Audit

git: https://github.com/je1att0/DEX_solidity/blob/main/src/DEX.sol

commit : `6974ccbabbfdf24a32ca8390aa39dbc4a0f58929`

ID	요약	위험도
JAC-001	`addLiquidity` 함수에서 유동성 비율 계산 없이 유동성 풀에 추가됩니다.	High

Description

`addLiquidity` 함수는 사용자가 두 개의 토큰(`_amountX` , `_amountY`)을 제공하여 유동성 풀에 추가할 때, 현재 유동성 풀의 비율에 따라 `_amountX` 와 `_amountY` 를 계산하지 않고 바로 추가합니다.

이는 유동성 풀의 균형을 고려하지 않은 방식입니다.

Impact

High

공격자가 풀의 비율을 조작하려면 **상당한 자산이 필요**하며, 이로 인해 공격 비용이 높을 수 있습니다.

하지만 **풀 비율을 충분히 조작시킬 수 있다면**, 공격자는 막대한 이익을 얻을 수 있습니다.

Recommendation

- 유동성 풀 비율 검증 로직 추가

ID	요약	위험도
JAC-002	`removeLiquidity` 함수는 유동성 공급자 개인의 LP 토큰량을 검증하지 않고 전체 LP 토큰량만 검증합니다.	information

Description

`removeLiquidity` 함수는 유동성 공급자 개인의 LP 토큰량을 검증하지 않고 전체 LP 토큰량만 검증합니다.

이로 인해 사용자가 자신의 LP 토큰량을 초과하여 함수 호출을 시도할 경우, 불필요한 가스비를 소모하게 됩니다.

Impact

information

함수 호출을 실패하는 경우 불필요한 가스비를 소모합니다.

Recommendation

- 유동성 풀에서 제거하려는 개인 LP 토큰량에 대한 검증 로직을 구현

ID	요약	위험도
JAC-003	`swap` 함수에서 입력 값이 모두 0인 경우에도 정상적으로 실행됩니다.	information

Description

`swap` 함수에서 사용자의 부주의로 인해 입력 값을 모두 0으로 설정한 경우에도 정상적으로 실행되기 때문에 불필요한 가스비가 소모됩니다.

Impact

information

사용자 부주의로 인한 함수 호출시 불필요한 가스가 소모됩니다.

Recommendation

- `swap` 함수 입력 값이 모두 0인지 검증하는 로직을 추가