Datenbanken in XAAMP fertigstellen

**GUI**

To do:

Logo erstellen                         (Jens, Aaron) done

Entwurf erstellen (Jens)

GUI implementieren