Web Technologies Exam

A.A. 2019-2020

Marco Petri

marco.petri@mail.polimi.it

Gruppo 62, Esercizio 4

Analisi dei requisiti sui dati

Un'applicazione web consente la gestione di trasferimenti di denaro online da un conto a un altro. Un utente ha un nome, un codice e uno o più conti correnti. Un conto ha un codice, un saldo, e i trasferimenti fatti (in uscita) e ricevuti (in ingresso) dal conto. Un trasferimento ha una data, un importo, un conto di origine e un conto di destinazione. Quando l'utente accede all'applicazione appare una pagina LOGIN per la verifica delle credenziali. In seguito all'autenticazione dell'utente appare l'HOME page che mostra l'elenco dei suoi conti. Quando l'utente seleziona un conto, appare una pagina STATO DEL CONTO che mostra i dettagli del conto e la lista dei movimenti in entrata e in uscita, ordinati per data discendente. La pagina contiene anche una form per ordinare un trasferimento. La form contiene i campi: codice utente destinatario, codice conto destinatario, causale e importo. All'invio della form con il bottone INVIA, l'applicazione controlla che il conto di destinazione appartenga all'utente specificato e che il conto origine abbia un saldo superiore o uguale all'importo del trasferimento. In caso di mancanza di anche solo una condizione, l'applicazione mostra una pagina con un avviso di fallimento che spiega il motivo del mancato trasferimento. In caso di verifica di entrambe le condizioni, l'applicazione deduce l'importo dal conto origine, aggiunge l'importo al conto destinazione e mostra una pagina CONFERMA TRASFERIMENTO che presenta i dati del conto di origine e destinazione, con i rispettivi saldi aggiornati. L'applicazione deve garantire l'atomicità del trasferimento: ogni volta che il conto di destinazione viene addebitato il conto di origine deve essere accreditato e viceversa.

Entità: rosso
Attributi: verde
Relazioni: ciano

Un primo sguardo ai requisiti presenti nel testo porta all'osservazione della presenza di diverse entità, mostrate a fianco in colore rosso.
L'informazione sui trasferimenti può essere progettata come entità debole, questa entità è infatti strettamente dipendente dal conto di origine e dal conto di destinazione, in ogni caso più di un trasferimento potrebbe avere lo stesso conto di origine e di destinazione nella stessa data. Per questa ragione l'identificazione della chiave primaria deve essere scelta di conseguenza, scegliendo ad esempio una chiave numerica per identificare un trasferimento.

Analisi dei requisiti applicativi PURE HTML

Un'applicazione web consente la gestione di trasferimenti di denaro online da un conto a un altro. Un utente ha un nome, un codice e uno o più conti correnti. Un conto ha un codice, un saldo, e i trasferimenti fatti (in uscita) e ricevuti (in ingresso) dal conto. Un trasferimento ha una data, un importo, un conto di origine e un conto di destinazione. Quando l'utente accede all'applicazione appare una pagina LOGIN per la verifica delle credenziali. In seguito all'autenticazione dell'utente appare l'HOME page che mostra l'elenco dei suoi conti. Quando l'utente seleziona un conto, appare una pagina STATO DEL CONTO che mostra i dettagli del conto e la lista dei movimenti in entrata e in uscita, ordinati per data discendente. La pagina contiene anche una form per ordinare un trasferimento. La form contiene i campi: codice utente destinatario, codice conto destinatario, causale e importo. All'invio della form con il bottone INVIA, l'applicazione controlla che il conto di destinazione appartenga all'utente specificato e che il conto origine abbia un saldo superiore o uguale all'importo del trasferimento. In caso di mancanza di anche solo una condizione, l'applicazione mostra una pagina con un avviso di fallimento che spiega il motivo del mancato trasferimento. In caso di verifica di entrambe le condizioni, l'applicazione deduce l'importo dal conto origine, aggiunge l'importo al conto destinazione e mostra una pagina CONFERMA TRASFERIMENTO che presenta i dati del conto di origine e destinazione, con i rispettivi saldi aggiornati. L'applicazione deve garantire l'atomicità del trasferimento: ogni volta che il conto di destinazione viene addebitato il conto di origine deve essere accreditato e viceversa.

Viste: rosso
Componenti visuali: verde
Eventi: ciano
Azioni: viola

Completamento della specifica PURE HTML

Gli utenti sono caratterizzati da un codice univoco autogenerato che li identifica all'interno del sito web per trasferimenti di denaro, questo codice univoco viene utilizzato anche per effettuare il login.
Un trasferimento è caratterizzato da una data dell'anno solare e deve controllare che la data inserita dall'utente sia corretta per evitare che quest'ultimo inserisca valori non validi.
Esiste una pagina comune senza necessità di autenticazione accessibile a chiunque che permette agli utenti registrati di effettuare l'autenticazione e accedere ai contenuti riservati.
Esiste una pagina comune senza necessità di autenticazione accessibile a chiunque che permette agli non utenti registrati di effettuare la registrazione al sito web e successivamente autenticarsi.
Esiste una pagina principale (home) che mostra l'elenco dei conti posseduti dall'utente autenticato che la visualizza, l'elenco è cliccabile e al click su uno dei conti viene aperta una pagina di stato del conto.
Esiste una pagina di stato del conto che mostra dettagli del conto e lista dei movimenti, essa contiene inoltre un form per l'ordinazione di trasferimenti monetari verso altri conti che invia una richiesta POST che viene verificata lato server.
Esiste una pagina di fallimento che mostra le ragioni del fallimento dell'ordinazione del conto se questo non fosse ordinabile.
Esiste una pagina di conferma del trasferimento che riporta i dati dei conti di origine e destinazione con i rispettivi saldi aggiornati.

Analisi dei requisiti applicativi RIA

Si realizzi un'applicazione client server web che modifica le specifiche precedenti come segue:

- ✓ L'applicazione supporta registrazione e login mediante una pagina pubblica con opportune form. La registrazione controlla la validità sintattica dell'indirizzo di email e l'uguaglianza tra i campi "password" e "ripeti password", anche a lato client. La registrazione controlla l'unicità dello username.
- ✓ Dopo il login, l'intera applicazione è realizzata con un'unica pagina.
- ✓ Ogni interazione dell'utente è gestita senza ricaricare completamente la pagina, ma produce l'invocazione asincrona del server e l'eventuale modifica del contenuto da aggiornare a seguito dell'evento.
- ✓ I controlli di validità dei dati di input (ad esempio importo non nullo e maggiore di zero) devono essere realizzati anche a lato client.
- ✓ L'avviso di fallimento è realizzato mediante un messaggio nella pagina che ospita l'applicazione.
- ✓ L'applicazione chiede all'utente se vuole inserire nella propria rubrica i dati del destinatario di un trasferimento andato a buon fine non ancora presente. Se l'utente conferma, i dati sono memorizzati nella base di dati e usati per semplificare l'inserimento. Quando l'utente crea un trasferimento, l'applicazione propone mediante una funzione di auto-completamento i destinatari in rubrica il cui codice corrisponde alle lettere inserite nel campo codice destinatario.

Pagine e componenti: rosso
Entità: viola
Attributi: verde
Eventi: ciano

È possibile osservare dalla specifica della versione RIA dell'applicazione che si specifica la presenza della registrazione che invece non era presente nel caso dell'applicazione HTML pura e specifica anche che l'applicazione deve controllare l'unicità del nome scelto dall'utente.
Inoltre la versione RIA aggiunge la presenza di una relazione fra un utente ed un altro gruppo di dati (utente e un suo conto), la rubrica, che si tradurrà nello schema logico in una tabella della base di dati essendo in principio un'entità debole poiché una rubrica non esiste se non vi è un utente che ne è proprietario.

Diagramma ER del database PURE HTML

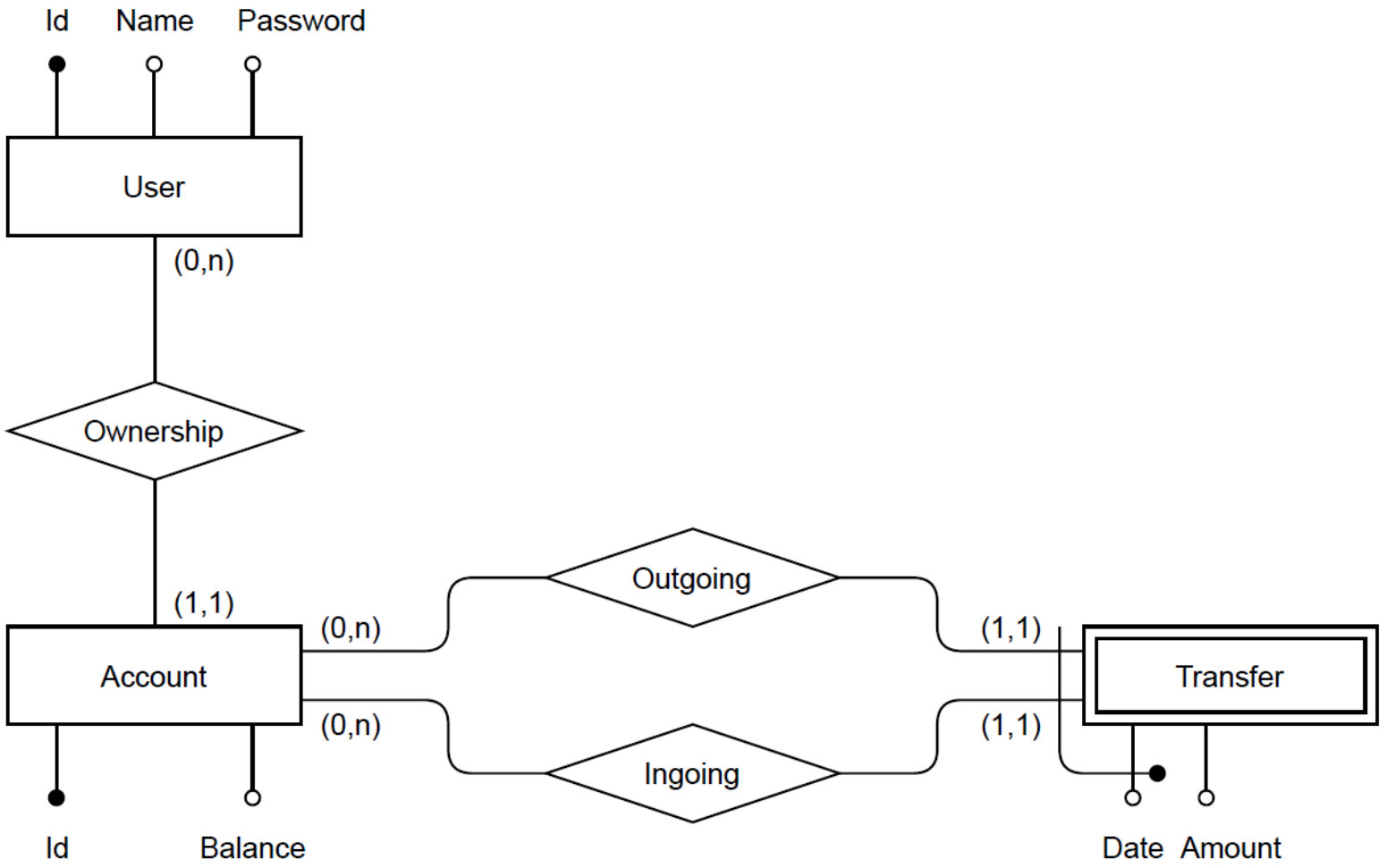
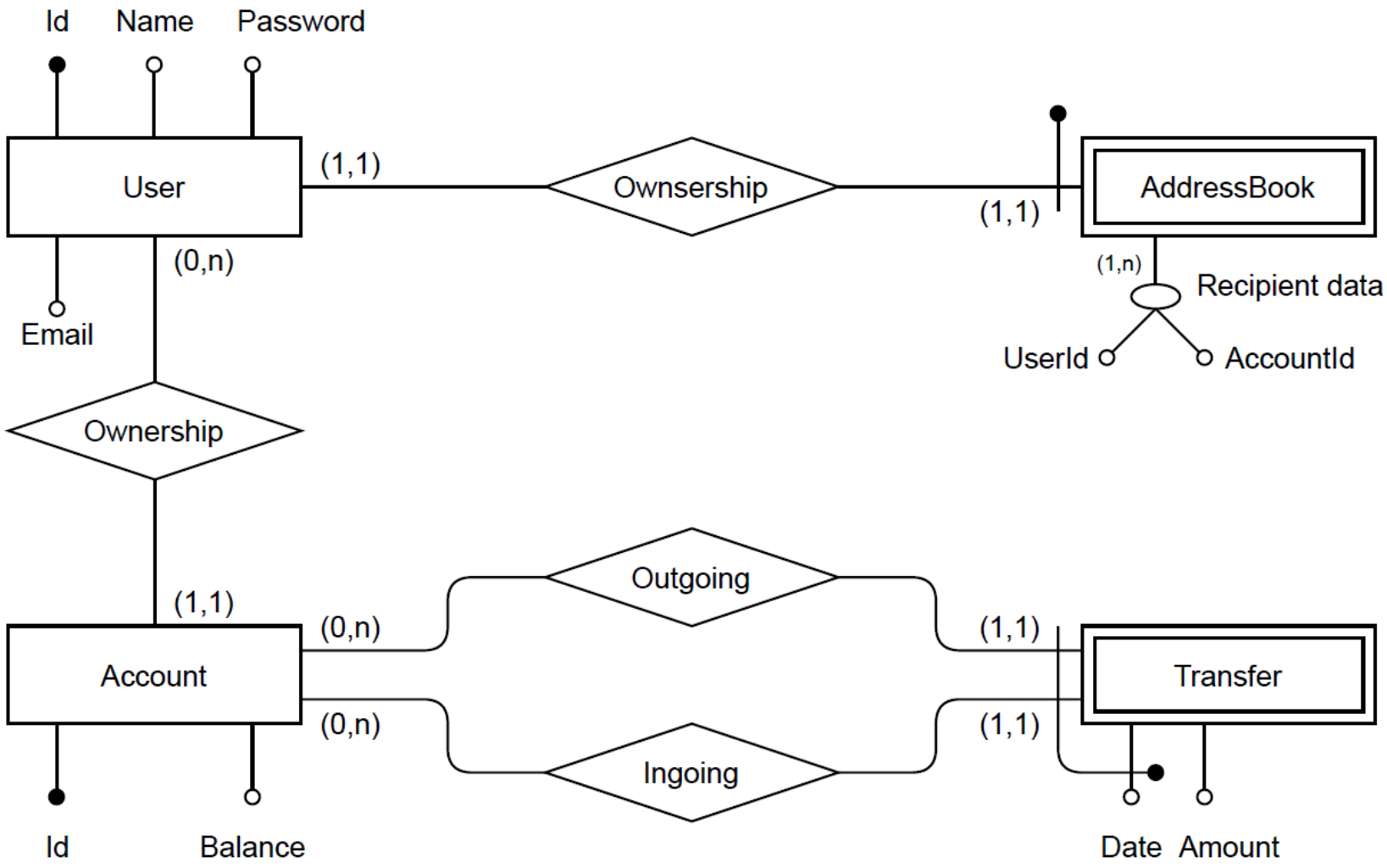
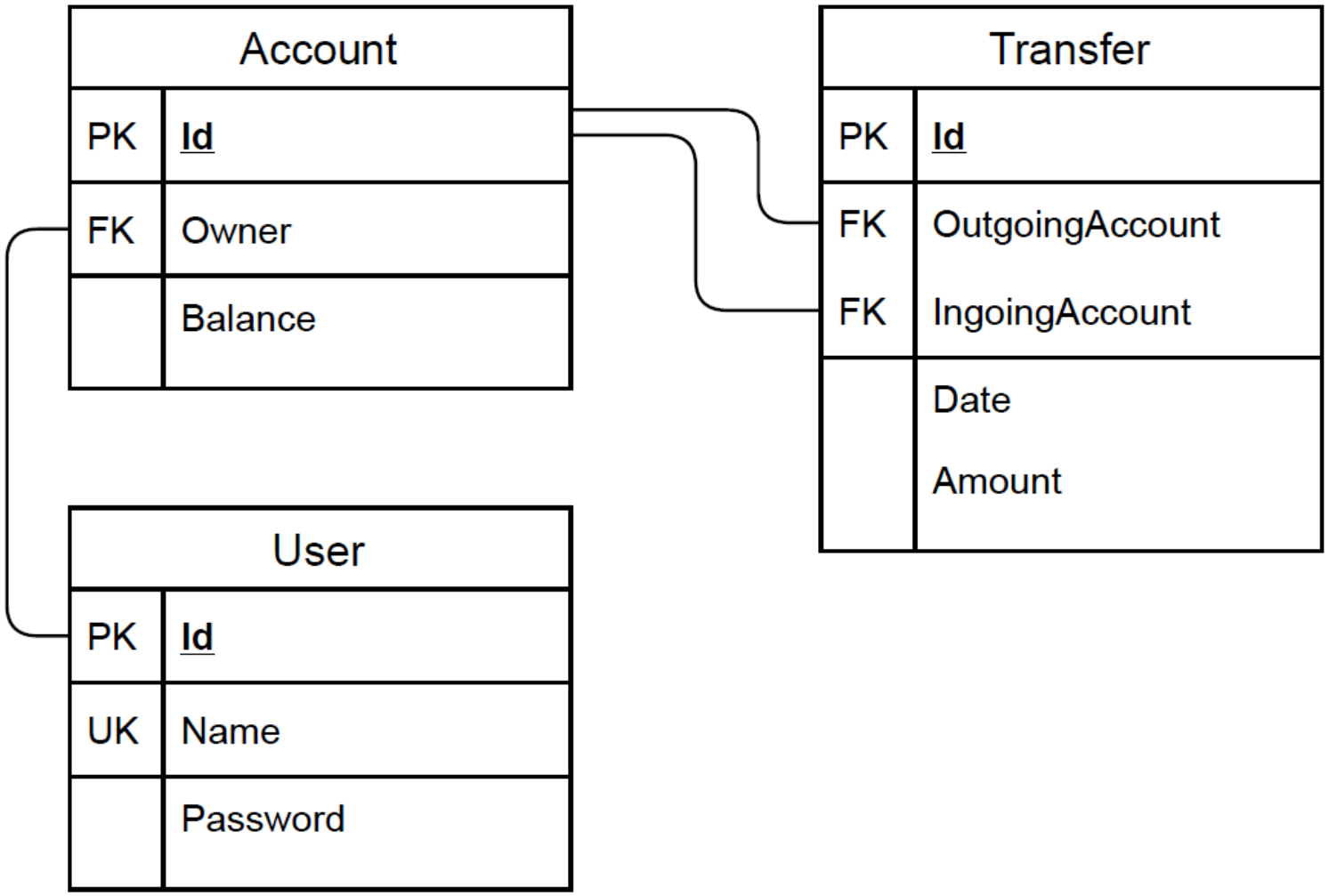


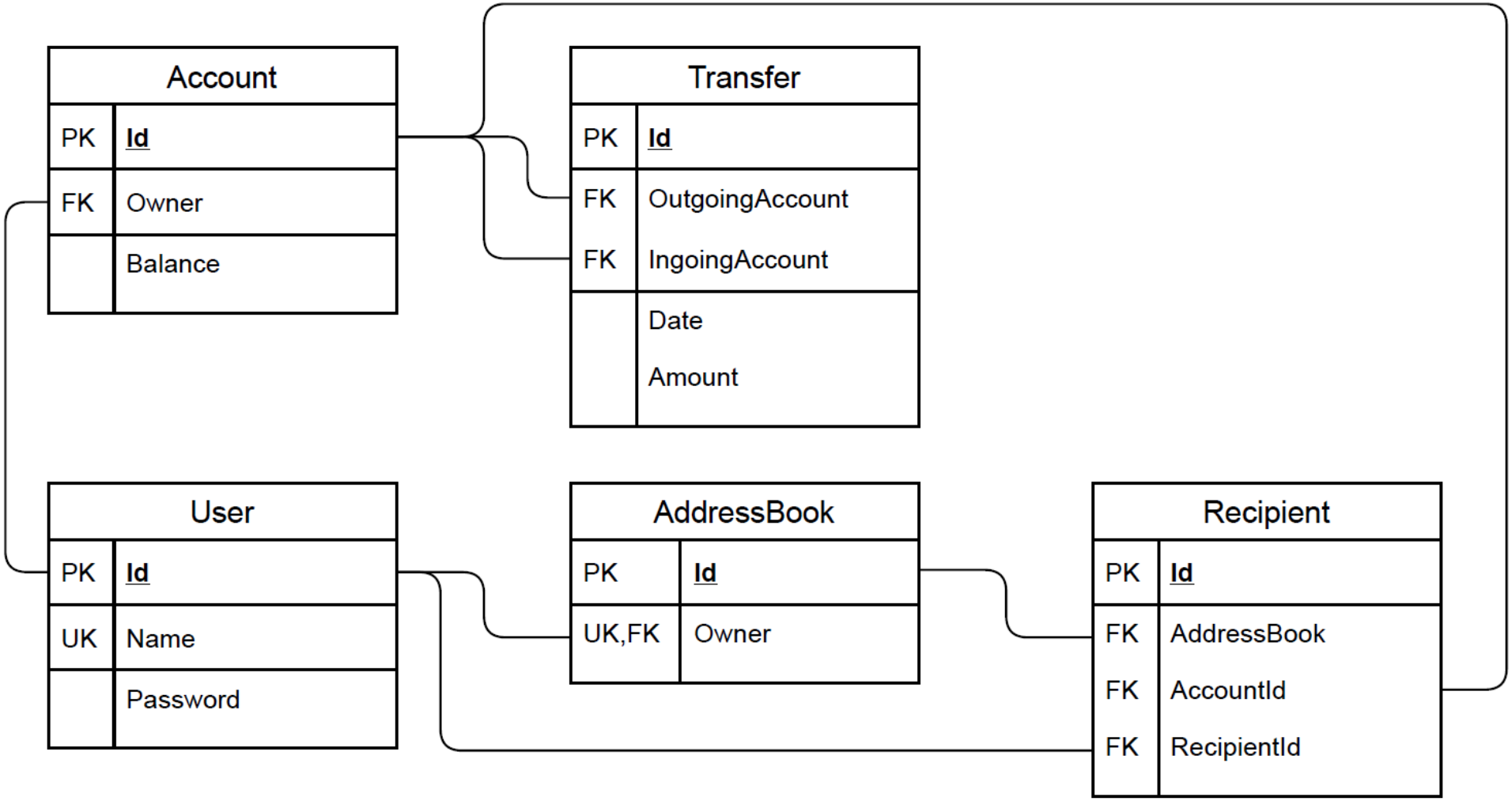
Diagramma ER del database RIA



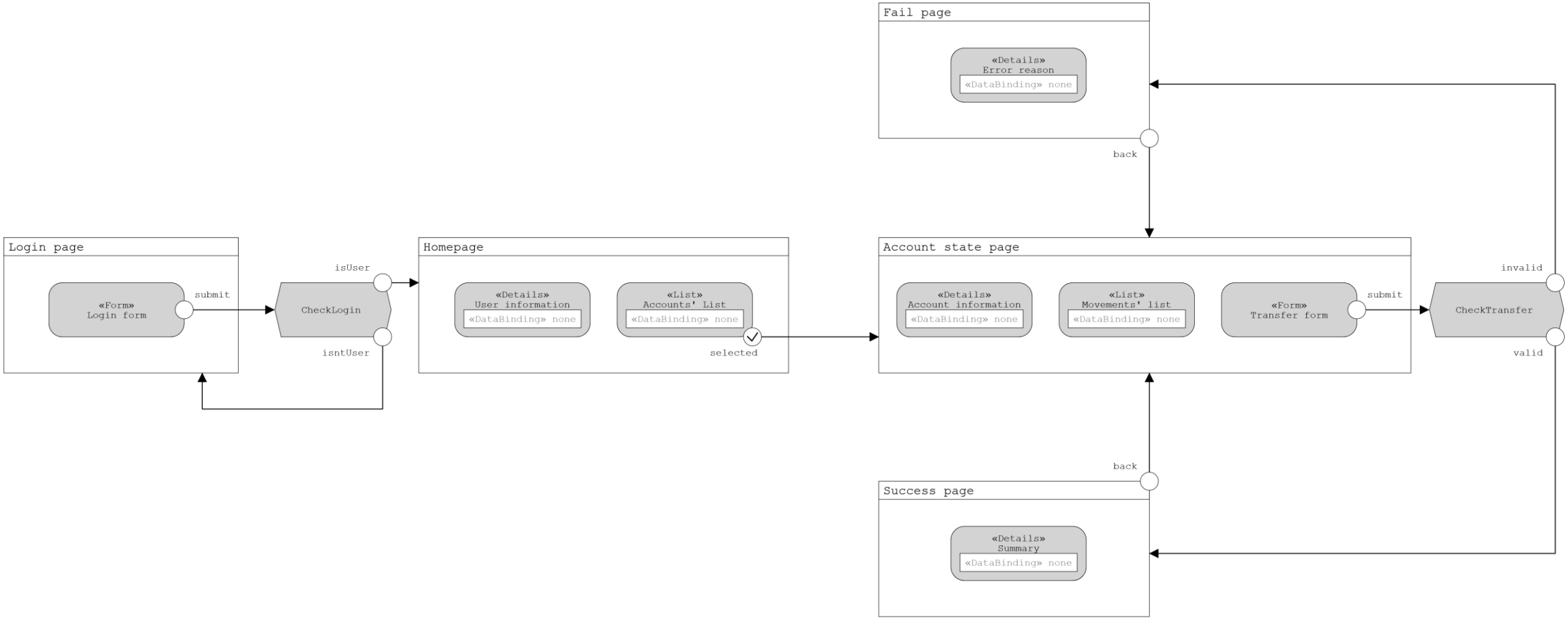
Progetto logico del database PURE HTML



Progetto logico del database RIA



Schema IFML dell'applicazione PURE HTML

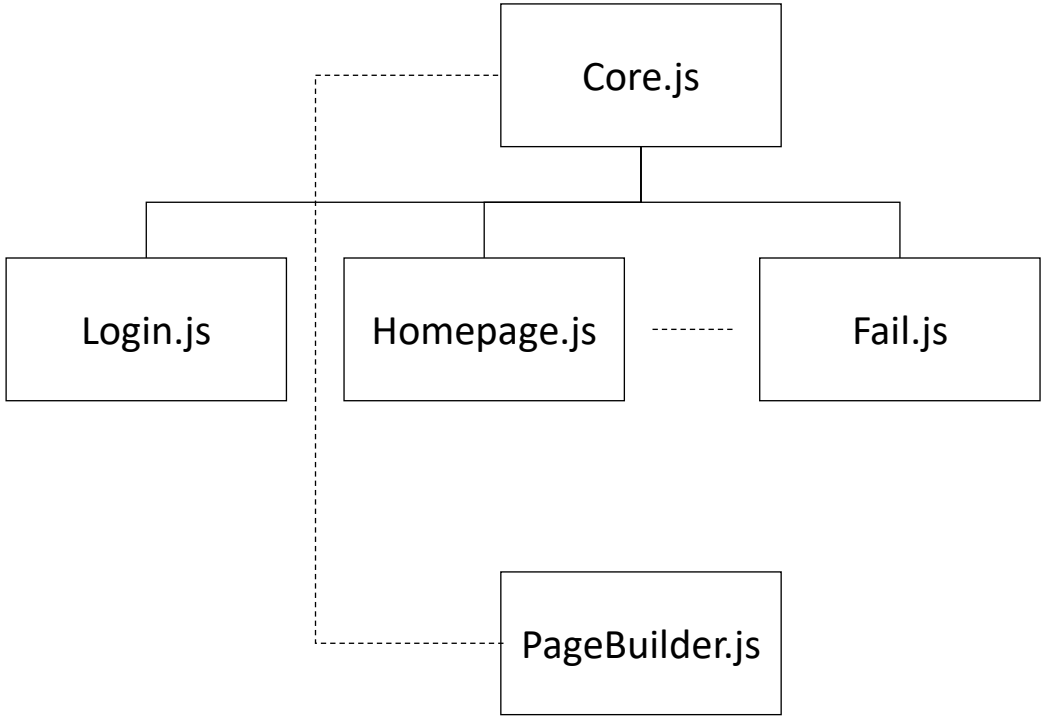


Schema progettuale della versione RIA

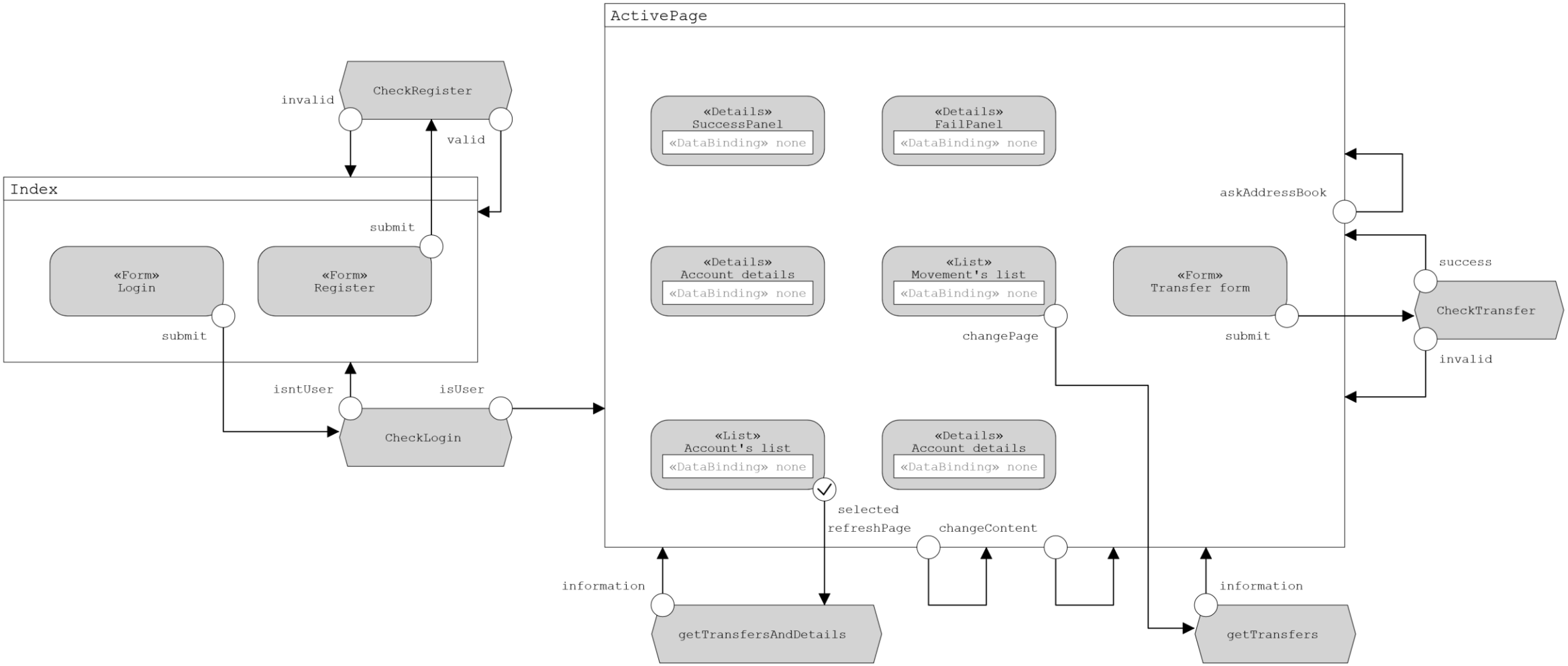
Contenuto di core.js:
La progettazione della versione RIA dell'applicazione consiste nella presenza di un file Javascript principale chiamato Core.js che contiene le classi fondamentali per la funzionalità dell'applicazione in versione one page usando le chiamate asincrone AJAX al server.
In questo file sono contenute anche le definizioni delle classi che rappresentano gli oggetti da rappresentare nell'applicazione mediante l'utilizzo del costrutto «class» di Javascript della versione ECMAScript 2015.

Contenuto dei file di pagina js:
Il contenuto all'interno dei file di pagina js è l'applicazione di listener javascript per i diversi tipi di evento che possono verificarsi sulla pagina e che sono supportati. Essi sono specifici per ogni pagina e vengono chiamati quando la pagina viene impostata per aggiungere un listener poiché i contenuti non sono sempre presenti e vengono generati man mano che l'utente naviga all'interno del sito web.

Contenuto del file PageBuilder.js:
Questo è un altro file principale dell'applicazione che viene utilizzato dal core e per la precisione da un PageManager che gestisce la pagina web. Questo opportunamente chiamato quando vi è la necessità di cambiare la pagina web effettua la chiamata ad un'apposita funzione del PageBuilder per costruire la nuova vista per il client.



Schema IFML dell'applicazione RIA



Componenti applicazione PURE HTML

Beans:
- User
- Account
- Transfer

DAO:

UserDAO
- + findUser
- + findAccounts

AccountDAO
- + findAccount
- + findTransfers
- + numberOfTransfers
- + changeAmount

TransferDAO
- + findTransfer
- + createTransfer

Controllers:
- CheckLogin
- CheckTransfer
- GetLogin
- GetLogout
- GetError
- GetHomepage
- GetAccountState

Filters:
- UserFilter
- AccountFilter

Views:
- Login page
- Home page
- Account state page
- Success page
- Fail page
- Error page

Componenti applicazione RIA

Beans:
- User
- Account
- Transfer
- AddressBook
- Recipient

DAO:

UserDAO
- + findUser
- + findAccount
- + createUser

AccountDAO
- + findAccount
- + findTransfers
- + numberOfTransfers
- + changeAmount

TransferDAO
- + findTransfer
- + createTransfer

AddressBookDAO
- + findAddressBook

RecipientDAO
- + findRecipient

Controllers:
- CheckLogin
- CheckRegister
- CheckTransfer
- CheckAddRecipient
- GetLogin
- GetActivePage
- GetUserDetails
- GetAccountDetails
- GetTransfers

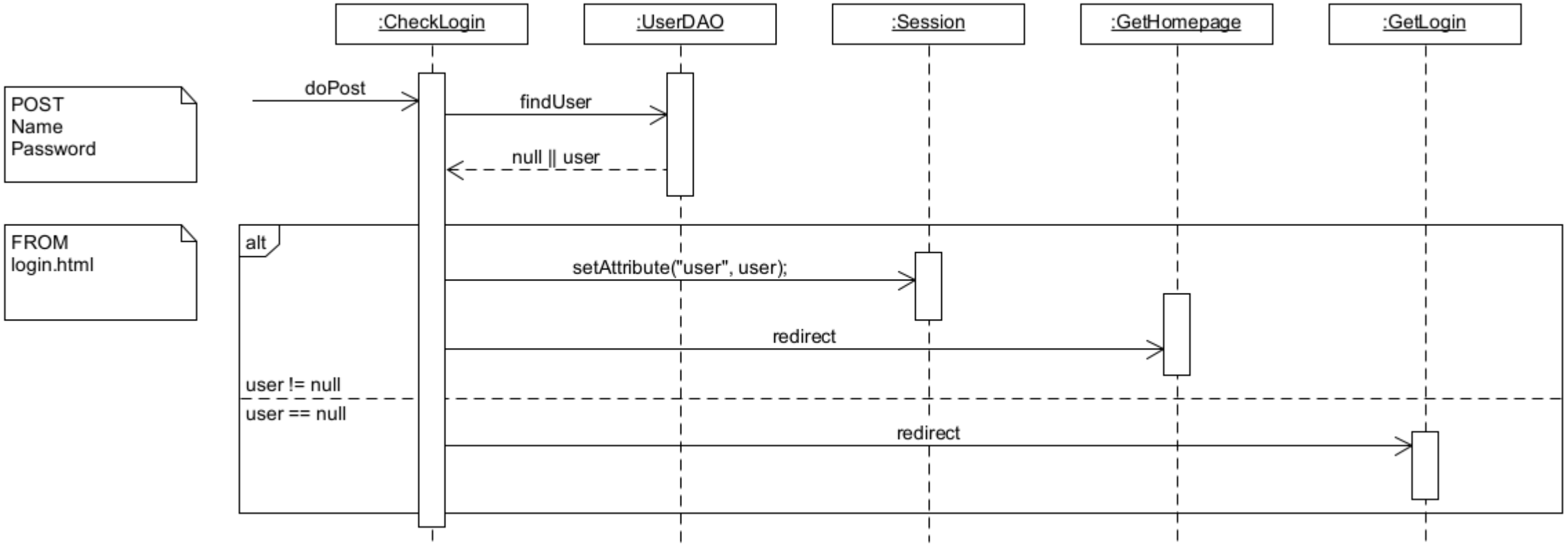
Filters:
- UserFilter
- AccountFilter

Pages:
- Login page
- Active page

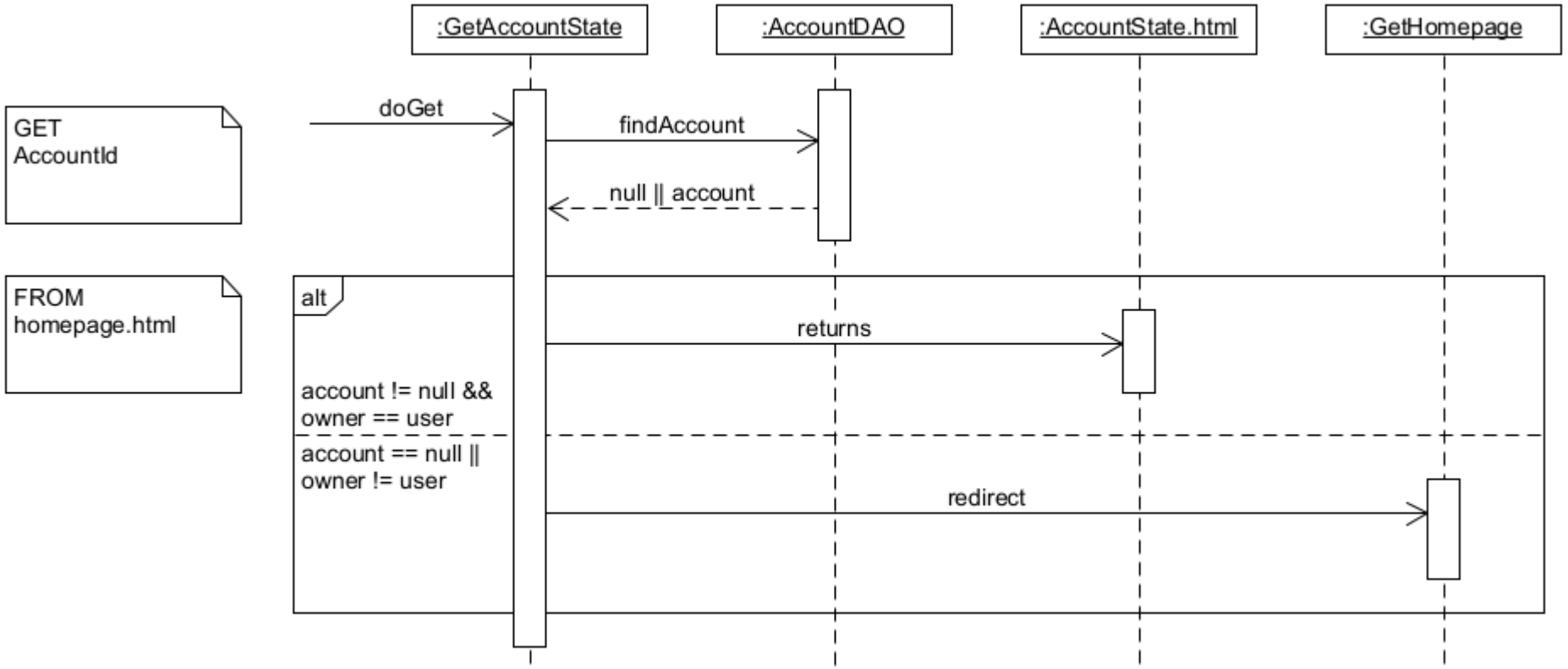
Views:
- Login view
- Home view
- Account view

Diagrammi di sequenza PURE HTML

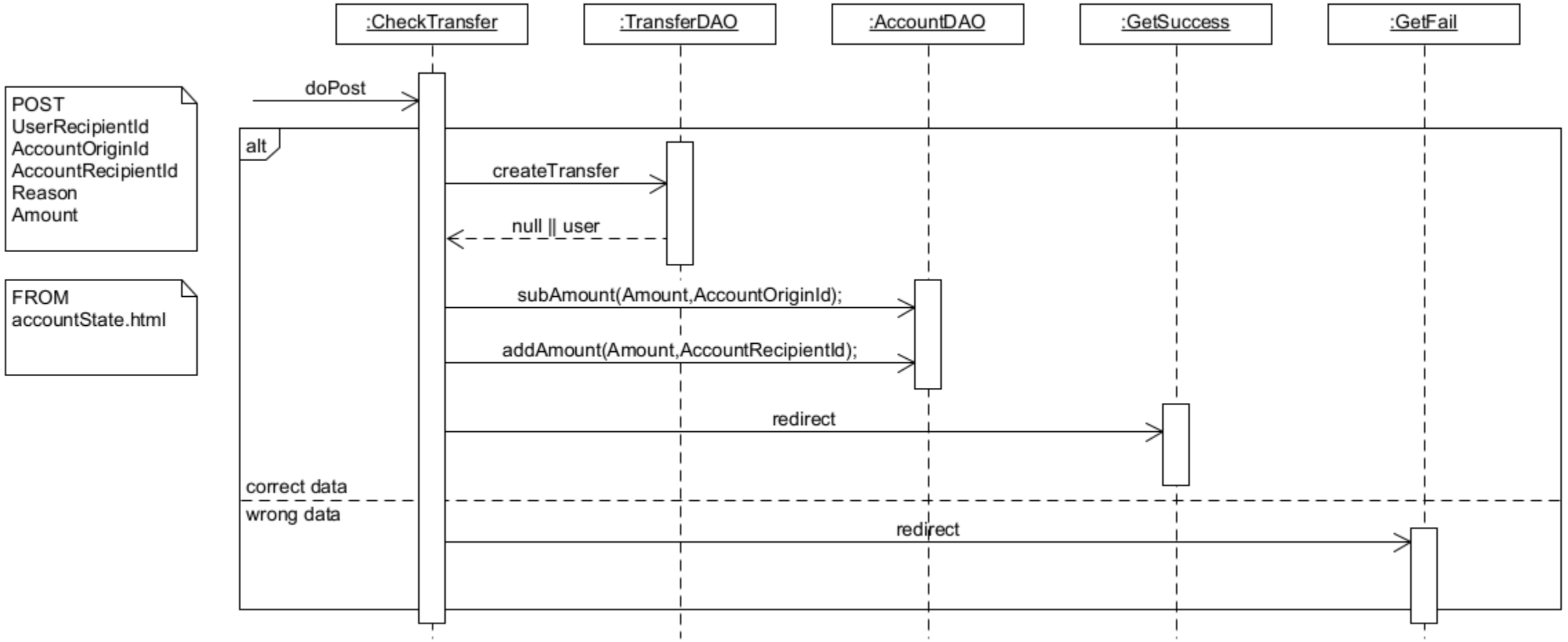
Accesso


:CheckLogin
:UserDAO
:Session
:GetHomepage
:GetLogin
POST
Name
Password
doPost
findUser
null || user
FROM
login.html
alt
setAttribute("user", user);
redirect
user != null
user == null
redirect

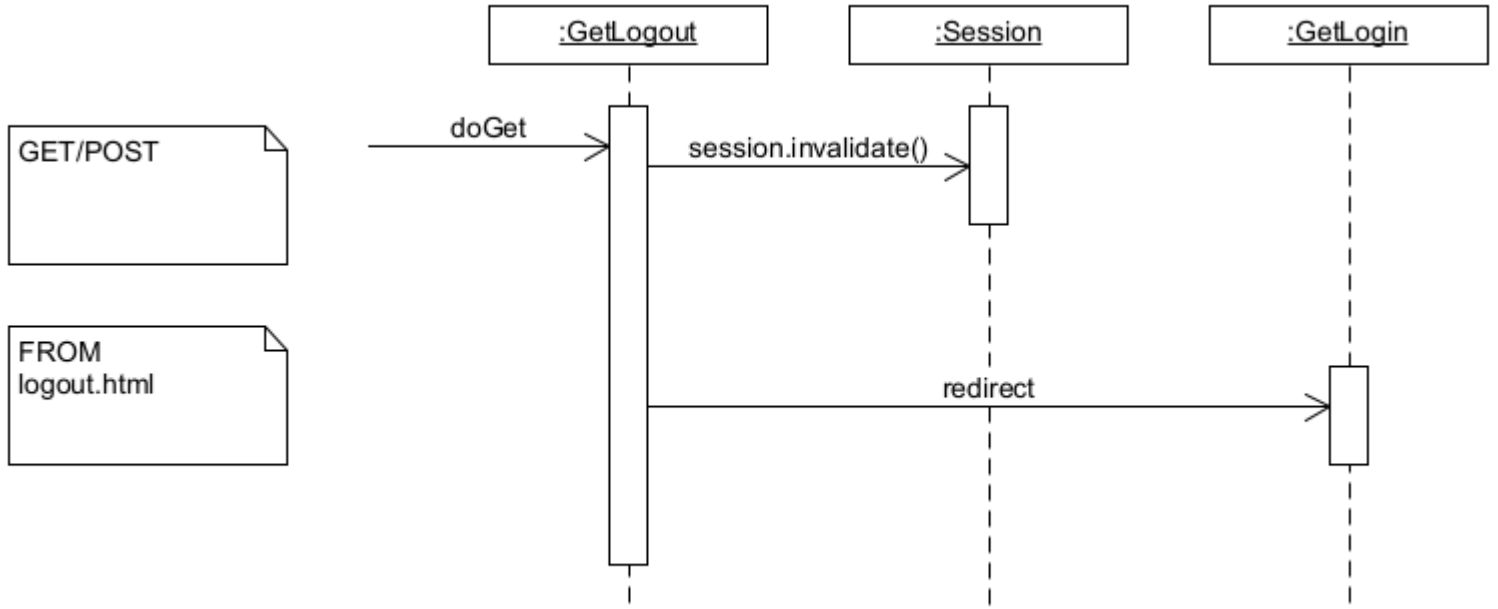
Selezione di un conto



Ordinazione trasferimento



Uscita


:GetLogout
:Session
:GetLogin
GET/POST
doGet
session.invalidate()
FROM
logout.html
redirect

Diagrammi di sequenza RIA

Registrazione

Accesso

Selezione di un conto

Cambio pagina dei trasferimenti

Ordinazione trasferimento

Aggiunta destinatario alla rubrica

Uscita