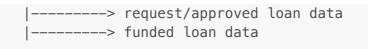
## Variable Names

Name	Туре	Structure	Visibility	Purpose
contractOwner	address	N/A	public	store the owner of the contract
User	struct	<pre>{bytes32 credId, address[] ethAddress,bytes32 brightID,bytes32 bloomId,bytes32 passportScanId}</pre>	N/A	Structure to store user id details
credId	mapping	<pre>(address ethAddress =&gt; bytes32 credId)</pre>	N/A	mapping of user address with the credId
userInfo	mapping	<pre>(bytes32 credId =&gt; User user)</pre>	N/A	mapping of credId with the User struct
credIdList	bytes32[]	N/A	N/A	Array of all the credids of the users. Note: In future, implement this off-chain
activeLoans	mapping	<pre>(bytes32 credId =&gt; bytes32 loanId)</pre>	N/A	mapping of credld with the active loanld as user has One Loan request. (Active Loans means: Requested/ Approved/ Accepeted/ Funded loans)
inactiveLoans	mapping	<pre>(bytes32 credId =&gt; bytes32[] loanId)</pre>	N/A	mapping of credId with inactive loanIds. (Inactive loans weans: Not approved/ Not Accepted/ Not funded loans)
LoanTerms	struct	<pre>{uint256 principalAmt, uint8 interestRate, uint8 duration}</pre>	N/A	Structure to store requested loan details. (principalAmt: amount in DAI, interestRate: rate is calculated annually, duration: duration is in months)
loanRequested	mapping	<pre>(bytes32 loanId =&gt; LoanTerms loanTerms)</pre>	N/A	mapping of loanId with the loan terms requested (loanTerms struct) with the user address

Name	Туре	Structure	Visibility	Purpose	
IoanApproved	mapping	<pre>(bytes32 loanId =&gt; LoanTerms loanTerms)</pre>	N/A	mapping of loan terms approved (loanTermsApproved struct) with the user address	
IoanStatus	mapping	<pre>(bytes32 loanId =&gt; uint8 loanStatus)</pre>	N/A	Store the loan status. [0: No status, 1: Requested, 2: Approved, 3: Not Approved, 4: Accepted, 5: Not Accepted, 6: Funded, 7: Not Funded]	
IoanRequestedDate	mapping	<pre>(bytes32 loanId =&gt; uint256 loanRequesteddate)</pre>	N/A	Store date and time when loan is requested	
IoanApproveDate	mapping	<pre>(bytes32 loanId =&gt;   uint256   oanApprovedDate)</pre>	N/A	Store date and time when loan is approved or not approved	
IoanAcceptedDate	mapping	<pre>(bytes32 loanId) =&gt; uint256 loanAcceptedDate)</pre>	N/A	Store date and time when loan approved terms is accepted or not accepted by user	
loanFundedDate	mapping	<pre>(bytes32 loanId =&gt; uint256 loanFundedDate)</pre>	N/A	Store date and time when loan is funded or not funded to user	
PaymentDetails	struct	<pre>{uint256[] dueDates, uint8[] statuses, uint256 monthlyPayment, uint256 totalPayment}</pre>	N/A	track payment details of the loan. Note: In fututre, may get rid of uint256 totalPayment variable. [0: Pending, 1: Successful, 2: Missed]	
userLoanPayment	mapping	<pre>(bytes32 loanId =&gt; PaymentDetails paymentDetails)</pre>	N/A	mapping of loanId with the PaymentDetails struct	

## Data connections

```
ethAddress --> credID --> user details |----> loan id --> loan status
```



## Functions

Function name	Input Parameters	Return Parameters	Called By	Description
isOwner	N/A	bool	User/ Owner	To check the contract owner address
registerUser	N/A	bytes32 credId	User/Owner	To register the user logged in for the first time and creates the credId for new users registered.
getCredId	N/A	bytes32 credId	Logged in User	Get the cred id for the current logged in user
getAllCredId	N/A	bytes32[] credIdList	Only Owner	Get all the cred ids for all users
getUser	bytes32 credId	<pre>address[] ethAddress, bytes32 brightId, bytes32 bloomId, bytes32 passportScanId (User struct)</pre>	Logged in User/ Owner	Get user details (lds)
getActiveLoan	bytes32 credId	bytes32 loanId (activeLoans[credId)	Logged in User/ Owner	Get the loanld of the active loan
getInactiveLoans	bytes32 credId	bytes32[] loanId	Logged in User/ Owner	Get the inactive loans from the credid of the user

Function name	Input Parameters	Return Parameters	Called By	Description
requestLoan	uint256 principalAmt, uint8 interestRate, uint8 duration	bytes32 loanId, uint8 loanStatus, uint256 loanRequestDate	Logged In User	submit the loan request and store the loan requested terms
approveLoan	bytes32 loanId, bool yesNo	uint8 loanStatus, uint256 loanApproveDate	Only Owner	approve the loan request raised by user
acceptLoan	byes32 loanId, bool yesNo	uint8 loanStatus, uint256 loanAcceptedDate	Only Logged In User	accept the approved loan terms
fundLoan	byes32 loanId, bool yesNo	uint8 loanStatus, (uint256[] dueDates, uint8 statuses, uint256 monthlyPayment, uint256 totalPayment(PaymentDetails (struct))), uint256 loanFundedDate	Only Owner	fund the user with the approved loan amount, create the payment schedule and change loan status to funded
getLoanStatus	bytes32 loanId	uint8 loanStatus	Logged In User/ Owner	Get the status of the loan
getRequestedLoan	bytes32 loanId	<pre>uint256 principalAmt, uint8 interestRate, uint8 duration(loanRequested (mapping (bytes32 loanId =&gt; LoanTerms loanTerms))),uint256 loanRequestedDate, uint8 loanStatus</pre>	Logged in User/ Owner	Get the loan requested terms

Function name	Input Parameters	Return Parameters	Called By	Description
getApprovedLoan	bytes32 loanId	<pre>uint256 principalAmt, uint8 interestRate, uint8 duration(loanApproved (mapping (bytes32 loanId =&gt; LoanTerms loanTerms))),uint256 loanApprovedDate, uint8 loanStatus</pre>	Logged in User/ Owner	Get the loan details for the approved loan terms
getPayment	bytes32 loanId	PaymentDetails (struct)	Owner	Get the payment schedule and monthly payment and total payment details
getPayment	N/A	<pre>uint256[] dueDates, uint8 statuses, uint256 monthlyPayment, uint256 totalPayment (PaymentDetails (struct))`</pre>	Only Logged in User	Get the payment schedule and monthly payment and total payment details using user address
makePayment	N/A	N/A	Only logged in User	Make the payment for the monthly loan amount