# 🚀최종 기획서: BlockPass Ω (오메가)

> “ID+Password 시스템에 블록체인 DNA를 완전히 주입한 차세대 로그인 인프라”

---

## 🧭 서비스 개요

**BlockPass Ω**는 기존의 `ID + Password` 로그인 방식을 유지하면서, 그 내부 인증 구조 전체를 블록체인 네트워크 기반으로 재설계한 **차세대 하이브리드 인증 인프라**다.

기존 로그인은 서버(DB)에 저장된 비밀번호와의 비교로 인증을 수행하지만, BlockPass Ω에서는 비밀번호 자체를 **블록체인 해시체인**, **스마트컨트랙트**, **분산 세션**, **DID 레이어**를 통해 관리한다.

즉, 사용자는 평범하게 로그인하지만, 실제로는 **다중 블록체인 합의와 스마트컨트랙트 검증**이 그 뒤에서 동시에 작동한다.

---

## 🛖 아키텍처 총괄 구조

```
사용자 브라우저 / 앱
 │
 ▼
[BlockPass Ω SDK]
 ├── 비밀번호 해시 생성 (로컬)
 ├─ DID-Key 매핑 생성
 ├─ Blockchain Auth API 호출
 ├─ Signature & Token 생성
 └── 결과 토큰 → 세션 발급
 │
 ▼
[BlockPass Ω API Gateway]
 ├─ Auth Router / GraphQL API
 ├─ Smart Contract Connector
 ├─ DID Resolver & SessionVault
 ├─ Node Cluster (Fabric + BDID)
 └─ IPFS/Arweave Storage Gateway
 │
 ▼
[Blockchain Network Layer]
 ├─ BDID Chain (Identity)
 ├─ Fabric Private Chain (Auth)
 ├─ Ethereum / Polygon (Audit)
 ├─ Arweave / IPFS (Logs)
 └─ Cosmos IBC Bridge (Inter-chain Sync)
```

---

## 🚚 인증 절차 (단계별 상세 흐름)

### ① 로그인 요청

사용자가 `ID + Password` 입력 → SDK가 로컬에서 비밀번호 해시(`SHA3-512 + Salt + Nonce`) 생성.

### ② DID 자동 매핑

- 최초 로그인 시: ID와 해시를 기반으로 BDID(DID) 자동 생성
- 이후 로그인 시: SDK가 자동으로 해당 BDID 키를 불러와 인증

```
did:bdid:<domainHash>:<hash(ID+deviceHash)>
```

### ③ Auth SmartContract 호출

SDK는 Fabric Private Chain의 `AuthContract`에 트랜잭션 생성:

```
POST /api/auth/login
{
  id: "user001",
  hash: "a17f9c...",
  did: "did:bdid:xyz...",
  deviceHash: "d9e0c..."
}
```

스마트컨트랙트가 저장된 해시와 비교 → 검증 성공 시 **AuthReceipt** 생성.

### ④ Public Chain 검증 (이중합의)

AuthReceipt가 Ethereum/Polygon의 `ΩValidatorContract`로 전달되어
블록체인 합의 노드에서 재검증 → 트랜잭션 해시를 Audit Layer에 기록.

### ⑤ Smart Session 생성

로그인 성공 시 자동으로 `SmartSessionContract(SSC)` 배포.
이 세션은 블록체인에 존재하며, 만료·로그아웃이 자동 실행된다.

```
contract SmartSession {
  address public user;
  bytes32 public sessionHash;
  uint256 public createdAt;
  uint256 public expiresAt;
  function invalidate() public {...}
}
```

### ⑥ 로그 & 폐기 프로세스

로그인 성공, 실패, 만료 내역은 모두 **IPFS**에 암호화 저장,
해시값만 Arweave에 영구 기록. 7일 후 SmartContract에 의해 자동 폐기.

---

## 블록체인 구성요소별 역할

| 계층 | 체인 | 주요 역할 | 특징 |
|---|---|---|---|
| **Identity Layer** | BDID Chain | 사용자, 서비스, 디바이스 DID 관리 | DPoS 합의 + Merkle 인증 |
| **Auth Layer** | Fabric | 로그인 해시 비교 및 세션 생성 | 초고속 검증 (0.3초 이하) |
| **Audit Layer** | Ethereum / Polygon | 로그인 영수증(Receipt) 영구기록 | 위변조 불가, 외부검증용 |
| **Log Layer** | IPFS + Arweave | 로그인 로그 저장 | 분산·암호화 로그 |
| **Bridge Layer** | Cosmos IBC | 체인 간 DID/세션 동기화 | 다중 블록체인 호환 |

---

## API 구조 및 예시 구현

### REST + GraphQL 하이브리드 API

| Method | Endpoint | Description |
|---|---|---|
| `POST` | `/api/auth/login` | 로그인 요청 (ID, PW, DID) |
| `POST` | `/api/auth/register` | 계정 등록 + DID 자동생성 |
| `GET` | `/api/auth/verify` | 세션 유효성 검증 |
| `GET` | `/api/auth/logs` | 블록체인 로그 조회 |

**Sample API Flow**

```
// 로그인 요청
const res = await fetch('/api/auth/login', {
  method: 'POST',
  headers: { 'Content-Type': 'application/json' },
  body: JSON.stringify({ id: 'user1', password: 'abcd1234' })
});

// 서버 내부 로직
// 1️⃣ PW Hash → Fabric Contract Compare
// 2️⃣ AuthReceipt 발급 → Ethereum Audit
// 3️⃣ SSC 생성 → SmartContract
// 4️⃣ Session Token 반환
```

응답:

```
{
  "success": true,
  "token": "eyJhbGciOiJI...",
  "txHash": "0xabc123...",
  "sessionContract": "0xSs001..."
}
```


---

## 🚛 보안 구조 요약

| 항목 | BlockPass Ω 처리 방식 | 효과 |
|---|---|---|
| **비밀번호** | 로컬 해시 후 체인 검증 | 서버 유출 불가 |
| **세션** | SmartContract 기반 세션 | 탈중앙 만료관리 |
| **로그** | IPFS 저장 + Arweave 해시 | 삭제·조작 불가 |
| **인증 합의** | Private + Public Chain 이중검증 | 신뢰성 보장 |
| **평판노드** | 노드별 공격패턴 분석 | 자동 방어 알고리즘 |
| **Revoke Control** | DID 리보크 스마트컨트랙트 | 계정 복구 및 인증 취소 |

---

## 차별화 포인트 (일반 로그인 대비)

| 구분 | 일반 로그인 | BlockPass Ω |
|---|---|---|
| 인증 검증 | 서버 DB 비교 | 블록체인 SmartContract 합의 |
| 세션 관리 | 서버 세션 쿠키 | SmartSessionContract 자동만료 |
| 로그 저장 | DB 테이블 | IPFS + 블록체인 기록 |
| 보안성 | 해킹 시 전체 유출 | 분산 노드로 해킹 불가능 |
| 투명성 | 감사 불가 | 모든 인증이 트랜잭션으로 남음 |
| 법적 증거 | 없음 | 블록체인 영수증으로 법적 효력 |

---

## BDID (Blockchain Distributed ID)

- 각 사용자는 ID 기반으로 자동 DID 발급
- `ID + PWHash + DeviceHash` → BDID 키쌍 생성
- DID는 BDID 체인에 등록되어 이후 모든 로그인에서 사용
- DID와 PW는 별개지만 상호 서명 가능 구조

**BDID 예시:**

```
did:bdid:omega:usr:0x8fA31C2b...
```

**DID Document:**

```
{
 "@context": "https://www.w3.org/ns/did/v1",
 "id": "did:bdid:omega:usr:0x8fA31C2b...",
 "publicKey": [{
  "id": "#keys-1",
  "type": "EcdsaSecp256k1VerificationKey2019",
  "controller": "did:bdid:omega:usr:0x8fA31C2b...",
  "publicKeyHex": "0456a3c..."
 }],
 "authentication": ["#keys-1"]
}
```

---

## UX 흐름

1️⃣ 사용자는 평범하게 ID+PW 입력
2️⃣ SDK가 자동으로 DID 매핑 및 해시 생성
3️⃣ 로그인 요청 시 블록체인 합의 자동 수행 (0.7초 내)
4️⃣ 로그인 성공 시 세션 컨트랙트 생성 + 토큰 반환
5️⃣ 로그, 감사, 평판 데이터 자동 기록

> ✅ 사용자 경험은 일반 로그인과 동일하지만,
> 내부적으로 5개의 블록체인 네트워크가 동시에 인증, 기록, 검증을 수행한다.

---

## 🏢 1 서비스화 전략

| 항목 | 내용 |
|---|---|
| **시장 타깃** | 금융권, 공공기관, SaaS 플랫폼, 보안 스타트업 |
| **사업 모델** | API 과금형 / SaaS 구독형 / Enterprise 노드 라이선스 |
| **수익 구조** | 로그인 트랜잭션, DID 발급, 로그저장 수수료 |
| **API 요금제 예시** | Basic(월 10만회 로그인), Pro(월 100만회), Enterprise(무제한) |
| **차별성** | ID+PW 체계를 유지하면서 블록체인 신뢰를 부여하는 유일한 시스템 |

---

## 🛖 1 기술 독창성 요약

- **SmartSessionContract (SSC)** : 세션 자체를 블록체인 자산으로 관리

- **Dual Consensus Verification** : Private + Public 체인 합의 병행
- **BDID 자동 생성 시스템** : 사용자 개입 없이 DID 부여
- **IPFS-LogBridge** : 로그인 기록을 암호화·분산저장
- **Reputation Oracle** : 악성 로그인 자동 차단
- **Passwordless Transition Bridge** : 향후 DID 전용 로그인으로 전환 가능

---

# 1 결론 요약

**BlockPass Ω**는 "ID와 비밀번호가 여전히 존재하지만,
그 모든 인증의 신뢰와 보안이 블록체인 위에서 보장되는 세계 최초의 로그인 인프라"이다.

- 사용자 입장: 평범한 로그인 → 빠르고 익숙함 유지
- 기업 입장: 중앙DB 유출 위험 제거, 감사투명성 확보
- 기술 구조: 블록체인 해시 + DID + 스마트컨트랙트 + IPFS 완전 통합
- 목표: 모든 로그인, 세션, 로그를 "트랜잭션화"하는 표준 프로토콜 구축

---

**슬로건:**
"보이지 않는 블록체인이 당신의 로그인 뒤에서 작동한다." 🔕