**User Account Use-Case**

Table of Contents

[Overview 2](#_Toc491474822)

[Technologies used 2](#_Toc491474823)

[Tools Used 2](#_Toc491474824)

[Node Js Components 2](#_Toc491474824)

[Resource URI’s 2](#_Toc491474825)

[Logging in the Application(API) 2](#_Toc491474826)

[Transaction List(API) 2](#_Toc491474827)

[Fund Transfer(API) 3](#_Toc491474828)

[Adding Customer(Mongo DB) 3](#_Toc491474829)

[Adding Benificiary(Mongo DB) 3](#_Toc491474830)

[Remove Benificiary(Maongo DB) 3](#_Toc491474831)

[Deposit money(API) 3](#_Toc491474832)

[Database Tables 3](#_Toc491474833)

[Validation/Error Message 4](#_Toc491474834)

[Repository URL 4](#_Toc491474835)

[Application Flow with screenshot 4](#_Toc491474836)

# Overview

This application has been created w.r.t. use case provided for user account management. Using this App user can do the following functionalities.

* User will be able to login the application with valid credentials.
* User can Register if he/she is a first user.
* User will be able to see the personal with provided values in Mango DB database.
* User can submit the amount in his/her own account as well as some other’s account as well.
* User can Add Benificiary if he/she wants to transfer the money.
* User can remove the beneficiary if not require.
* User can select transfer option and transfer the funds to another user after adding him as a beneficiary.
* User can see the transaction details.

# Technologies used

* AngularJs 1.x
* Restful Web Service (Jax-RS)
* Hibernate
* Node Js

# Tools Used

* Eclipse Mars 2.0
* Visual Studio Code
* Git Repository
* H2 Database
* Maven
* Mongo DB
* Brackets

# Node Js Components

## package.json

generated by npm init command

## node\_module package

generated by npm install command

## app.js

Application starts with this Js

## route.js

Handling all routed request

# Resource URI’s

## Logging in the Application(API)

users/{userName}/{password}

## Transaction List(API)

Users/transactions/{accountId}

## Fund Transfer(API)

/accounts/{fromAccountId}/{benificiaryAccountId}/fundTransfer/{amountTransfered}/{remarks}

## Adding Customer(Mongo DB)

/addCustomer

## Adding Benificiary(Mongo DB)

/addBenificiary

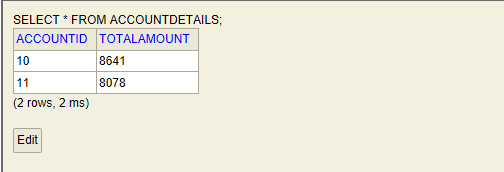
## Remove Benificiary(Maongo DB)

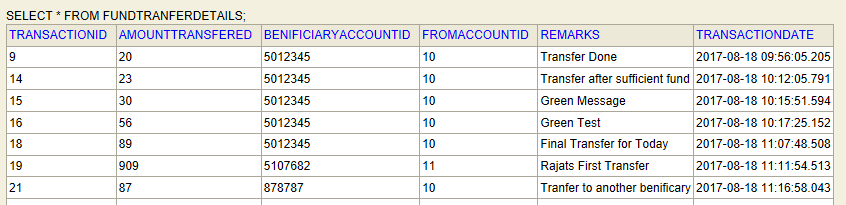
/removeBenificiary

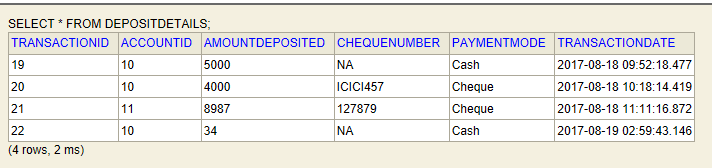
## Deposit money(API)

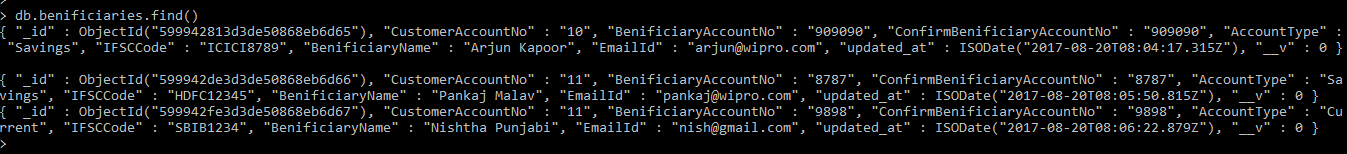
users/accounts/{accountId}/deposits/{amountDeposited}/{paymentMode}/{chequeNumber}

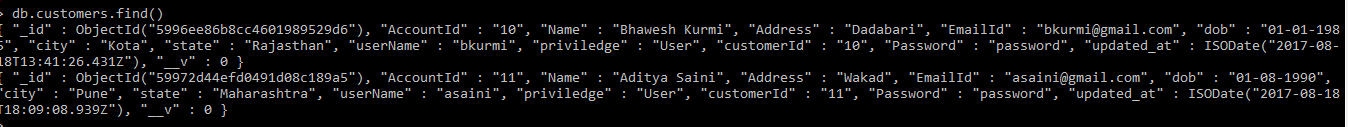
# Database Tables











# Validation/Error Message

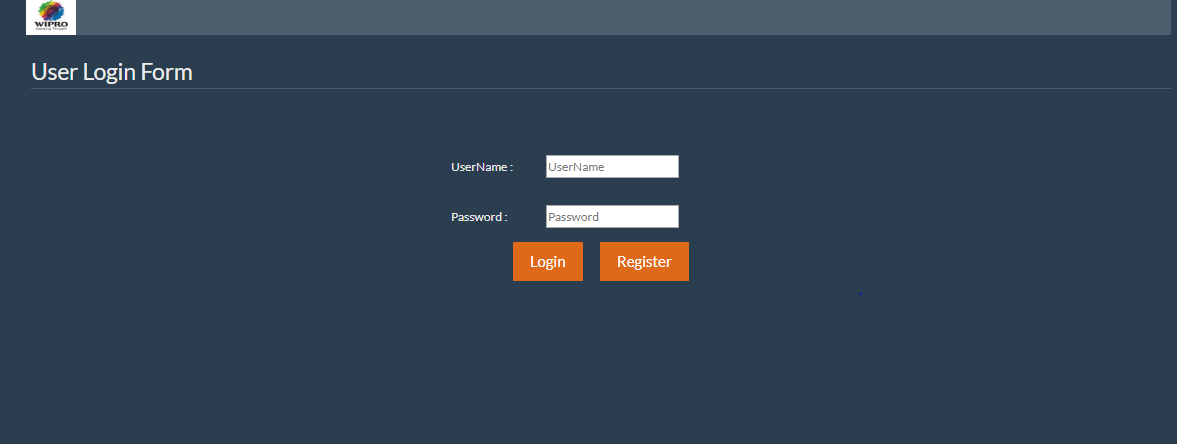
* If User trying to deposit money and server or Database is down.
  + Exception Message: “Database or Server is down.”
* If User trying to deposit money via Cash/Cheque and Account does not exist.
  + Exception Message: “Account does not exist.”
* User trying to login and username or password is invalid.
  + Exception Message: “Invalid Username and Password.”
* User trying to add invalid email.
  + Exception Message: “Please enter valid email address”
* User trying to add beneficiary account number does not match.
  + Exception Message: “Benificiariy account number does not match”
* Submit button only enables after filling all the fields of the form.

# Repository URL

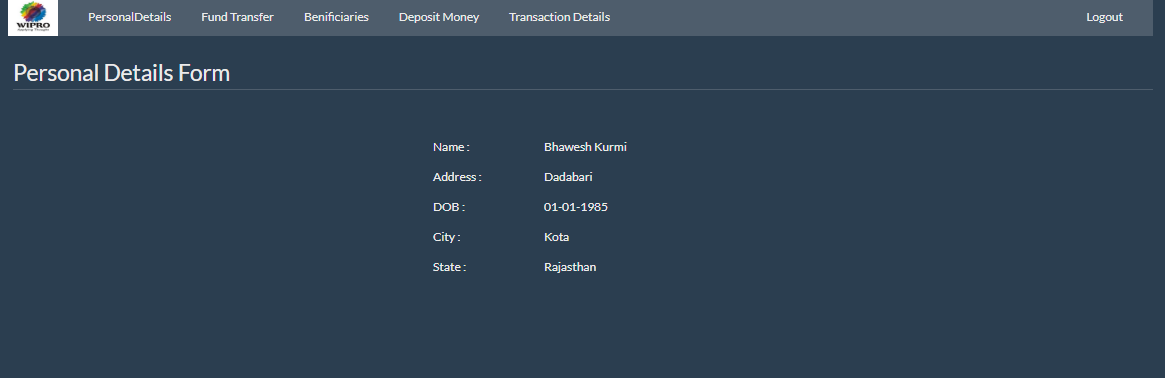
* UI Code (Angular JS)
  + <http://wosggitlab.wipro.com/PA316586/WproProjectUISubmission.git>
* RESTFul API (JAX-RS)
  + <http://wosggitlab.wipro.com/PA316586/UserAccountAPI.git>
* User Account API Document
  + http://wosggitlab.wipro.com/PA316586/WiproUseCaseDocuments.git

# Application Flow with screenshot

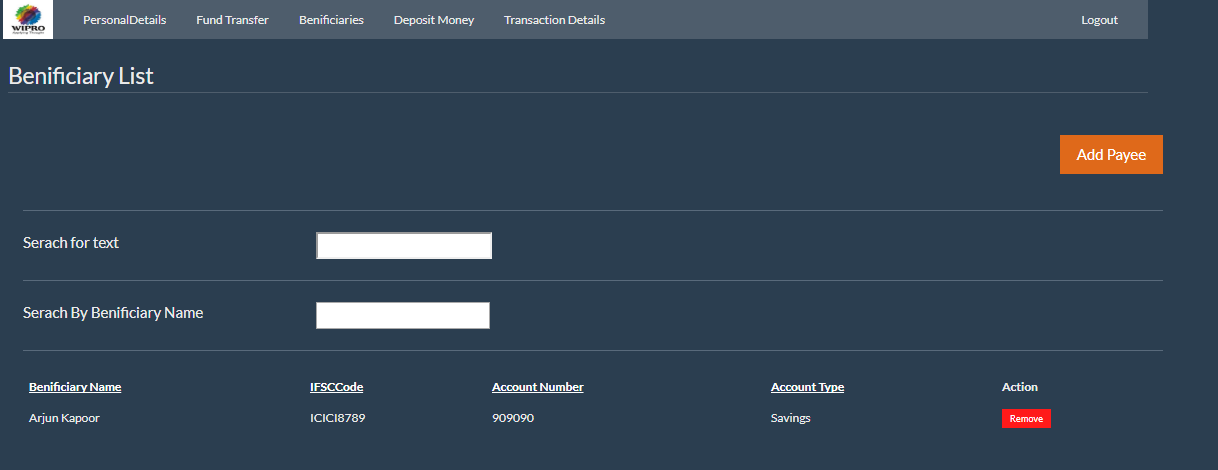
1. Application Login



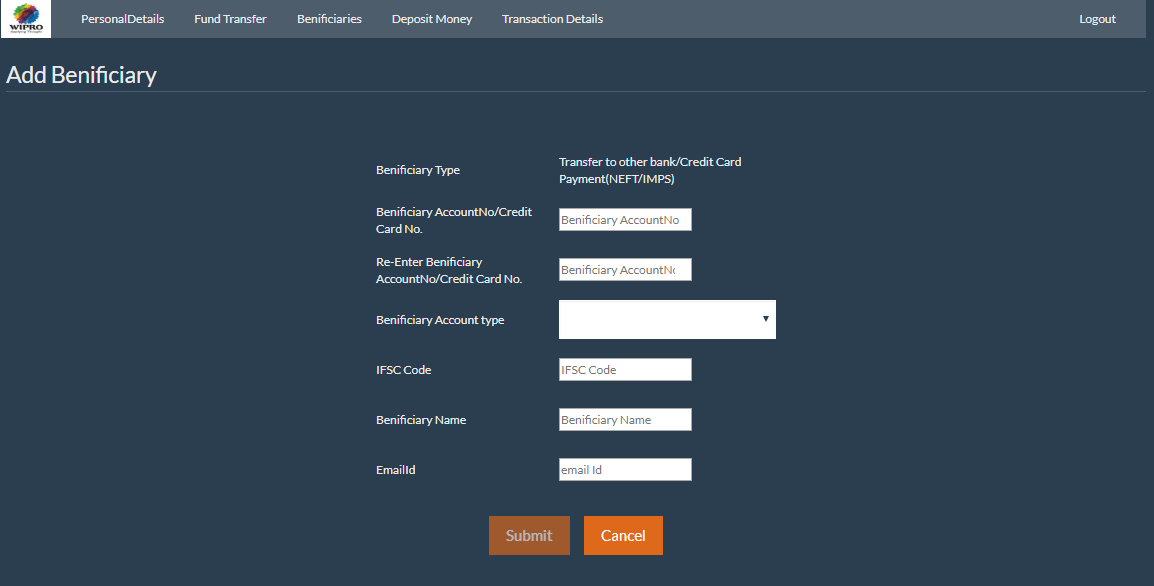
1. Personal Details



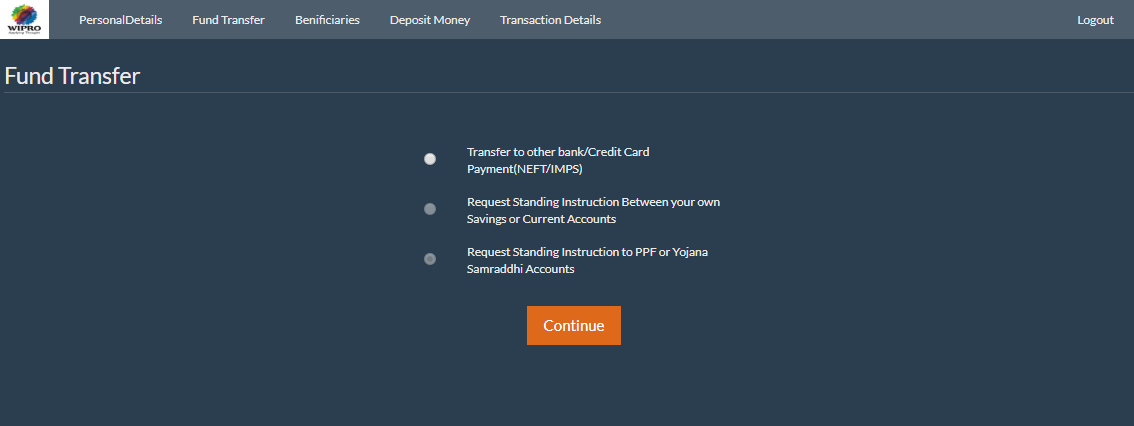
1. Benificiary List



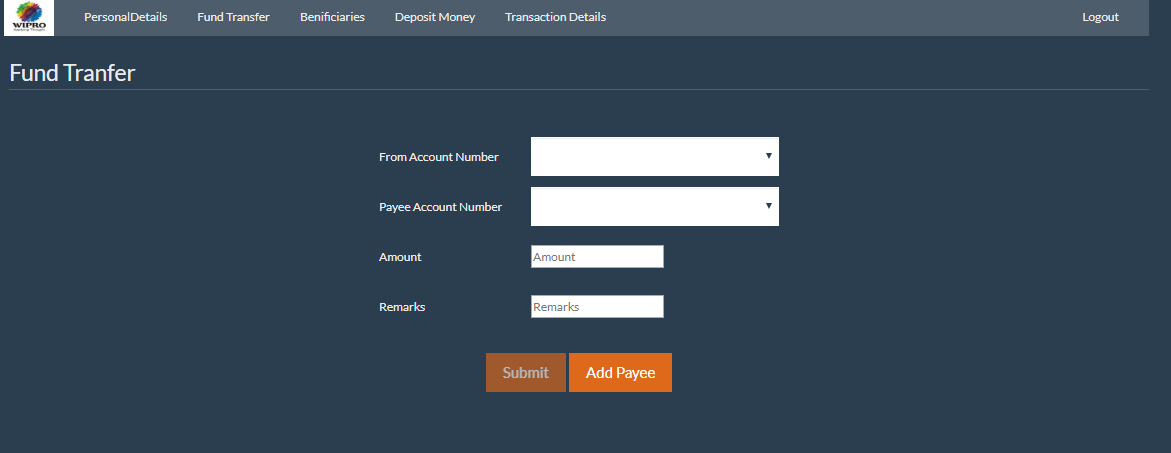
1. Add Benificiary



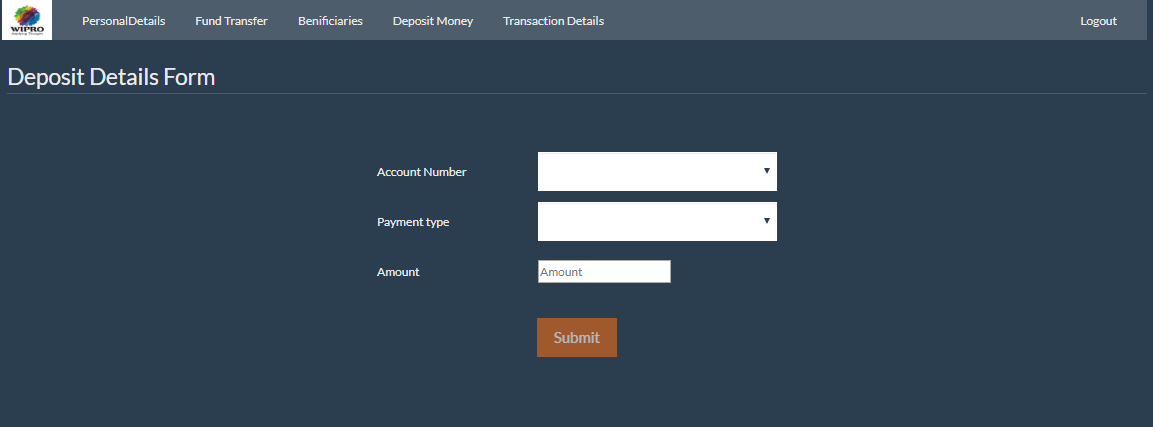
1. Fund Transfer Options



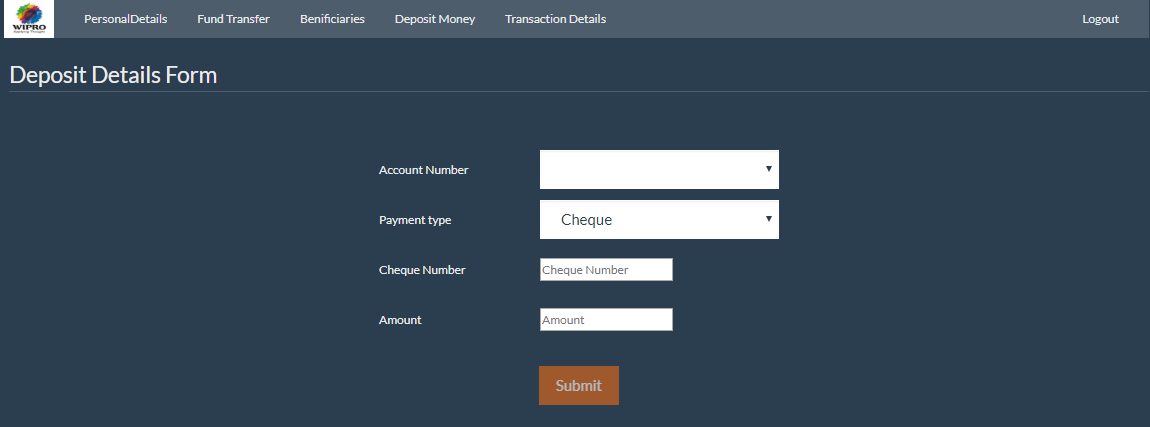
1. Fund transfer



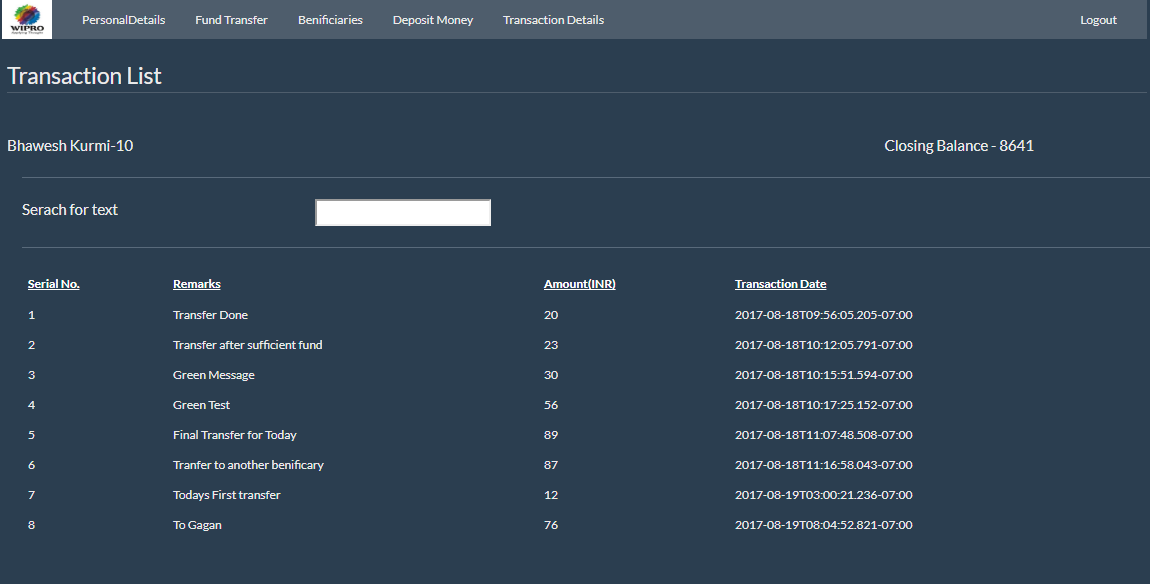
1. Deposit Money with Cash



1. Deposit Money with Cheque



1. Transaction List



1. Add Customer Page

