## **1. Einleitung**

## **1.1 Ziel des Projekts**

Das Ziel dieses Projekts ist die Entwicklung eines sicheren und benutzerfreundlichen Finanzverwaltungssystems. Das System ermöglicht Benutzern:

* Sich sicher anzumelden,
* Ihre Kontoeinstellungen zu verwalten (z. B. Passwortänderung),
* Finanztransaktionen durchzuführen,
* und diese in einer Historie nachverfolgen können.

Ein besonderer Fokus liegt auf der Implementierung moderner Sicherheitsmechanismen wie **risikobasierter Authentifizierung** und **Zwei-Faktor-Authentifizierung (2FA)**.

## **1.2 Motivation**

Die Motivation hinter diesem Projekt ist es, ein System zu entwickeln, das den aktuellen Sicherheitsanforderungen entspricht und gleichzeitig eine gute Benutzererfahrung bietet. Viele bestehende Systeme sind entweder unsicher oder zu komplex für Endbenutzer. Dieses Projekt versucht, eine Balance zwischen Sicherheit und Benutzerfreundlichkeit zu finden.

## **1.3 Zielgruppe**

Die Zielgruppe des Systems sind:

* Endbenutzer, die ihre Finanzen verwalten möchten,
* Administratoren, die das System warten und verwalten.

## **2. Systemarchitektur**

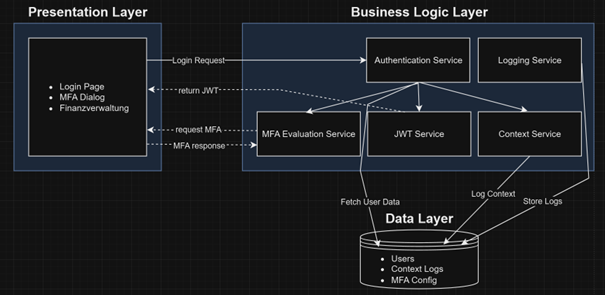
## **2.1 Überblick**

Das System basiert auf einer **Client-Server-Architektur**, bei der das Backend mit Spring Boot entwickelt wurde und das Frontend HTML, CSS, JavaScript sowie Thymeleaf verwendet.

## **2.2 Technologien**

| **Komponente** | **Technologie** | **Beschreibung** |
| --- | --- | --- |
| Backend | Kotlin, Spring Boot | RESTful API für Geschäftslogik |
| Frontend | Thymeleaf, JavaScript | Benutzeroberfläche mit serverseitigem Rendering |
| Sicherheit | Spring Security | Authentifizierungs- und Autorisierungsmechanismen |
| Datenbank | PostgreSQL | Speicherung der Benutzerdaten, Transaktionen und Login\_Events |

## **2.3 Architekturdiagramm**

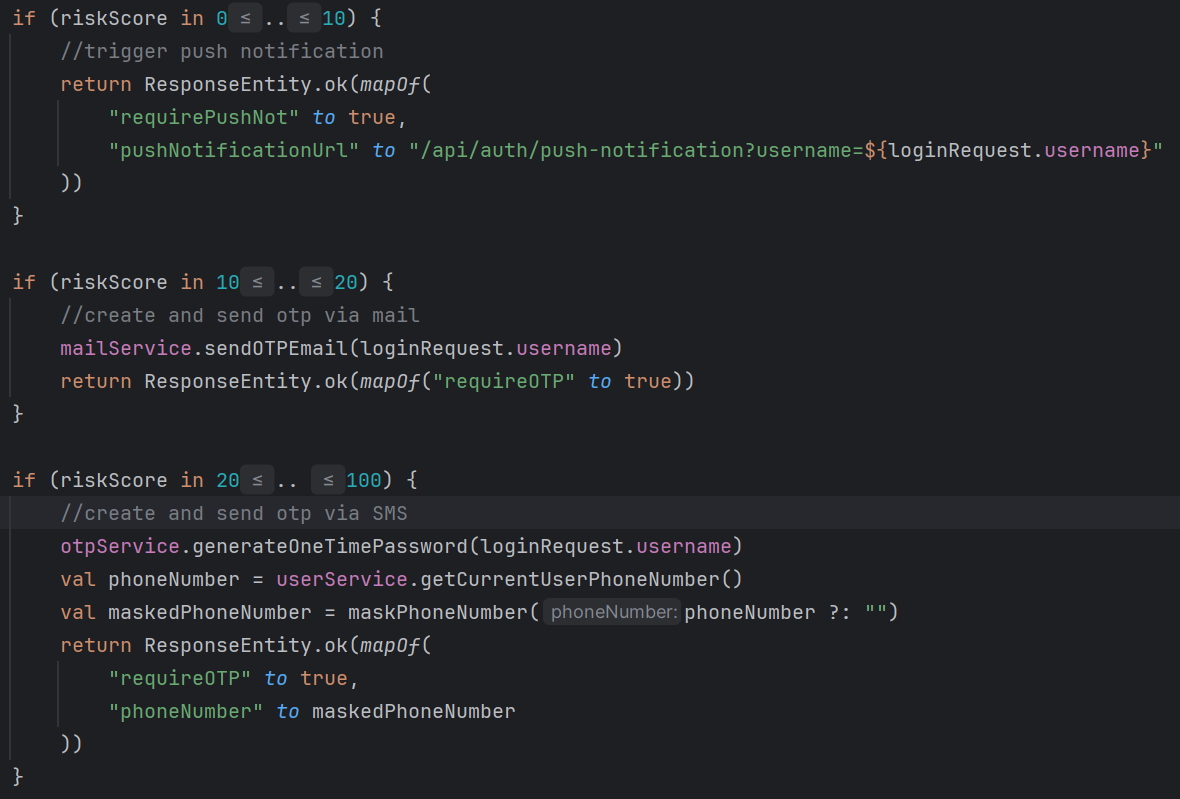


## **3. Hauptfunktionen**

## **3.1 Risikobasierte Benutzerauthentifizierung**

Die Authentifizierung basiert auf einem risikobasierten Ansatz:

* **Risikobewertung:** Ein Algorithmus berechnet einen Risiko-Score basierend auf Benutzerverhalten.
* **Dynamische Authentifizierungsmethoden:**
  + Score < 10: Push-Benachrichtigung.
  + Score 10–20: E-Mail OTP.
  + Score > 20: SMS OTP.

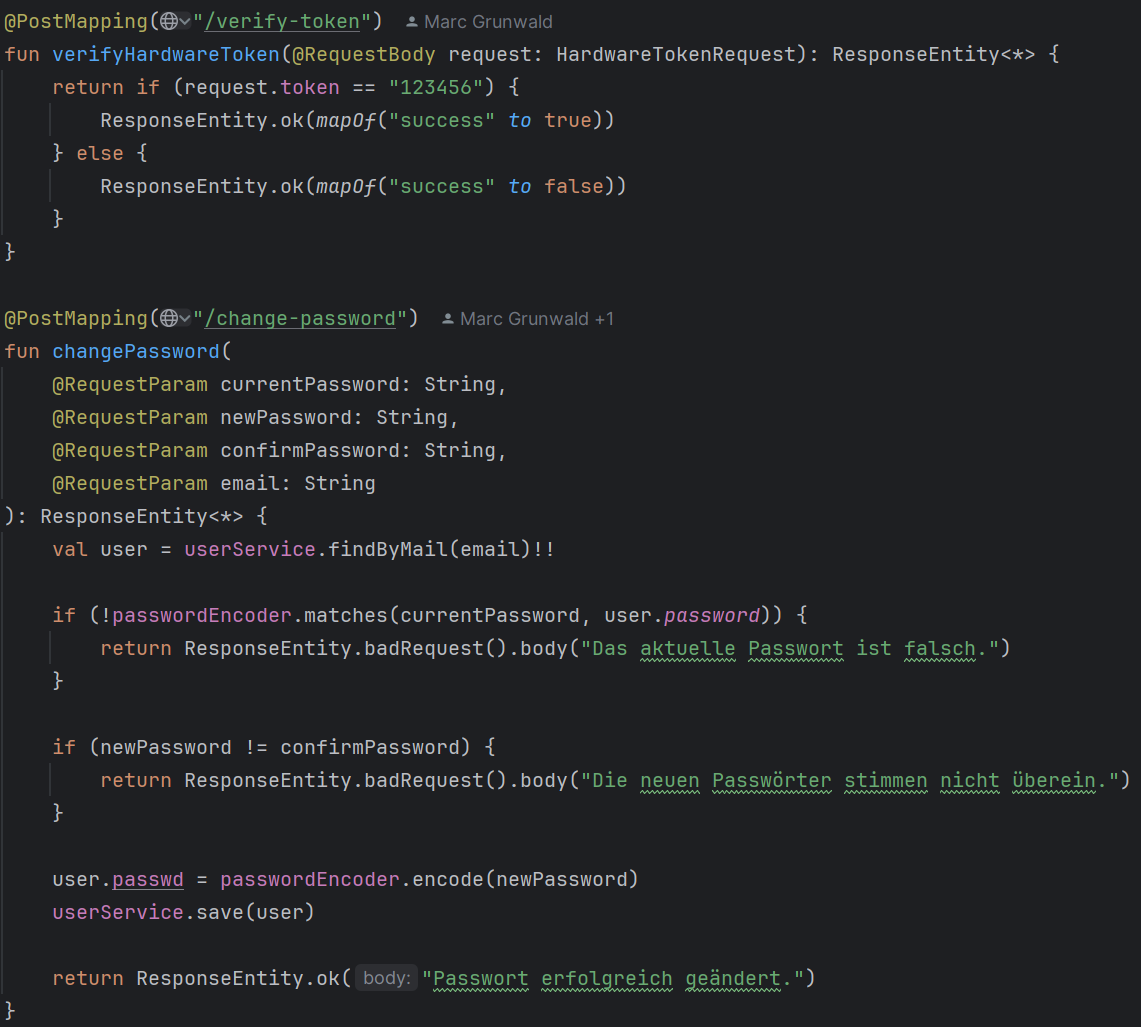


## **3.2 Kontoeinstellungen**

Benutzer können ihre Kontoeinstellungen (z. B. Passwort) ändern. Dabei wird ein **Hardware-Token** simuliert, um zusätzliche Sicherheit zu gewährleisten.

## **Workflow:**

1. Benutzer füllt das Formular zur Passwortänderung aus.
2. System fordert den Benutzer auf, seinen Hardware-Token einzugeben.
3. Nach erfolgreicher Verifizierung wird das Passwort geändert.



## **4. Sicherheitskonzept**

## **4.1 Risiko-basierte Authentifizierung**

Die Authentifizierungsmethode wird basierend auf einem Risiko-Score ausgewählt:

* **Niedriges Risiko (<10):** Push-Benachrichtigung.
* **Mittleres Risiko (10–20):** E-Mail OTP.
* **Hohes Risiko (>20):** SMS OTP.

## **Vorteile:**

* Reduziert unnötige Sicherheitsmaßnahmen für vertrauenswürdige Anmeldungen.
* Erhöht die Sicherheit bei verdächtigen Aktivitäten.

## **4.2 Multi-Faktor-Authentifizierung (MFA)**

* Für verschiedene Szenarien wird MFA verwendet:

1. Anmelden von einem neuen Gerät: SMS/E-Mail OTP
2. Hohe Transaktion: Biometrische Authentifizierung
3. Kontoänderungen: Hardware-Token

## **Workflow:**

1. Benutzer gibt Anmeldedaten ein.
2. System entscheidet anhand des Risiko-Scores über die MFA-Methode.
3. Benutzer führt zusätzliche Verifizierung durch.

## **4.3 JWT-basierte Authentifizierung**

* Verwendung von JWTs für zustandslose Authentifizierung.
* Struktur des JWT:
  + Header: Algorithmus und Typ
  + Payload: Benutzerinformationen
  + Signature: Gehashte Kombination aus Header, Payload und Secret

## **4.4 Dynamisches Zugriffskontrollsystem**

* Analyse des Benutzerverhaltens (z.B. IP-Adresse, Standort).
* Anpassung der Sicherheitsmaßnahmen basierend auf dem Risiko.

## **4.5 Passwort-Sicherheit**

* Verwendung von PasswordEncoder für sicheres Hashing.
* Verifizierung des aktuellen Passworts vor Änderungen.

**5. Persistence-Kompontenten**

**5.1 User**

**Funktionen:**

* Registrierung und Anmeldung in der Applikation
* Tätigen einer Überweisung
* Fähigkeit seine Daten (Passwort) anzupassen

**Komponenten:**

* **User\_Id**: Eindeutige Identifizierung des Benutzers
* **mail**: E-Mail Adresse des Benutzers (erforderlich zur Authentifizierung)
* **password**: Passwort des Benutzers (erforderlich zur Authentifizierung)
* **role**: Rolle des Benutzers
* **enabled**: Status des Benutzers, ob dieser freigeschaltet ist oder nicht
* **budget**: Das derzeitige Budget des Benutzers
* **one\_time\_password**: One Time Passwort des Benutzers, falls erforderlich
* **otp\_request\_time**: Zeitpunkt der Anfrage des OTPs
* **otp\_expiry\_date**: Zeitpunkt des Ablaufs des OTPs (20 Minuten)
* **phone\_number**: Telefonnummer des Benutzers

**5.2 Transaktion**

**Funktionen:**

* Speichern aller Transaktionen, die zwischen Benutzern stattfinden
* Erforderlich, um die Historie von Transaktionen nachvollziehen zu können

**Komponenten:**

* **id**: Eindeutige Identifizierung einer Transaktion
* **sender\_id**: Eindeutige Id des sendenden Benutzers
* **recipient\_id**: Eindeutige Id des empfangenen Benutzers
* **amount**: Betrag, der überwiesen wird
* **timestamp**: Zeitpunkt der Überweisung

**5.3 Login Events**

**Funktionen:**

* Berechnung des Risk-Scores

**Komponenten:**

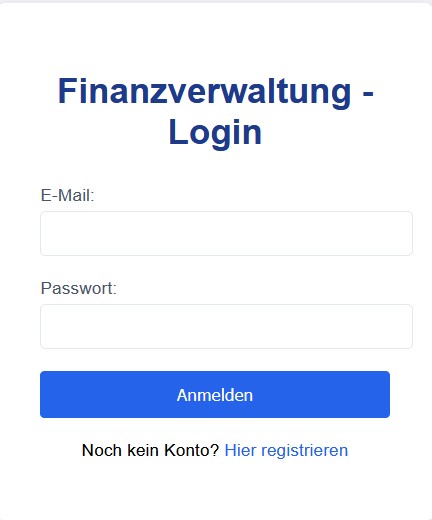
* **id**: Eindeutige Identifizierung eines Login Events
* **user\_id**: Verweis auf den Benutzer des Login Events
* **mail**: E-Mail Adresse des Benutzers
* **login\_time**: Zeitpunkt des Logins
* **ip\_adress**: IP-Adresse des Benutzers
* **location**: Standort des Benutzers
* **browser**: Browser des Benutzers während des Login Prozesses
* **browser\_version**: Derzeitige Browserversion
* **operating\_system**: Betriebssystem des Benutzers
* **status**: Status des Benutzers, ob dessen Authentifizierung erfolgreich war (Ja/Nein)

## **6. Frontend-Komponenten**

## **6.1 Login-Seite**

## **Funktionen:**

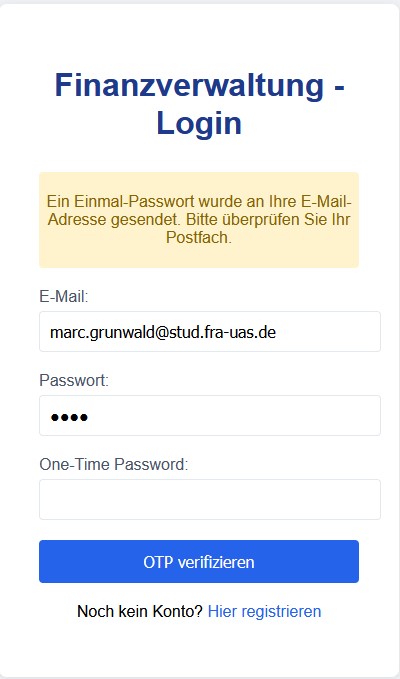
* Eingabefelder für E-Mail und Passwort.
* Dynamische Anzeige von OTP-Feldern basierend auf der Serverantwort.

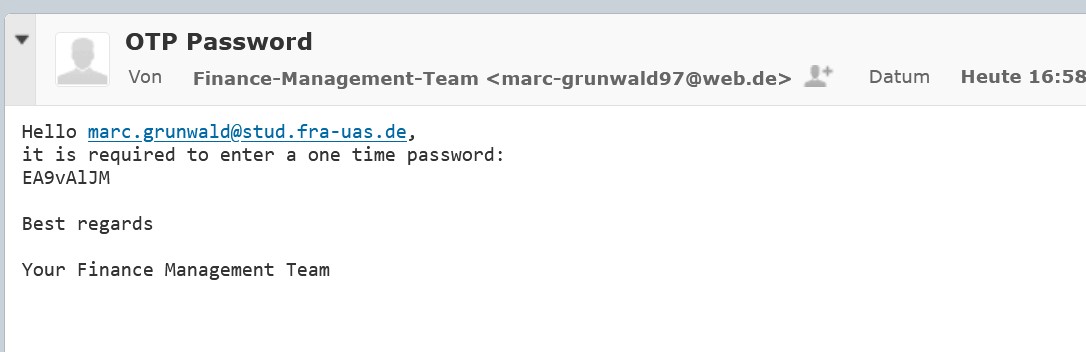


**6.1.1 Login-Seite mit Anforderung einer Push-Bestätgigung**

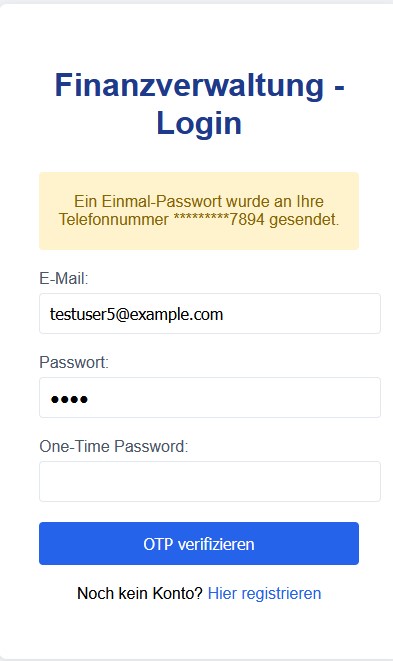


**6.1.2 Login-Seite mit Anforderung eines E-Mail OTP**





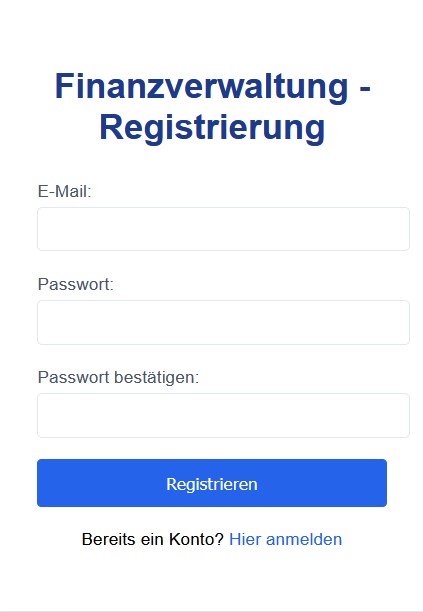
**6.1.3 Login-Seite mit Anforderung eines SMS OTP**



**6.2 Registierungs-Seite**

**Funktionen:**

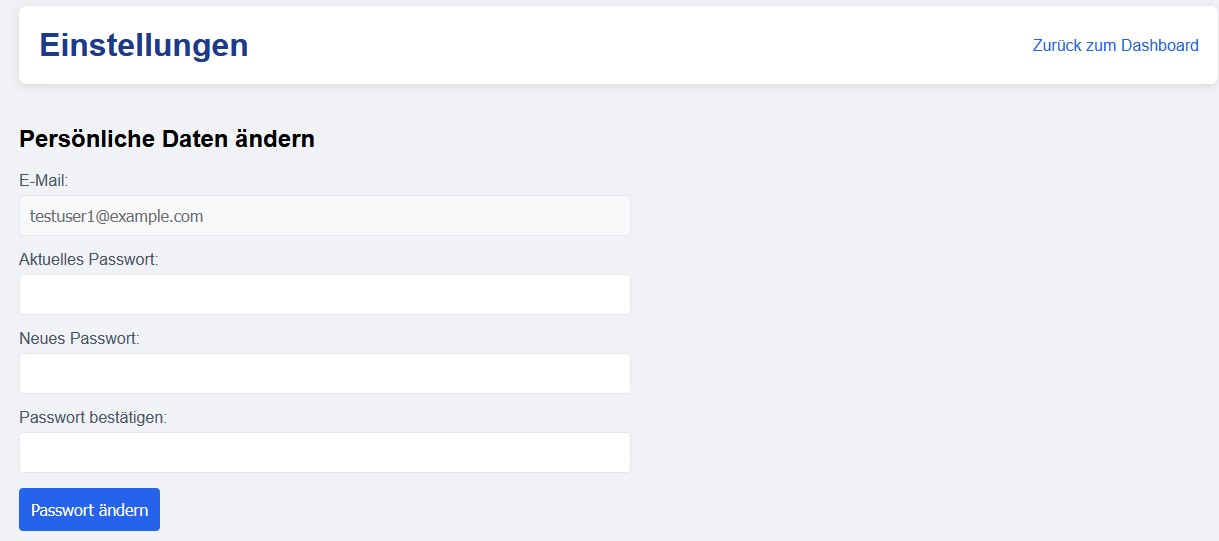
* Registrierung eines neuen Benutzers durch Angabe seiner E-Mailadresse und eines Passworts

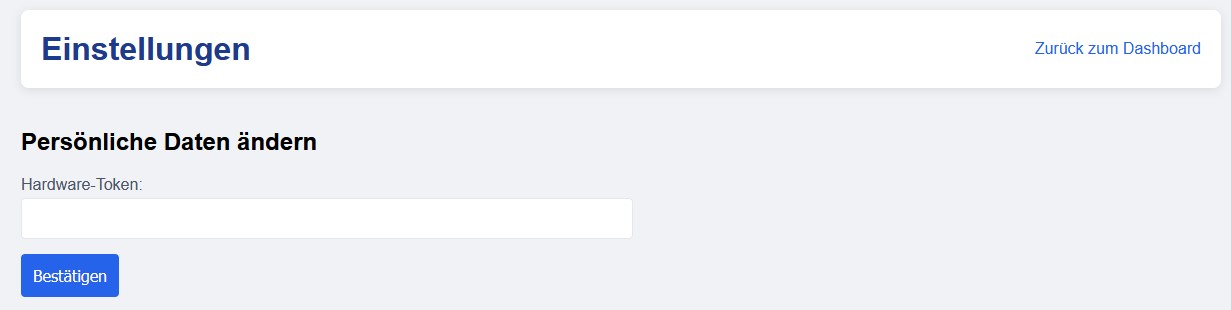


## **6.3 Einstellungsseite**

## **Funktionen:**

* Formular zur Änderung des Passworts.
* Eingabefeld für Hardware-Token bei sicherheitsrelevanten Aktionen.

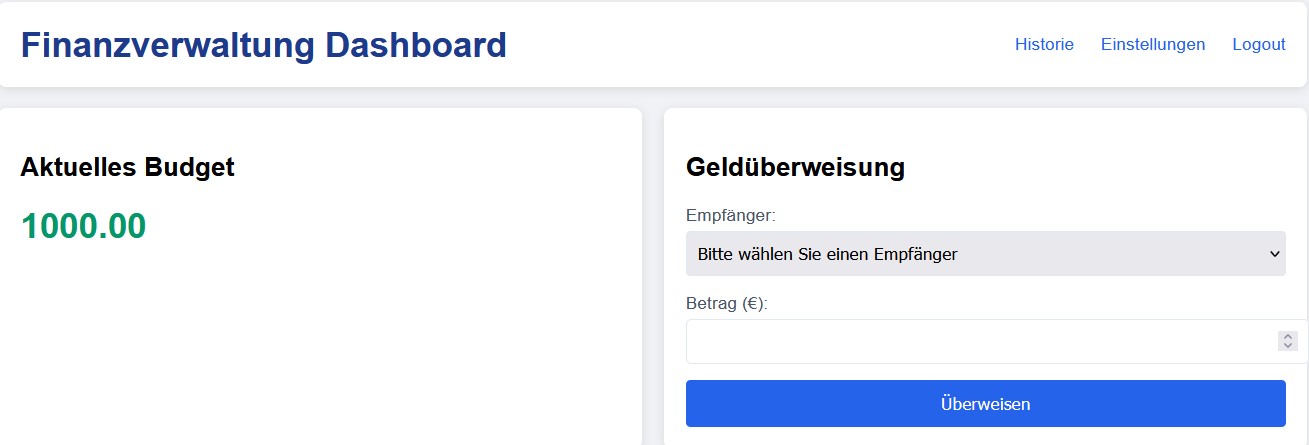




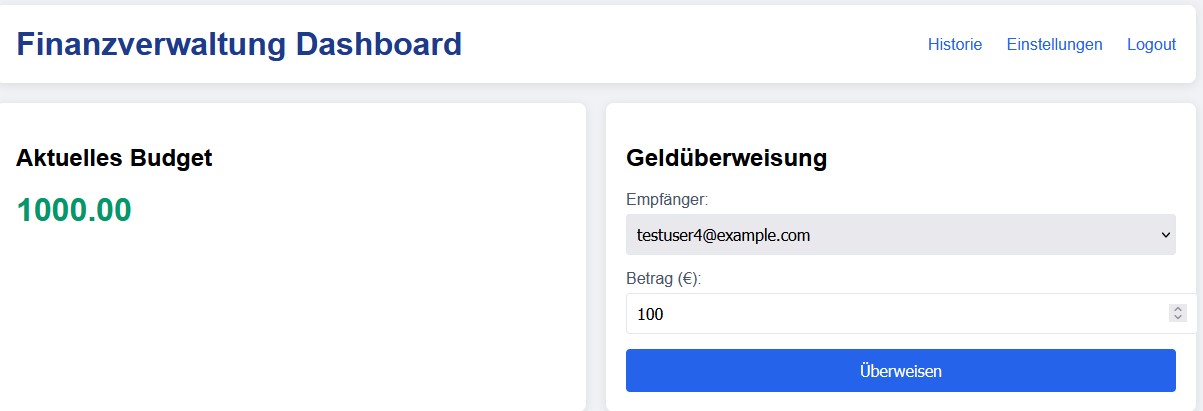
**6.4 Dashboard-Seite**

**Funktionen:**

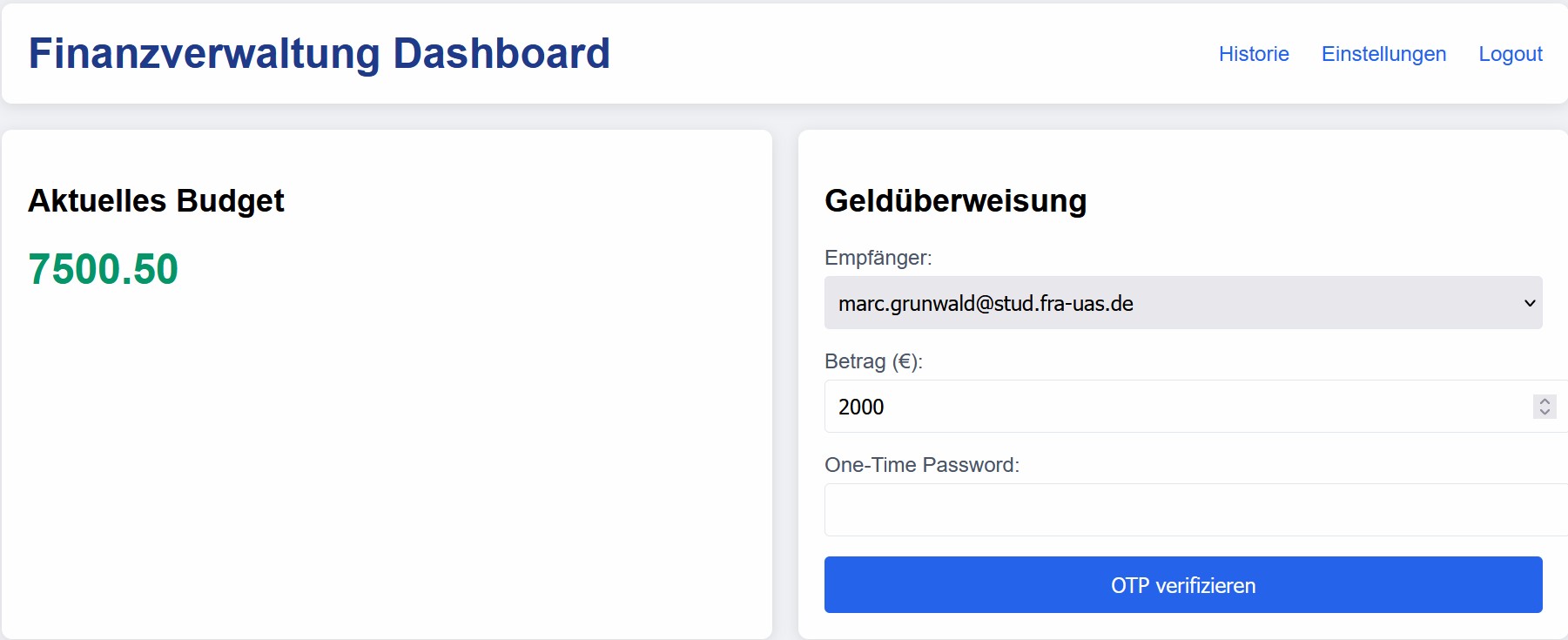
* Formular zur Überweisung an einen Benutzer



* Überweisung eines Betrags an einen Benutzer



* Überweisung eines hohen Betrags an einen Benutzer mit anschließender OTP-Verifizierung



**6.5. Historie**

**Funktionen:**

* Übersicht der Transaktionshistorie bestehend aus:
  + Datum
  + Sender und Empfänger
  + Betrag

