FILE CATALOGUE

FACELETS

index.xml - Initial web page allowing all users to register or login to the payment service

Registration.xml - Web page to allow payment service users to register to the payment service

login.xml - Web page to allow all users to login to the payment service

error.xml - Users are redirected to this page if they login with an incorrect password or non existent username

User Web Pages

home.xml - When a payment service user logs in they are presented with a welcome message, their balance, their sent and received transactions and they can access all functionality that they are authorised for along with being able to log out.

makePayment.xml - Users can send a payment amount to other registered users. Errors are displayed if the amount is not an integer, the recipient is not an email address, the recipient email address does not exist or if the sender does not have enough funds to perform the payment.

paymentSent.xml - User is redirected to this page when a payment has been successfully made

sendPaymentRequest.xml - Users can send a payment request using an amount and an email address. Errors are displayed if the amount is not an integer, the recipient is not an email address or if the recipient email address does not exist.

paymentRequestSent.xml - User is redirected to this page when a payment request has been successfully sent.

checkPaymentRequests.xml - Users can view all incoming payment requests to their account, the amount is converted into the users's currency and the user can respond to the request by clicking on the send payment button which sends a payment back to the sender of the payment request or the user can reject the payment request.

paymentAccepted.xml - User is redirected to this page when a payment request has been accepted and a payment has been sent in response.

paymentRejected.xml - User is redirected to this page when a payment request has been rejected.

Converter

MultiDateConverter.java - This java file is used as part of a date converter. It takes the date of birth date string inputted by a payment service user upon registration and converts it to a Date object to allow other tiers to process it. It also converts a Date object into a string when the web tier wants to display the date. The file also acts as a pattern matcher when the user registers, as it ensures the date must be in at least one of the two specified date patterns, otherwise the user cannot register.

Admin Web Pages

adminHome.xml - When an administrator logs in they are presented with a welcome message and they have access to all administrative functionality along with being able to log out.

adminViewUsers.xml - Administrators are able to view all payment service users in the database along with all their information.

adminViewTransactions.xml - Administrators can view all payments and payment requests along with more information about these transactions such as the sender, recipient, amount sent, the converted amount received and the status of the transaction.

adminUserTransactions.xml - Administrators are able to view all transactions of a specific user by inputting a username into the search box. Administrators can also view more information about these transactions such as the sender, recipient, amount sent, the converted amount received and the status of the transaction. Errors are displayed if the username does not exist in the database.

addAdmin.xml - Administrators can add other administrators into the database by inputting a username and password.

adminAdded.xml - Administrators are redirected to this page when they have successfully added an administrator into the database.

CONFIGURATION FILES

facelets-config.xml - This page contains a set of navigation rules detailing how users and administrators navigate between pages.

web.xml - contains configuration about the web application and also security related configuration such as security roles and security constraints.

glassfish-web.xml - auto generated configuration file upon changing the context root to /webapps2017.

JSF MANAGED BEANS

RegistrationBean.java - This bean is used to register payment service users into the database. It works by binding all of its fields to the input fields in the registration form and obtaining the inputs selected by the user upon registration, it finally calls the relevant EJB method to actually conduct the registration process.

LoginBean.java - This bean takes the username and password entered in the login form, securely logs in the user and redirects to them the relevant page depending on their role. An error outcome navigating the user to an error page is returned if the entered username does not exist in the database of if the password is incorrect

WelcomeBean.java - This bean is used by both administrators and payment service users for different purposes upon their login. In terms of payment service users this bean is used to obtain the user's first name and surname, their balance, their sent and received recent transactions and format the amounts of those transactions with the relevant currency symbol, however for administrators it is only used to obtain their username upon login.

PaymentBean.java - This bean is used to capture the amount entered by the user and the email address of the payment recipient in order to call the EJB method to conduct the sending of a payment using these parameters. During the sending payment process the bean calls various EJB methods to determine if the email address entered is valid, whether the sender has sufficient funds and whether the sender is trying to send a payment to themselves. The bean also ensures if a concurrent access conflict occurred in the EJB it notifies the user.

PaymentRequestBean.java - This bean is used to capture the requested amount entered by the user and the email address of the payment request recipient in order to call the EJB method to conduct the sending of a payment request. During the sending of a request process the bean calls various EJB methods to determine if the email address entered is valid and whether the sender is trying to send a payment request to themselves.

PendingRequestBean.java - This bean is used to determine whether or not to accept or reject a payment requests. It is responsible for obtaining the payment requests of a logged in user and responding to a particular request by accepting it, in which case an EJB method is used to send the responding payment or if the request is rejected and EJB method is used to set the status of the payment request to rejected. The bean also ensures that whilst retrieving all payment requests the amounts are formatted and converted into the logged in user's currency.

AdminViewUsers.java - This bean is used to obtain a list of all payment service users from the database whilst also formatting the amounts of their balances by adding their respective currency symbols and formatting their date of births to the dd/mm/yyyy pattern. The list of users is later used by the adminViewUsers.xml page in order to display the users in a table.

AdminViewTransactions.java - This bean is used to obtain a list of all payment service transactions from the database along with formatting the amount sent and the received amount by adding their respective currency symbols. This list is used on the adminViewTransactions.xml page in order to display all transactions in a table.

AdminUserTransactionsBean.java - This bean is used to obtain a list of a particular payment service user's sent and received transactions. The bean also calls an EJB method to determine if the username entered on the adminUserTransactions.xml page exists in the database. The bean also ensures that the amount and converted amount of those transactions is formatted by adding the relevant currency symbol.

AddAdminBean.java - This bean is used to add an administrator into the database using the username and password entered in the addAdmin.xml page. The bean also calls an EJB method to determine if an administrator with the entire username already exists in the database, if not then the bean calls an EJB method to add the administrator into the database, otherwise the bean displays a message notifying the administrator that an administrator already exists with the supplied username.

LOCAL INTERFACES

UserService.java - This interface is used to define methods that determine a payment service user's functionality.

AdminService.java - This interface is used to define methods that determine an administrator's functionality.

ENTERPRISE JAVA BEANS

UserServiceBean.java - This EJB implements the UserService interface and therefore provides implementations for all methods defining an online payment service user's functionality.

AdminServiceBean.java - This EJB implements the AdminService interface and therefore provides implementations for all methods defining an administrator's functionality

AdminLoadBean.java - This Singleton EJB upon it's construction inserts an administrator into the database with a username of admin1 and a password of admin1.

CurrencyConverterBean.java - This EJB provides one method to convert an amount from one currency to another and is used when sending payments and when a user checks their payment requests.

ENTITIES

SystemUser.java - This file defines an entity representing both payment service user's username and password and all administrator's. This entity is used for authorisation when a payment service user logs in by checking the password field. The reason this entity was used to store administers is that it simply provides username and password fields, so rather than defining a separate entity administrators can be stored here. T

AppUser.java - This file defines an entity which is more user specific in that it represents the information that an online payment service user will need to store e.g. an account balance, rather than administrators. This entity also provides a version field for the purpose of providing OptimisticLocking when dealing with concurrent access control.

SystemUserGroup.java - This file defines an entity representing all users and their respective groups.

UserTransaction.java - This file defines an entity representing a generic user transaction which can be either a payment or a payment request.