# Technical Documentation: Vereinskassen-System

7157747, Gellien, 8425470, Heidusch

30. Januar 2025

## 1 Analysis

### 1.1 Project Requirements

Development of a virtual club treasury system with the following key features:

- User authentication system with different roles
- Department account managemen
- Transaction handling and history tracking
- Balance monitoring and prevention of negative balances
- Data persistence using CSV files

### 1.2 System Architecture

#### 1.2.1 Model-View-Controller (MVC) Pattern

The system implements the MVC pattern:

- Models: User, Account, Transaction, UserRole
- Views: MainView, LoginView, AdminView, TreasurerView, FinanceView
- Controllers: AuthController, TransactionController

### 1.3 Data Structure

#### 1.3.1 CSV File Structure

- users.csv: username, password, role, department
- accounts.csv: accountid, department, balance, treasurer
- transactions.csv: transactionid, timestamp, amount, type, source, target, description

# 2 Implementation

## 2.1 Key Classes

- **DataHandler**: Manages data persistence and retrieval
- **User**: Handles user authentication and permissions
- **Account**: Manages account balances and transactions
- **Transaction**: Records financial transactions

## 2.2 Controllers

- **AuthController**: Manages user authentication and authorization
- **TransactionController**: Handles financial transactions and balance updates

## 2.3 Views

- **MainView**: Primary application window
- **LoginView**: User authentication interface
- **AdminView**: Administrator management interface
- **TreasurerView**: Department account management interface
- **FinanceView**: Financial overview interface

# 3 Testing

## 3.1 Test Cases

### 3.1.1 Authentication Tests (TestAuthController)

- **test_check_password**
  - Purpose: Verify password verification functionality
  - Test Data: User "hanz"with password "1234"
  - Expected Result: Password check returns True
  - Status: Passed
- **test_login**
  - Purpose: Verify login process and role assignment
  - Test Data: User "hanz"with admin role
  - Expected Result: Login returns correct user role
  - Status: Passed

### 3.1.2 Data Handling Tests (TestDataHandler)

- **Data Creation Tests**
  - **test_add_user**
    - Purpose: Verify user creation and storage
    - Test Data: Admin user "hanz"
    - Expected Result: User correctly stored in users list
    - Status: Passed
  - **test_add_account**
    - Purpose: Verify account creation and storage
    - Test Data: Account äbc"with balance 0.4
    - Expected Result: Account correctly stored in accounts list
    - Status: Passed
  - **test_add_transaction**
    - Purpose: Verify transaction creation and storage
    - Test Data: Transaction between two accounts
    - Expected Result: Transaction correctly stored in transactions list
    - Status: Passed
- **Data Retrieval Tests**
  - **test_load_user**
    - Purpose: Verify user retrieval functionality
    - Test Data: User franz"
    - Expected Result: Correct user object returned
    - Status: Passed
  - **test_load_account**
    - Purpose: Verify account retrieval functionality
    - Test Data: Account äbc"
    - Expected Result: Correct account object returned
    - Status: Passed
  - **test_load_transaction**
    - Purpose: Verify transaction retrieval functionality
    - Test Data: Transaction äb"
    - Expected Result: Correct transaction object returned
    - Status: Passed
- **Data Persistence Test**

- **test_export_import**
  - Purpose: Verify CSV export and import functionality
  - Process: Export data, clear memory, import data
  - Expected Result: All data correctly persisted and retrieved
  - Status: Passed

### 3.2 Test Environment

- Test files located in tests/files/
- Separate CSV files for test data
- Clean file state maintained between tests

### 3.3 Test Data

Test files located in tests/files/:

- usersfile.csv
- accountsfile.csv
- transactionsfile.csv

## 4 Setup and Dependencies

- Python 3.12
- Tkinter for GUI
- Setup via setup.py
- Project structure follows standard Python package layout