Lending Features

Area	SubArea	Compound	Aave	Acala	Parallel Finance	Sushi Kashi	Genshiro	Blacksmith (Warp v2)	dXdY	Composable V0	vNext
Summary							Statistics Based	LP tokens collateral yields on Uniswap. Solidity, Compound, Chainlin.		Modeled after warp. finance, providing over-collateralized pair isolated markets	
Integration	Vault	It seems there is only liquidity pool, no vault.						User puts Collateral into Vault User WhiteLists Pair to allow Lending use Asset/Collateral in Lending		Deposits /Withdraws assets in Vaults	
Configuration	Market state				Active, Pending, Supervision,			Paused, Unpaused		-	
Features	Borrow several times same asset							-		-	
Configuration	Borrow Limit	Collateral Factors => [0,1] The sum of the value of an accounts underlying token balances, multiplied by the collateral factors, equals a user’s borrowing capacity				Vault is responsible for providing Borrower underlying amount. Pair is responsible for its limit See viewBorrowLimit					
Incentives /Attacks	Create many accounts and borrow onto each				Account address for each market shares borrows capacity with other markets						
Internals	Math							256 bit		64/128 bit	
Internals	Wrapping				deposits are done via lending to allow WrappedTokens and some accounting			Proxies all calls to Oracle/Vault and Wraps token		Mostly abstracted by Vault and Oracle	
Configuration	Reserve Rate /Factor/Ratio	The different between borrow and supply rate.									
Configuration	Reserve	Admin can add or remove. Accumulated as part of Interest rate More reserve means smaller utilization factor (smaller interest rate) and smaller redeem (people are less likely to take away deposit asset)								No in traditional sence	
Internals	Debt Tokens		https://docs.aave.com/developers/the-core-protocol/debt-tokens					Accounting easier Used for Credit Delegation			

Configuration	Borrow Rate (raw, reference or copy paste)	3+ interest rate modles https://github.com/compound-finance/compound-protocol/tree/master/contracts (and white paper for example)			Jump Curve V2 + custom formula	https://github.com/sushiswap/kashi-lending/blob/94d9bfcadcdf5157e7bb14904cf8495ad1ca2e4f/contracts/KashiPair.sol#L152 https://docs.sushi.com/faq-1/kashi-bentobox-faq Dynamic rate changing with time		Jump Curve V2 https://github.com/warpfinance/blacksmith/blob/master/contracts/interest/JumpRateModelV2.sol			
Assets	Deposits /Reservers /from Lenders							Stable coins			
Assets	Assets	Not isolated	(v1) Not isolated		Not isolated. Market owner sets assets and collaterals. User puts collaterals of any kind and deposits of any kind. Each time on borrow, user only provides currency id, all his collaterals are normalized to some stable price via oracle.	Pairs Isolated Users can create pairs		Pairs Any supported by price Oracle. Can create any pair by anybody via initializes. Can avoid sharing vault sharing vault if upload smartcontract, but sharing is default.		Any Pairs Supported by Vaults and Oracle and Vault provides liquidity on Market creation	
Assets	Governance (in code, voted, configured, voted pair, voted currency)	Compound will begin with centralized control of the protocol (such as choosing the interest rate model per asset), and over time, will transition to complete community and stakeholder control.									? Provided LP tokens may then be used for per-pool governance, such as changes in configurations (lending rate curves etc.) If governance for the overall protocol need is required, some custom logic will need to be implemented.
References	Documents	https://compound.finance/documents/Compound.Whitepaper.pdf	https://raw.githubusercontent.com/aave/aave-protocol/master/docs/Aave_Protocol_Whitepaper_v1_0.pdf https://raw.githubusercontent.com/aave/protocol-v2/master/aave-v2-whitepaper.pdf		https://docs.parallel.fi/white-paper		https://genshiro.equilibrium.io/docs/genshiro-whitepaper.pdf	https://docs.warp.finance/warp-finance/resources/litepaper-1/the-warp-protocol		https://composablefinance.atlassian.net/wiki/pages/resumedraft.action?draftId=2916374	
References	Code							https://github.com/warpfinance/Warp-Contracts/tree/contracts/contracts https://github.com/warpfinance/Warp_Smart_Contracts to get actual deployed address that can be used on etherscan.		https://github.com/ComposableFi/composable-pallets/tree/lending/frame/lending	

Integration	Staking							https://www.element.fi/		Vaults	
Integration	DEX Demand (Borrower) yields from Collateral							Uniswap Sushiswap (based on https://github.com/warpfinance/Warp_Smart_Contracts)	Uniswap	-	DEX
Integration	Leverage					Can be used to leverages trading on platform				-	
@ dzmitry		easy	little bit complicated, but sane; so many small nice features in v2		math is not clearly written, so can use code			how difference is measured for stable vs not on chain?	build on L2 ZKF		
Configuration	Tokens	For each asset token Wrapped token issued						For each asset and collateral token Wrapped tokens are issued			
Configuration	Governance	• white listed tokens • collateral factor • started from admin, promised to delegate to community									
Configuration	Liquidation							`LendingPair.sol/liquidate` LiquidationPrice = LoanStableCoin * LiquidationRatio / LoanCollateral ? LiquidationRatio should me no more than 1 /Collateral ratio		-	
Configuration	Collateral ratio https://www.investopedia.com/terms/o/overcollateralization.asp					`// Settings for the Medium Risk KashiPair CLOSED_COLLATERIZATIO`		Over collateralized v1: Collateral /Deposit = 1.50		Over collateralized	

							N_RATE = 75000; // 75% OPEN_COLLATERIZATION_RATE = 77000; // 77% COLLATERIZATION_RATE_PRECISION = 1e5; // Must be less than EXCHANGE_RATE_PRECISION						

							(due to optimization in math) MINIMUM_TARGET_UTILIZATION = 7e17; // 70% MAXIMUM_TARGET_UTILIZATION = 8e17; // 80% UTILIZATION_PRECISION = 1e18; FULL_U						

							TILIZATION = 1e18; FULL_UTILIZATION_MINUS_MAX = FULL_UTILIZATION - MAXIMUM_TARGET_UTILIZATION; FACTOR_PRECISION = 1e18; STARTING_INTEREST_PER						

							_SECOND = 317097920; // approx 1% APR MINIMUM_INTEREST_PER_SECOND = 79274480; // approx 0.25% APR MAXIMUM_INTEREST_PER_SECOND = 317097920000; // approx						

							1000% APR INTEREST_ELASTICITY = 28800e36; // Half or double in 28800 seconds (8 hours) if linear EXCHANGE_RATE_PRECISION = 1e18; LIQUIDATION_MUL						

							TIPLIER = 112000; // add 12% LIQUIDATION_MULTIPLIER_PRECISION = 1e5; // Fees PROTOCOL_FEE = 10000; // 10% PROTOCOL_FEE_DIVISOR = 1e5; BORROW_OP						

						ENING_FEE = 50; // 0.05% BORROW_OPENING_FEE_PRECISION = 1e5;					
Collateral type	?Fiat-collateralized									Any	
Features	Rewards							Reward for Incentives Pair Liquidity Mining program		decided by Vault	
Features	Iliquidity (allow to take collateral if there is not enough linquidity to return asset to lender) lender										
Features	Credit Delegation - Allow to delegated borrowers to uncollateralize loans - Allows delegation to other procols		https://docs.aave.com/developers/guides/credit-delegation					YES		-	
Flash Loans	Collateral Swap										
Flash Loans	Arbitrage										

Flash Loans	Flash Loans		+		• Flesh Liqudation for FlashLoans • liquidate loans and put collateral to third • replay dept in collateral (it prise known for block) • debt swap • margin trading			yes, Vault provides API for flesh loans		Protected from doing flash loans without fee	
Configuration	Interest • supply/demand based SD (if there is too much of cache - borrowing is cheap) • or Stable (S)	SB, fixed, governance for adjustement most simplest formula with small slope when utilization high	SD, S(but it is not really stable - it can change)		by default is like compound, but has formula (experimental) making rate going faster if utilization becomes high (not sure so what parameters are). BorrowInteresteRate can be manupilated by Utilizaton_target (chanes formula)	Elastic interest rates responding to supply and demand (see formula in docs)			https://help.dydx.exchange/en/articles/2924246-how-do-interest-rates-work		
Configuration	Interest Accrual							`Lendingpair.sol/accrueInterest`			
Configuration	Protocol Fee	See Reserve Rate				10%		Fixed for Pair User knows all before he enters		decided by Vault	
Yield	Liquidator							Collateral -15%			
Yield	Lender										
Yield	Platform							5% of StableCoinInterest of Lender 15% of Collateral during Liquidation			
Yield	Borrower							YieldTotalCollateralInterestRate+Stablecoin (YieldUserDerived=OtherPlatformInterestRateYieldTotalCollateral)			
Interest	Lender							YiledOnInterestEarningTokens+StableCoinInterest			
Pain points											
Feature	Maturity		definite for stable rates							non definite	
Liquidation		collateral must be larger than borrow discount to give more lending token for liqudation. who defines collateral factor?	Partial Liqudation								
Integration	Oracle	Uses top 10 exchanges to determine price, so not formula given and it is not automatics seems			Semi decentralized, decided not use any from Polkadot or shared; Elected	• can be chosen by market		ChainLink		Oracle	

Withdrawal		no restrictions								Vault allowance (like 15% of Vault)	
Assets	Collateral		Can have multi currency collateral (so these evaluated into ETH)		Wrapped token or Chosen Collateral			LP token of DEX. Several pairs, with stable coin in each			
Double use		-			• can stake what is not yet lended • ? borrowers can borrow collaterals to borrow more?						
Feature	Personification				If user staked some currencies, DOT and KSM, his reputation increased his yields are increased; Borrowers Penalty						
Purpose Specific loans					For auctions						
Feature	Identity (KYC), useful for Credit risk reduction. Using data (examples, CeFi bank account) to reduce size of collateral										
Fees		mentions direct interests to lenders			mentioned						
Delegation			• allow to redirect interest to other users • allows users to fill collater and pass it to other user								
Security			Wallet indirection - so special trading wallet is used which stores only collateral								