

ATM Dokumentation

Die Panzerknacker

Die Panzerknacker

© 2022 Die Panzerknacker

Inhaltsverzeichnis

1. Anforderungsdokumentation	3
1.1 Produktvision und Produktziele	3
1.2 Rollen und Personas	3
1.3 User Stories	3
1.4 Aufgaben	4
1.5 Begriffslexikon	4
1.6 Mengengerüst	4
1.7 Use Cases	4
2. Codedokumentation	5
2.1 Code Ist-Dokumentation	5
2.2 Delta-Dokumentation	8
3. Die Panzerknacker	9

1. Anforderungsdokumentation

1.1 Produktvision und Produktziele

1.1.1 Produktvision

Eine regionale Bank hat unser externes Software-Entwicklerteam für einen Auftrag eingestellt. Bei dem uns übertragenem Projekt handelt es sich um die fehlerhafte Software einer ATM (Automated Teller Machine) zu deutsch Bankautomat. Der bereits existente Programmcode wurde von einem externen Unternehmen entwickelt, so dass der Kunde kein Expertenwissen zum Programm verfügt, außerdem fehlt auch die Dokumentation vollständig.

Um dem Bankunternehmen nun die Verwendung des Systems zu ermöglichen, muss das Programm komplett überarbeitet werden, darüber hinaus soll eine detaillierte Dokumentation (vollständig in deutsch) für die Bank erstellt werden. Das Fehlerfreie Programm mit den bereits integrierten Features und einer strukturierten Dokumentation ist unser Basisfaktor. Das Programm ist für die Bankautomaten der Bank in Deutschland vorgesehen. Die Dokumentation soll die Entwicklung sowie die Funktionen der Software zusammenfassen und den zuständigen Mitarbeiter verständlich machen.

1.1.2 Produktziele

Die Aufgabe unseres Teams ist es den bereits vorhandenen Code so zu überarbeiten, dass dieser voll funktionsfähig ist und eine sichere Laufzeit gewährleistet werden kann. Zur Entwicklung der Software ist eine vollständig deutsche Dokumentation vorgesehen mit **Anforderungs-, Architektur-, Test-, Abnahme-, Benutzer-, Projekt-, und Codedokumentation.**

1.2 Rollen und Personas

1.2.1 Rollen

Team Mitglied	Rolle	Vertiefungsthema
Julian Thiele	Lead	UML/Kollaborations-Werkzeug
Juri Kaemper	Coach	Test und Qualitäts Sicherung
Felix Möhler	-	Requirements Engineering
Michel Franz	-	User Experience
Christian Andrés	-	Programmierung

1.2.2 Personas

coming soon

1.3 User Stories

Als [Rolle] möchte ich [Ziel/Wunsch], um [Nutzen]

Als Bankkunde möchte ich verschiedene Geldbeträge eingeben, um diese abzuheben. Als Bankkunde möchte ich sehen, wie viel Geld auf meinem Konto ist, um zu wissen, wieviel ich noch abheben kann. Als Bankkunde möchte ich einen vierstelligen Pin zu meiner Karte eingeben müssen, um zu wissen, dass mein Bankkonto sicher ist. Als Bankkunde möchte ich eine Stückelung auswählen können, um gewünschte Scheine zu erhalten.

Als Bankmitarbeiter möchte ich mich einloggen können, um geleistete Transfäre zu sehen.

Als It-Mitarbeiter der Bank möchte ich eine vollständige und detaillierte Dokumentation, um im Fehlerfall schnell handeln zu können.

1.4 Aufgaben

coming soon

1.5 Begriffslexikon

Begriff	Bedeutung
Cash Dispenser	Bargeld im ATM-Dispenser
Deposit Slot	Geldfach zum Ein- und Auszahlen
Balance	Ist-Saldo auf einem Account
Withdrawal	Geld abheben
Account Pin	Geheimpin eines Accounts (unique)
Account number	Nummer eines Accounts (unique)
Credit	Gutschrift
Debit	Lastschrift

1.6 Mengengerüst

Mengengerüst Stichpunkte:

- Pinlänge: Immer genau 4 Stellen
- Maximaler Betrag pro Account an einem Tag abheben: 1000€?
- Maximale Nutzer gleichzeitig: ???
- Maximale Transaktionen pro Minute: ???
- Maximal registrierte Nutzer: ???

1.7 Use Cases

coming soon

2. Codedokumentation

2.1 Code Ist-Dokumentation

2.1.1 Klassen

ATMCaseStudy.java

- Erstellt eine ATM Instanz und startet diese, wenn noch keine vorhanden

ATM.java

- Stellt die Hauptklasse des ATMs dar
- Initialisiert UI mit Keypad, CashDispenser, DepositSlot und Bankdatabase
- Es gibt viele unbenutzte konstante int Variablen
- Sobald Enter betätigt wird, wird die PIN überprüft (login)
- Wenn man eingeloggt ist, wird das Menü angezeigt, wenn man als Admin eingeloggt ist, wird das Admin-Menü angezeigt
- Im Menü kann man nun zwischen Funktionen wählen:
- `balance` : Eigenes Guthaben anzeigen
- `withdrawal` : Geld abheben, indem man die Scheine einzeln wählt
- `deposit` : Geld einzahlen. Geld ist erst verfügbar, wenn überprüft.
- `exit` : Führt Login erneut aus, öffnet allerdings neues Fenster
- Sollte man als Admin angemeldet sein, öffnet sich die Adminoberfläche mit diesen Funktionen:
- Kontostand jedes Nutzers einsehen
- Zwischen Accounts wechseln
- Accounts löschen
- Neue Accounts hinzufügen

Transaction.java

- Abstrakte Klasse, die mit einer AccountNummer, Dem Screen-Objekt und dem BankDatabase-Objekt initialisiert wird.

BalanceInquiry.java

- Erbt von Transactions und überschreibt die Execute-Funktion
- Die Execute-Funktion gibt den Kontostand auf dem Screen aus

Withdrawal.java

- Erbt von Transactions und überschreibt die Execute-Funktion
- Die Execute-Funktion zeigt die Buttons zur Scheinauswahl an
- Die Transaction-Funktion ermöglicht das abheben von Geld, wenn noch genügend auf dem Konto und im CashDispenser verfügbar ist.
- Man kann nur in 20er Scheinen abheben

Deposit.java

- Erbt von Transactions und überschreibt die Execute-Funktion
- Die Execute-Funktion zeigt UI zum Geldeinzahlen an
- Beim Geldeinzahlen wird geprüft, ob das Geld eingezahlt wurde

`DepositSlot.java`

- Klasse ist nicht vorhanden.
- Hier sollte überprüft werden, ob das Geld vorhanden ist

`CashDispenser.java`

- Startet mit 500 20\$ Scheinen

`BankDatabase.java`

- Initialisiert alle Accounts
- Authentifiziert Nutzer anhand der PIN
- Funktionen um anhand der AccountNumber Daten über den Account abzurufen (verfügbares Guthaben, etc)
- Besitzt Funktionen um Guthaben von Accounts abzuziehen oder aufzuladen
- Fehler: `getaccpin` funktioniert nicht
- Funktion um temporär einen Account zu erstellen und dem Account-Array hinzuzufügen
- Funktion um temporär einen Account zu löschen

`Account.java`

- Besitzt Eigenschaften eines Benutzers
- Funktion um Pin mit aktuellem Account zu verifizieren
- Getter und Setter

`AccountFactory.java`

- Wird nicht verwendet
- Erbt von Account, initialisiert einen Account

`Iterator`

- Interface, das zwei Funktionen beinhaltet, die einen Wahrheitswert zurückgeben, ob von der aktuellen Position ein nächstes oder vorheriges Element existiert
- Funktion, die ein Objekt zurück gibt, anhand einer Position

`AccountIterator.java`

- Implementiert das Iterator Interface und überschreibt dessen Funktionen

`Screen.java`

- JFrame-Komponente, die Textfelder, Labels und Buttons besitzt
- Besitzt Funktionen um Nachrichten in der Konsole auszugeben
- Besitzt Funktionen um UI-Elemente anzuzeigen:
 - Login
 - Menü
 - Kontostand
 - Geldauszahlung
 - Geldeinzahlung
 - Admin-Ansicht

Keypad.java

- Besitzt unbenutzte Scanner-Funktion
- Besitzt JButtons für ein Tastenfeld mit Löschen und Enter Funktionen
- Funktion um ein JPanel mit Buttons zu initialisieren und zurückgeben
- Fehler: Endlos-Schleife `userinput()`

2.1.2 Verbesserungen

- Pin-Authentifizierung beim Login
- Login mit Accountnummer?
- Länge des PIN`s?
- "Exit" schließt das Programm
- Verbesserung des Event-Handlings mit Actionlisteners
- Verbesserung der Klassenstruktur (weniger Klassen?)
- Verbesserung des UI-Handlings mit JFrame und JPanel
- Verbesserung der CashDispenser-Funktion
- Bankautomat in Deutsch oder Englisch
- Lizenz nur 1x erwähnen

2.2 Delta-Dokumentation

comming soon

3. Die Panzerknacker

Wir sind die Panzerknacker.

Mitglied	Vertiefungsthema
Michél Franz	UX
Juri Kaemper	Text & QS
Christian Lopéz	Programmierung
Felix Möhler	Requirements Engineering
Julian Thiele	UML/Kollab.-Werkzeug, Entwicklungsumgebung