Personal Finance Tracker

Project description

The personal finance tracker is an application you can use to manage your bank accounts. Users are able to log in, create multiple bank accounts and even share it with others, but the owner has the most authority over the bank account. The application allows users to make transactions to others. A person can set up budget goals so that application can give advice: the tracker will tell the person how much money he still needs in order to reach his goal, or how much he is allowed to spend. The user also has the ability to look through all of his expenses/income and sort them by certain amounts, names or dates. Additionally, there are bank administrators that can manage bank accounts and delete users to keep the application safe.
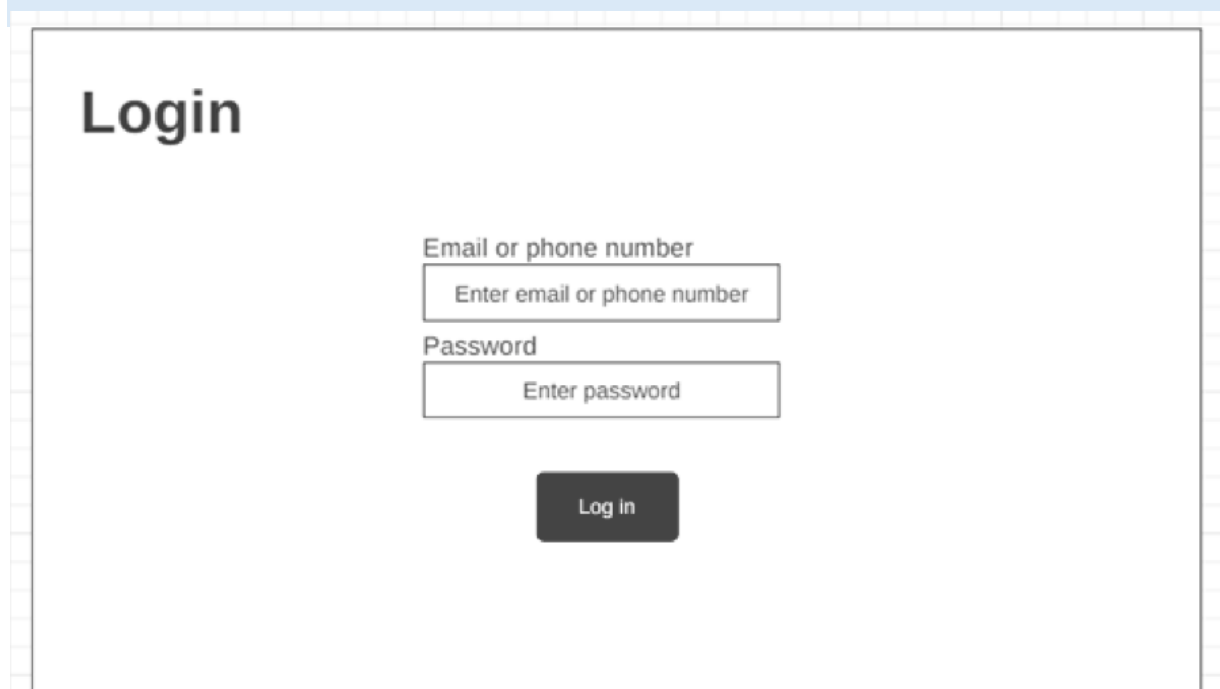
User stories

As a user,

I'm able to log into my account,

So that I can access my bank accounts.

WIREFRAME

ACCEPTANCE CRITERIA

- User can log in with his credentials.
- Login validation: username (email address/phone number) and password have to be correct, if one of them is wrong send a warning.
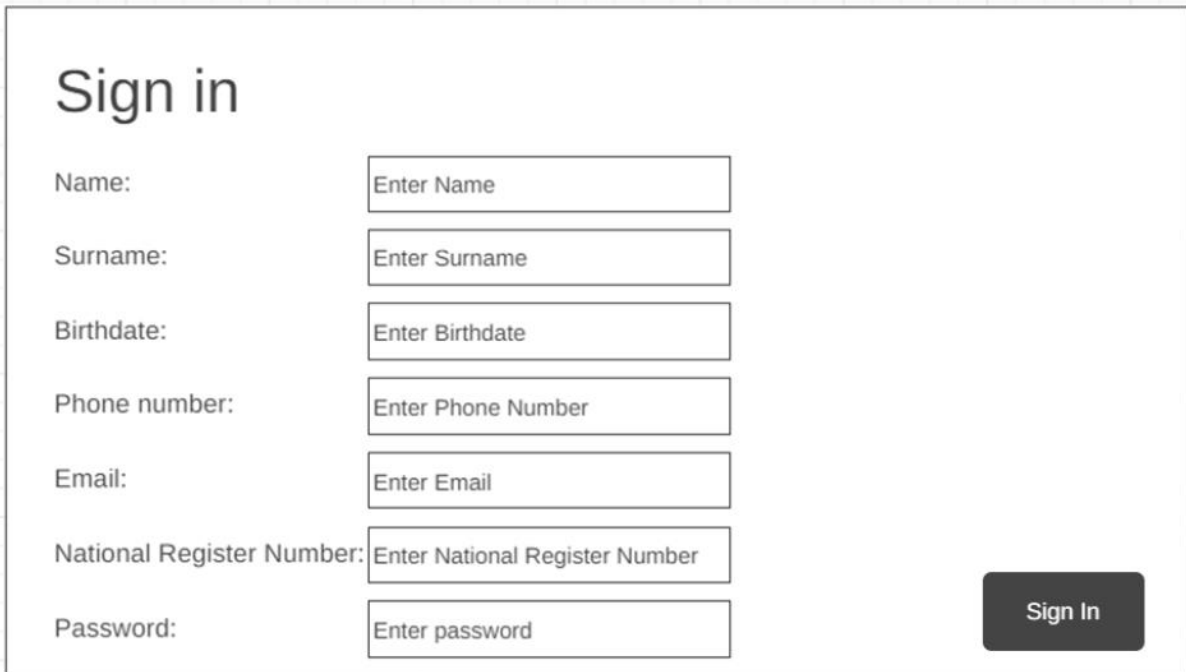- After a successful login the user is brought to the main page.

STORY – CREATE ACCOUNT

As a new client,

I'm able to create an account,

So that I can create and manage my back accounts.

WIREFRAME

ACCEPTANCE CRITERIA

- User has to enter his name, surname, birthdate, phone number, email address and national register number.
- Only one account can be created with phone number/email address/national register number.
- User can choose his own password.
- Validation on user information and password.
- User can immediately log in with his email address/phone number and password after creating an account.
- Password needs to be at least 8 characters long, with a number, an uppercase letter, a lowercase letter and a special character.

STORY – UPDATE ACCOUNT

As a user,

I'm able to update my account information,

So that user information is correct.

WIREFRAME

ACCEPTANCE CRITERIA

- A user can change account information.
- Validation: phone number and email address are all unique, when updating information, these values remain unique.
- User cannot change his name, birth date (age) and national register number.
- After updating information, application automatically changes user information from all accounts that have the user in their contacts.

STORY - DELETE ACCOUNT

As a user,

I'm able to delete my account,

So that everything is deleted.

WIREFRAME

Settings

Delete account

Enter password to confirm deletion | Enter Password

Confirm

ACCEPTANCE CRITERIA

- A user can delete his account, when deleting it the user has to enter his password.
- An account can only be deleted if it only has closed bank accounts or no bank accounts.
- After deleting the account, the application will delete information about the user in all contacts that have the user in their contacts.

STORY – DELETE ACCOUNT V2

As a bank administrator,

I'm able to delete accounts,

So that fraudulent accounts are deleted.

WIREFRAME

Settings

Delete user account   [Enter the user email address]

(Optional) Tell the user why you are deleting his account

[Note]

[Confirm]

ACCEPTANCE CRITERIA

- A bank administrator can delete accounts, using the email address of the user.
- Validation on email address, if there isn't an account with the email send an error.
- An account can only be deleted if it only has closed bank accounts or no bank accounts.
- After deleting the account, the application will delete information about the user in all contacts that have the user in their contacts.
- An administrator can add a note to tell the user why his account has been deleted.

STORY – CREATE BANK ACCOUNT

As a user,

I'm able to create a new bank account,

So that I can manage multiple accounts.

WIREFRAME

Create new bank account

Bank account type   [Type ▾]

Shared bank account   ○ Yes   ○ No

Share with   [Contact ▾]

Name   [Name]

[Confirm]

ACCEPTANCE CRITERIA

- A user can create a new bank account and choose what type of bank account it is, select if it is shared or not
- If the bank account is shared, user can choose who he shares it with among his contacts
- When a new back account is created, status is open
- When a new bank account is created, balance is zero
- Bank account is automatically added when account is created.
- The user can add a nickname for the bank account.
- Number of bank accounts cannot exceed 8.

STORY – VIEW BANK ACCOUNT

As a user,

I'm able to see a certain bank account,

So that I can see my bank account in more detail.

WIREFRAME

Bank account name

+900	Status: Open
Transaction made by user1	Shared contacts: user1, user2, …
-500	
+100	
-200	
-50	
-200	

ACCEPTANCE CRITERIA

- User can see all his expenses and income on the bank account.
- The user can see who he shared his accounts with and can see the status of the bank account (open, blocked, frozen and closed).
- If there were transactions made on a shared bank account, the name of the person who made the transaction is visible.
- The user can click on a transaction to see more detailed information.

STORY – UPDATE BANK ACCOUNT

As a user,

I'm able update a bank account,

So that I can change certain aspects of the bank account

WIREFRAME

Bank Account Name Settings

Change contacts:

User1	-
User2	-
User3	+

Freeze bank account: ○ Freeze

Close bank account: ○ Close

Enter password

Confirm

ACCEPTANCE CRITERIA

- User can change the status of the bank account to frozen, but cannot unfreeze it.
- The owner of the bank account can remove or add people to the bank account.
- The owner can close the bank account, but the balance has to be zero.
- The application requests user to enter his password before closing the bank account.

STORY – UPDATE BANK ACCOUNT STATUS

As an administrator,

I'm able to update the status of a bank account,

So that I can help the owner when he isn't able to do it himself.

WIREFRAME

ACCEPTANCE CRITERIA

- The administrator can (un)block the bank account
- The administrator can unfreeze the bank account, if the user froze it.
- The administrator can open the bank account, if the user had closed it.

STORY – DELETE BANK ACCOUNT

As an owner,

I'm able to delete a bank account,

So that I don't have to manage it.

WIREFRAME

ACCEPTANCE CRITERIA

- The owner can delete a bank account, if the status is closed. When deleting it, the owner has to enter his password.
- After deleting the bank account, the application will remove it from every user that used this bank account.

STORY – DELETE BANK ACCOUNT V2

As an administrator,

I'm able to delete a bank account,

So that fraudulent bank accounts are deleted.

WIREFRAME

ACCEPTANCE CRITERIA

- The administrator can delete a bank account even if it still has a balance.
- The administrator can add a note to tell the user why it has been deleted.
- When the administrator deletes the bank account, the application will remove it from every user that used this bank account.

STORY – MAIN PAGE

As a user,

I'm able to see all bank accounts,

So that I can manage them on the main page.

Main Page

Bank account 1	Bank account 3
200	10000
Open	Open
Bank account 2	Bank account 4
500	Balance: 1900
Frozen	Open

ACCEPTANCE CRITERIA

- The user can see all bank accounts he has or that are shared with him.
- The name, status and balance of the bank accounts are visible on the main page.
- The user can click on the bank accounts that redirect them to a more detailed page of the bank account.
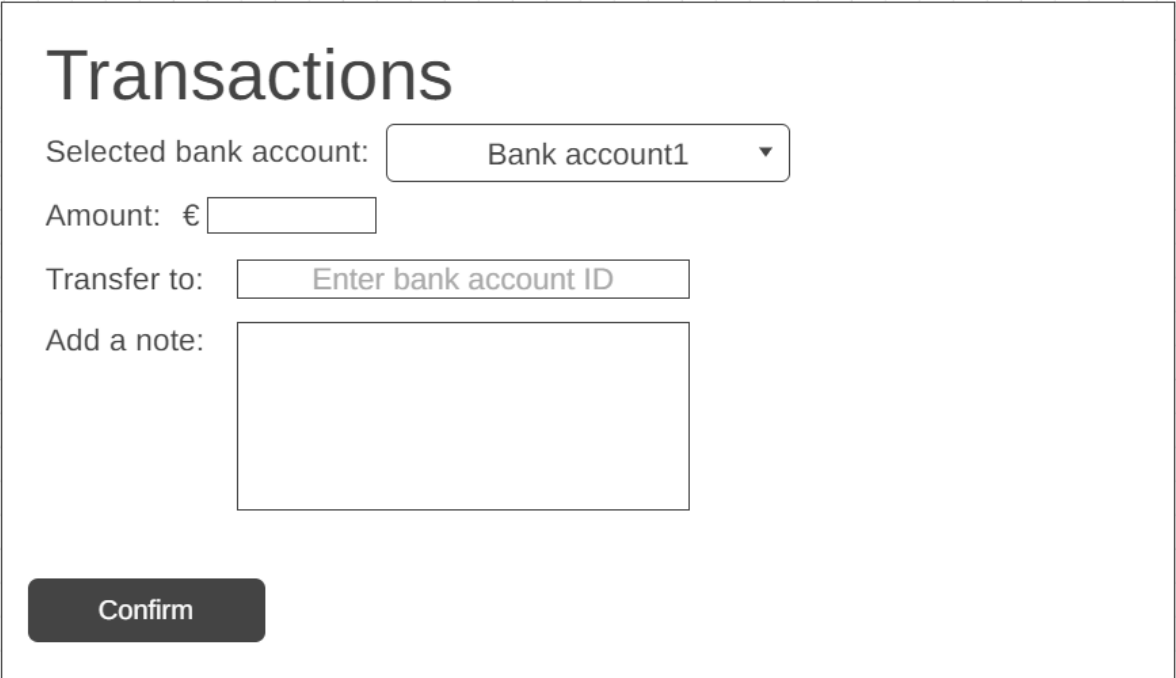
STORY – TRANSACTION PAGE

As a user,

I'm able to access the transaction page,

So that I can make transactions to other accounts

WIREFRAME

Transactions

Selected bank account: Bank account1   ▼

Amount: € [_____]

Transfer to: [Enter bank account ID]

Add a note: [_____]

[Confirm]

ACCEPTANCE CRITERIA

- When the user clicks on the transaction page, he has to select an bank account from which he wishes to make a transaction from.
- The user can enter a number in, but the number cannot be higher than the balance of this bank account or negative.
- The user can transfer money to another account that he owns or to someone else's account. The money is immediately taken from the account and transferred to the other bank account.
- The user can add a note to the transaction.
- The user cannot select a bank account that is frozen, blocked or closed.

STORY – VIEW BUDGET GOAL

As a user,

I'm able to view my budget goal,

So that I have a more deeper understanding.

WIREFRAME

Budget goal

Advice: save money

Goal: € 40 000	Current: € 30 000	8 months left	Current: Save for car
Goal: € 5 000	Achieved		Vacation
Goal: € 5 000	Not Achieved		Goal 3

ACCEPTANCE CRITERIA

- The user can see all the budget goals he made in the past.
- Advice is given to the user for his current goal. The application tells the user how much he is allowed to spend/how much he has to save.
- If the set date has passed, the status of budget goal automatically updates.

STORY – CREATE BUDGET GOAL

As a user,

I'm able to create a budget goal,

So that I can easily get advice.

WIREFRAME

New budget goal

Budget goal: €   [amount]

Deadline: [dd/mm/yyyy]

Include bank accounts:
Bank account 1 ✓
Bank account 2
Bank account 3 ✓

Budget goal name: [Name]

Budget goal description: [Description]

[Confirm]

ACCEPTANCE CRITERIA

- The user can only have one active budget goal at a time.
- Validation on budget goal and date, budget goal cannot be a negative number and date has to be in the future.
- The user can choose which bank accounts he wants to include in his budget goal. The user has to select at least one bank account.
- The user can set a name and description for his budget goal.

STORY – SORTING INCOME PAGE

As a user,

I'm able to access a sorting page of a certain bank account,

So that I can check for more detailed information.

WIREFRAME

Income of Bank account1

Filter by [amount ▼] [2400]

From user3	+ 2400	11/01/2011
From user2	+ 2400	11/01/2011
From user3	+ 2400	10/10/2010
From user5	+ 2400	01/01/2010

ACCEPTANCE CRITERIA

- The user can sort by certain dates, amounts and ID of bank accounts.
- Validation on date: cannot be in the future.
- Validation on amount: cannot be negative.
- Validation on ID: if no ID has been found return an error.
- The user can see an income overview
    o Total income overview
    o Yearly income overview
    o Monthly income overview

STORY – SORTING EXPENSE PAGE

As a user,

I'm able to access a sorting page of a certain bank account,

So that I can check for more detailed information.

WIREFRAME

Expenses of Bank account1

Filter by | user name ▼ | user3

From user3	- 100	11/01/2011
From user3	- 50	11/01/2011
From user3	- 1000	10/10/2010
From user3	- 500	01/01/2010

ACCEPTANCE CRITERIA

- The user can sort by certain dates, notes, amounts and ID of bank accounts.
- Validation on date: cannot be in the future.
- Validation on amount: cannot be negative.
- Validation on ID: if no ID has been found return an error.
- The user can see an expense overview
    - o Total expense overview
    - o Yearly expense overview
    - o Monthly expense overview

STORY – OVERVIEW CONTACTS

As a user,

I'm able to see all my contacts,

So that I can see who is in my contacts.

WIREFRAME

Contacts

user 2	Edit
user 3	Edit
user 4	Edit
user 5	Edit
+ add new contact	

ACCEPTANCE CRITERIA

- The user can see all his contacts.
- On the contacts page, the user can see the name, surname and a note of his contact.
- When clicking on a contact, a more detailed description is given of the contact.

STORY – ADD CONTACTS

As a user,

I'm able to add contacts,

So that I can easily send them transactions.

WIREFRAME

Add contact

Name: Nickname

Email address: Email address

Phone number: Phone number

User ID   ID

Relationship with contact: Family ▼

Add contact

ACCEPTANCE CRITERIA

- The user is able to add a contact to his contacts list
- The application asks for name (nickname), ID, email, phone number and relation of the contact with the user.
- Validation on email, phone number and ID: if no user with the same values has been found return an error.
- Validation on email, phone number and ID: if no value has been given, return error. There has to be a value in the correct form.

STORY – UPDATE CONTACTS

As a user,

I'm able to update contacts,

So that I can change information about them.

WIREFRAME

Edit contact information

Name: Nickname

Email address: Email address

Phone number: Phone number

Relationship with contact: Family

Confirm

ACCEPTANCE CRITERIA

- The user can edit contact information of users in his contact list.
- Validation on email, phone number and relation.
- ID cannot be edited.
- After editing the contact information is updated immediately.

STORY – DELETE CONTACTS

As a user,

I'm able to delete contacts,

So that I don't have them in my contacts.

WIREFRAME

Edit contact information

Name: [Nickname]

Email address: [Email address]

Phone number: [Phone number]

Relationship with contact: [Family ▼]

[Confirm]   [Delete Contact]

ACCEPTANCE CRITERIA

- A user can delete a contact.
- After deleting a contact, contact information is removed from his contact list.

STORY – VIEW LOANS ON BANK ACCOUNT

As a user,

I'm able to view my loans on a bank account,

So that I am informed about the loans.

WIREFRAME

ACCEPTANCE CRITERIA

- User can view all loans on bank account.
- The user can view the amount that he lend from the bank, interest rate and status.
- If the loan is still to be paid off, the user sees the next payment day and amount.

STORY – CREATE LOAN

As a user,

I'm able to request a new loan,

So that I can borrow money

WIREFRAME

ACCEPTANCE CRITERIA

- The user can request money from the bank, the user can choose the amount and type.
- Validation on amount: cannot be negative value.
- Loan request is sent to administrator, he decides whether or not to accept the request and suggest an interest rate and end date.

STORY – ACCEPT LOAN REQUEST

As an administrator,

I'm able to deny/accept loan request,

So that we make good deals.

WIREFRAME

ACCEPTANCE CRITERIA

- Administrator can see a list of requested loans, with the user who requested it, the amount and the loan type.
- The administrator can deny or accept the request, after suggesting an interest percentage.
- After the loan has been accepted, the loan is viewable in the user's loan list and the money is deposited in the user's bank account.

Account

PK	Accountnumber
	Shared
	Status
	Type
	Balance
FK	Budgetgoal_goalname
FK	Transaction_referencenumber

Budget goal

PK	Goalname
	Target_amount
	Current_amount
	Status

Transaction

PK	Referencenumber
	Date
	Amount
	Type
	Currency
	Description
	Payment_method

Loan

PK	LoanID
	Type
	Interest
	Amount
	Fee

account_has_loan

PK,FK1	LoanID
PK,FK2	Accountnumber
	From
	Till

user_owns_account

PK,FK1	Accountnumber
PK,FK2	Nationalregisternumber
	From
	Till

Bank

PK	Bankname
	Phonenumber
	Address
FK	Account_accountnumber

User

PK	Nationalregisternumber
	Name
	Phonenumber
	Birthdate
	Password
	Email

<<Transaction>>

-referenceNumber: String

-date: LocalDate

-amount: int

-currency: String

-description: String

-type: String

-account: Account

+ Transaction(referenceNumber, date, amount, currency, description, account)

+getType(): String

Expense

-destination: String

+ Expense(destination)

+ setDestination():

+ getDestination():

Income

-source: String

+ Income(source)

+ setSource():

+ getSource():

Budget goal

-goalName: String

-targetAmount: int

-currAmount: int

-isActive: boolean

-account: Account

+ Budgetgoal(goalName, targetAmout, currAmount, isActive,)

+ calculateBestMonthlySpending(): int

Account

-accountNumber: String

-balance: int

-isShared: boolean

-startDate: LocalDate

-endDate: LocalDate

-isActive: boolean

-type: String

-transaction: ArrayList<Transaction>

-users: ArrayList<User>

-loans: ArrayList<Loan>

-bank: Bank

-budgetGoals: ArrayList<Budgetgoal>

+ Account(accountNumber, isShared, balance, endDate, startDate, isActive, type, users, bank)

+ setBalance(balance): int

+ addTransaction(): Transaction

+ removeTransaction(): Transaction

User

-nationalRegisterNumber: String

-name: String

-isAdministrator: boolean

-phoneNumber: String

-password: String

-accounts: ArrayList<Account>

+ User(nationalRegisterNumber, name, isAdministrator, phoneNumber, password)

+ verifyLogin(name, password): boolean

+ blockAccount(isAdministrator, account): Account

Loan

-loanID: String

-type: String

-amount: int

-startDate: localDate

-endDate: localDate

-accounts: ArrayList<Account>

+ Loan(loanID, type, interest, amount, startDate, endDate)

+ removeAmount(): int

+ isDone(): boolean

+ calulcateFee(): int

+ calulcateInterest(): int

Bank

-bankName: String

-phoneNumber: String

-address: String

-accounts: ArrayList<Account>

+ Bank(bankName, phoneNumber, address)

+ addAccount(): Account

+ removeAccount(): Account

+ giveUserPermission(): boolean

+ removeLoanFromAccount(): Loan

+ addLoanToAccount(): Loan