NGI TRUST CHAIN

D2. ' DETAILED TECHNICAL SPECIFICATION, IMPLEMENTATION PLAN, AND PRELIMINARY BUSINESS PLAN

>DECAST.LIVE<

07/03/2025

Funded by the European Union

TrustChain Project. Funded by the European Union under CA No 101093274. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Commission. Neither the European Union nor the granting authority can be held responsible for them.

Due date	07/03/2025
Submission date	07/03/2025
Team	DECAST Live
Version	V1.0
Authors	Shivam Dhawan, Ajmal Azad, Peyman Pourjafar, Yasrab

Disclaimer

Copyright notice

The TrustChain Consortium is the following:

Participant number	Role	Participant organisation name	Short name	Country
1	COO	EUROPEAN DYNAMICS LUXEMBOURG SA	ED	LX
2	BEN	F6S NETWORK LIMITED	F6S	IE
3	BEN	UNIVERZA V LJUBLJANI	UL	SI
4	BEN	ATHENS UNIVERSITY OF ECONOMICS AND BUSINESS - RESEARCH CENTER	AUEB	EL
5	BEN	FUNDACION CIBERVOLUNTARIOS	CIB	SP
6	BEN	CONSORCIO RED ALASTRIA	ALA	SP
7	BEN	TIMELEX	TLX	BE
8	BEN	ETHNIKO KAI KAPODISTRIAKO PANEPISTIMIO ATHINON	NKUA	EL
9	AP	CITY UNIVERSITY OF LONDON	ICS	UK

Funded by the European Union

TABLE OF CONTENTS

Funded by the European Union

Funded by the European Union

LIST OF FIGURES

Funded by the European Union

LIST OF TABLES

Funded by the European Union

ABBREVIATIONS

DC	Dissemination and Communication
DID	Decentralised Identifiers
DIH	Digital Innovation Hub
DLT	Distributed Ledger Technology
EDIH	European Digital Innovation Hub
EEN	European Enterprise Network
EIC	European Innovation Council
EU	European Union
GA	Grant Agreement
GDPR	General Data Protection Regulation
NCP	National Contact Point
NGI	Next Generation Internet
NGO	Non-Governmental Organisations
OC	Open Call
OC#4	Open Call #4
SEO	Search Engine Optimization
SME	Small and Medium-sized Enterprises
SSI	Self-Sovereign Identities
WP	Work Package

Funded by the European Union

1. INTRODUCTION

In today's hyper-connected world, live streaming and online messaging are vital for everyday communication. However, ensuring secure, reliable, and privacy-preserving interactions remains a significant challenge. Traditional centralized systems create single points of failure and often require users to share more personal information than necessary, increasing the risk of data breaches and identity theft.

Decast.live addresses these issues by reimagining the underlying infrastructure using a decentralized approach. Our platform leverages technologies such as blockchain, Decentralized Identifiers (DIDs), and zero-knowledge proofs (ZKPs) to provide a secure live casting environment. This enables users to authenticate without revealing sensitive information, maintain control over their data, and access content securely.

From a technical perspective, Decast.live is built with a multi-layered architecture that includes:

- **Authentication and Authorization Layers:** These enforce secure access to live streams by supporting various login methods and dynamically managing permissions.
- **Accessibility and Storage Layers:** By combining traditional cloud services (e.g., AWS) with decentralized storage solutions, we ensure that content is both scalable and tamper-proof.
- **Modular Design:** Our hexagonal architecture separates business logic from infrastructure, allowing for flexible upgrades and seamless integration of emerging technologies.

This document details our design philosophy, pilot results, software modules, API specifications, and implementation plans that together form the backbone of the Decast.live platform. By combining robust cryptographic methods with decentralized data management, we aim to set a new standard for secure, efficient, and user-centric live streaming solutions.

2. PILOT RESULTS

Funded by the European Union

Decentralized Identifiers (DIDs)

1. **Anonymous Yet Verifiable Interaction (New)**
    - **Story:** "As a user, I want to interact anonymously while still being sure that contributors are genuine."
    - **Reason:** Users stressed the importance of balancing privacy with authenticity in communications.
2. **Verify Participant Identities (Existing)**
    - **Story:** "As an event organizer, I want to verify participant identities securely so that only authorized users can access my events."
    - **Reason:** Ensures event security by preventing unauthorized access.
3. **Private Authentication (Existing)**
    - **Story:** "As a viewer, I want to authenticate using DIDs so that my personal information remains private."
    - **Reason:** Protects user privacy while maintaining trust.

Live Streaming

1. **Secure Live Streaming for Teachers (Existing)**
    - **Story:** "As a teacher, I want to stream live lessons securely so that my students can access content without interruptions."
    - **Reason:** Delivers uninterrupted, secure educational content.
2. **High-Quality Streaming for Students (Existing)**
    - **Story:** "As a student, I want to access high-quality streaming so that I can engage in real time."
    - **Reason:** Ensures an interactive and engaging experience.

Funded by the European Union

Storage

1. **Secure Lecture Storage (Existing)**
    - **Story:** "As a teacher, I want to save my lectures securely so that students can review them later."
    - **Reason:** Guarantees safe storage of educational content for future reference.
2. **Tamper-Proof Storage (Existing)**
    - **Story:** "As an administrator, I want tamper-proof storage for session recordings so that legal proceedings are preserved securely."
    - **Reason:** Protects the integrity of legal records by preventing unauthorized modifications.

Access Control

1. **Dynamic Role Assignment (New)**
    - **Story:** "As an admin, I want to assign roles dynamically so that I can manage event audience participation effectively."
    - **Reason:** Flexible and efficient participant management.
2. **Dynamic Session Permissions (Existing)**
    - **Story:** "As a lawyer, I want to control session permissions dynamically so that only authorized individuals participate."
    - **Reason:** Manages access to sensitive consultations.
3. **Customizable Content Visibility (New)**
    - **Story:** "As a user, I want control over who sees my content stored in my blockchain-integrated wallet."
    - **Reason:** Users require practical privacy settings for tailored content sharing.
4. **Easy Account Management (New)**

Funded by the European Union

- **Story:** "As a business owner, I need a quick way to deactivate my account and securely delete my data."
- **Reason:** The previous process was time-consuming and inefficient, impacting user control.

5. **Simplified Registration & Onboarding (New)**
   - **Story:** "As a new user, I want an intuitive, fast registration process so I can start using the platform immediately."
   - **Reason:** Early pilots indicated the registration was overly complex, causing delays in room allocation.

Version Control

1. **Edit Recordings (Existing)**
   - **Story:** "As an educator, I want to edit recordings so that I can provide concise and relevant content."
   - **Reason:** Improves the clarity and relevance of recorded lectures.
2. **Track Recording Changes (Existing)**
   - **Story:** "As a legal team member, I want to track recording changes so that I maintain an accurate history."
   - **Reason:** Provides a reliable audit trail of all modifications to ensure evidence integrity.

Pilot Methodology & Efforts

Users Involved:

- **Educators**: From schools, universities, and online training platforms.
- **Legal Professionals**: Lawyers, paralegals, and court officials.
- **Corporate Trainers and Event Organizers**: From diverse industries.
- **Blockchain Developers**: To test and provide feedback on functionalities.
- **General Users**: Representing a wide range of demographics.

Funded by the European Union

Channels Employed to Reach Users:

- **Professional Networks:** Leveraging platforms like LinkedIn.
- **Educational and Industry Collaborations:** Working with schools, universities, and online communities.
- **Direct Outreach:** Email campaigns and targeted social media initiatives.
- **Ambassador Partnerships:** Engaging domain experts to recruit participants.

Methodological Approach:

1. **User Registration:**
   - Invitations were sent with a streamlined registration link.
   - Role allocation was clearly defined for participants versus campaign initiators.
   - Comprehensive onboarding materials and support guides were provided.

2. **Functional Testing:**
   - Participants validated key features including registration, access control, and account management.
   - Real-time interactions during virtual consultations and live events were simulated.

3. **Feedback Collection:**
   - Surveys and one-on-one interviews gathered qualitative insights on usability, interface design, and practical performance.

Pilot Results

- **Enhanced Onboarding:** Simplified registration and detailed onboarding guides significantly improved user satisfaction.

Funded by the European Union

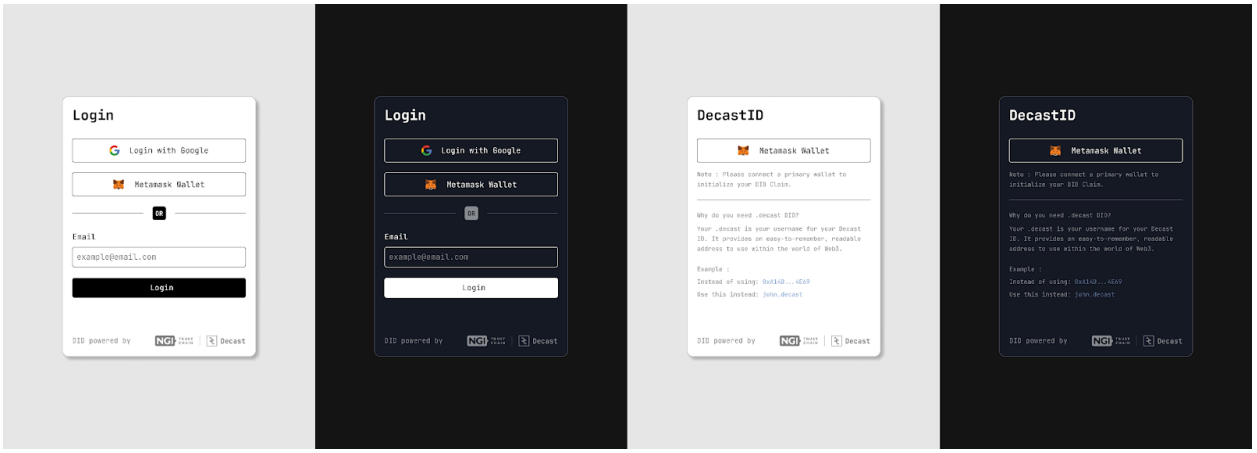

Figure 1 Onboarding

- **User Interface Improvements:** Streamlined navigation between core functions (e.g., secure meeting access, storage control) was well received.

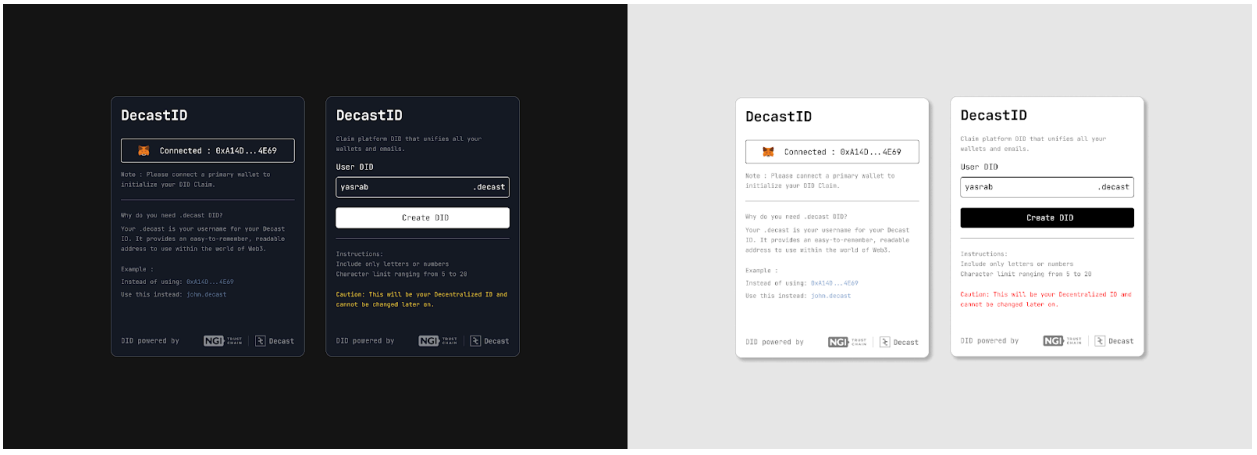

Figure 2 Interface Improvements

- **Practical Privacy Controls:** Customizable visibility settings and easier account management met the practical needs of both individual users and business owners.

Funded by the European Union

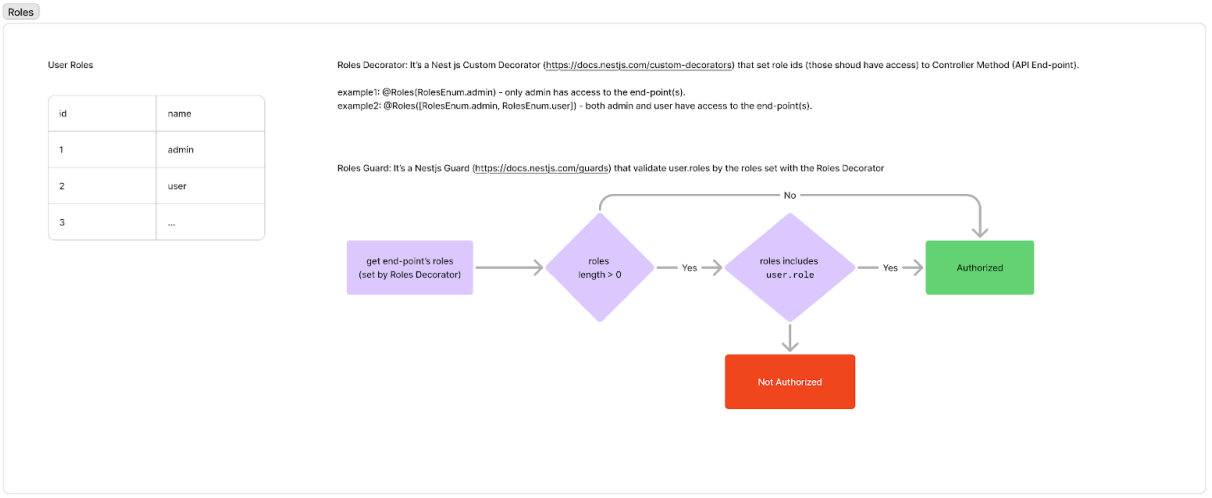

Figure 3 Role Based Access

- **Cost and Open-Source Value:** The open-source approach was recognized for its potential to reduce deployment costs while fostering community-driven improvements.

Conclusions:

The pilot confirmed that DECAST.LIVE effectively addresses real-world user needs with enhanced security, improved usability, and cost-efficient deployment. Feedback highlighted that practical feature—such as simplified onboarding and customizable controls—are as critical as advanced security measures, ensuring broad applicability across diverse user groups.

Design Specifications Drawn from the Pilots

- **Enhanced Onboarding Process:**
  A step-by-step tutorial was introduced to simplify the registration and role allocation.

Funded by the European Union

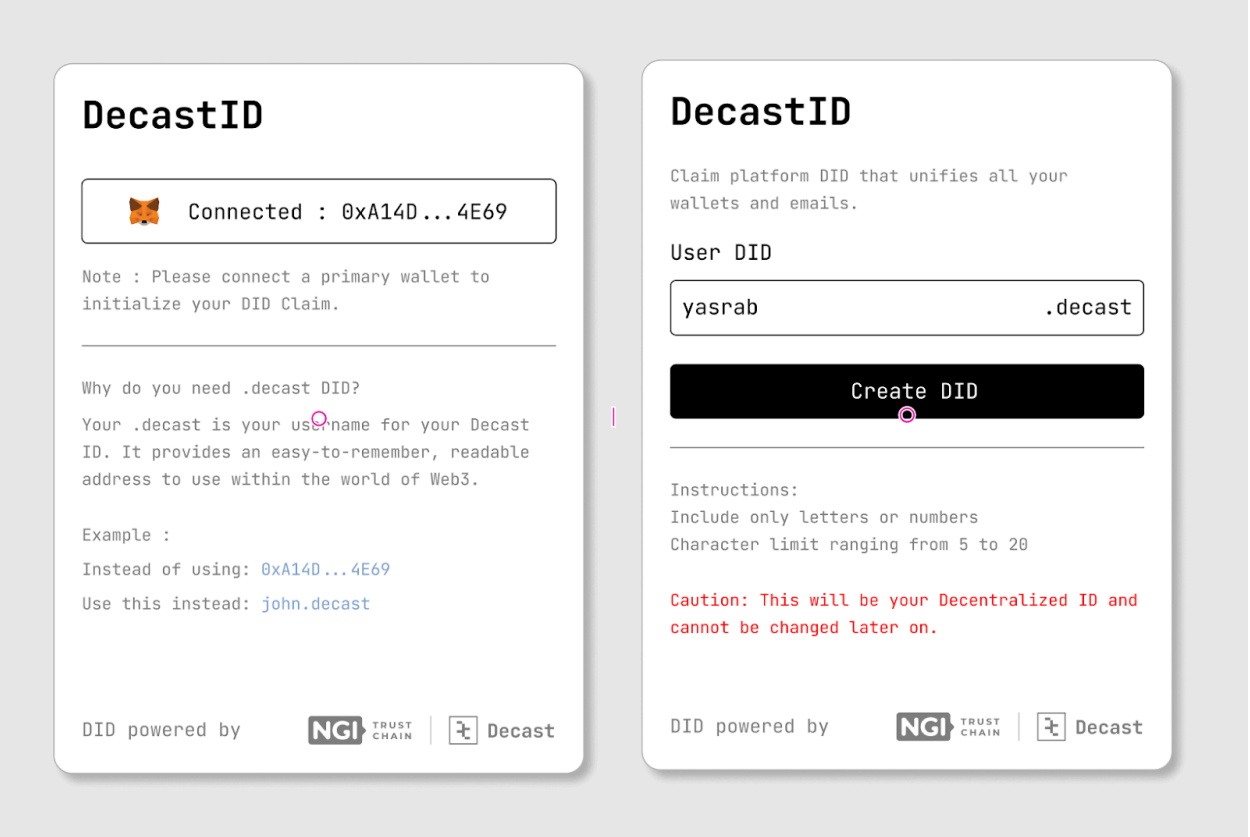

Figure 4 Onboarding Instructions

- **UI/UX Refinements:**
  Navigation was streamlined for easier access to key modules such as secure meeting access and message control.

Funded by the European Union

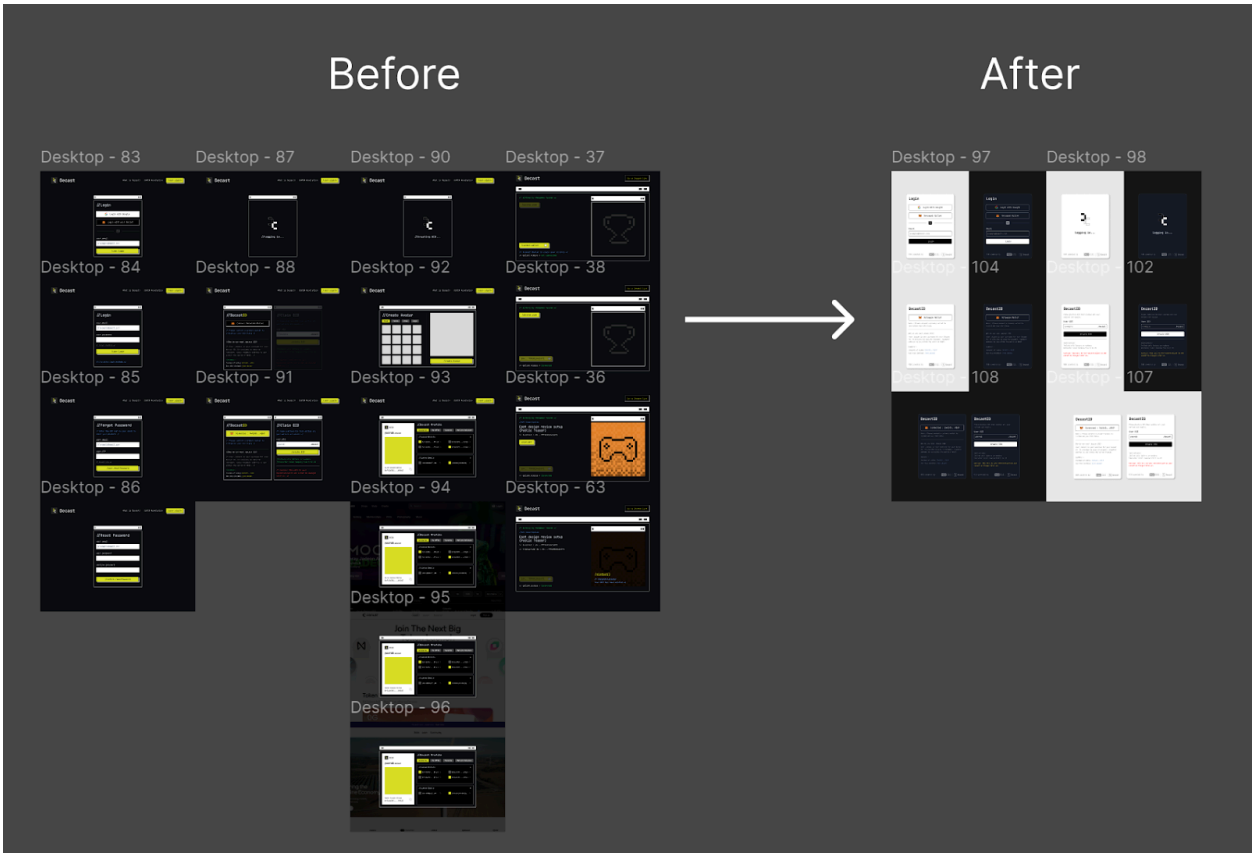

Figure 5 Design Improvements

- **Improved Account Management:**
  The account management processes were redesigned for faster execution and comprehensive data operations.

Funded by the European Union

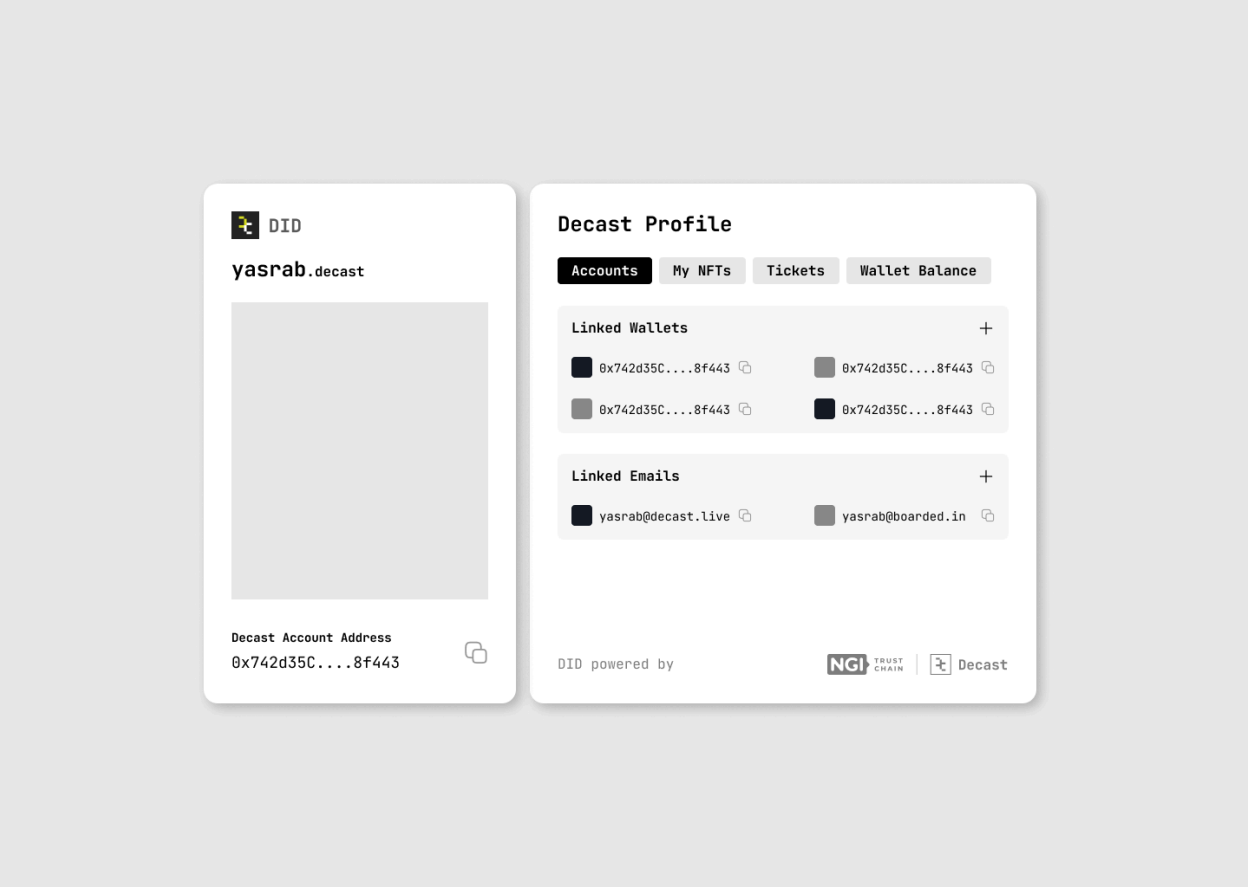

Figure 6 Account Management

- **Open-Source Integration:**
  The platform's design now emphasizes modular, open-source components to keep costs low and allow community contributions.

- **Practical Privacy Settings:**
  New features enable users to set and manage the visibility of their communications easily, addressing the practical concerns of both legal and casual users.

3. SOFTWARE DESIGN AND ANALISYS, COMPONENT SPECIFICATION (FINAL)

Funded by the European Union

This section outlines the architecture of DECAST.LIVE, highlighting its core modules and the integration of decentralized components.

Architecture Overview

Decast will follow a Hexagonal Architecture. The main reason for using it is to **separate the business logic** from the infrastructure. This separation allows us to easily change the database, the way of uploading files, or any other infrastructure without changing the business logic.

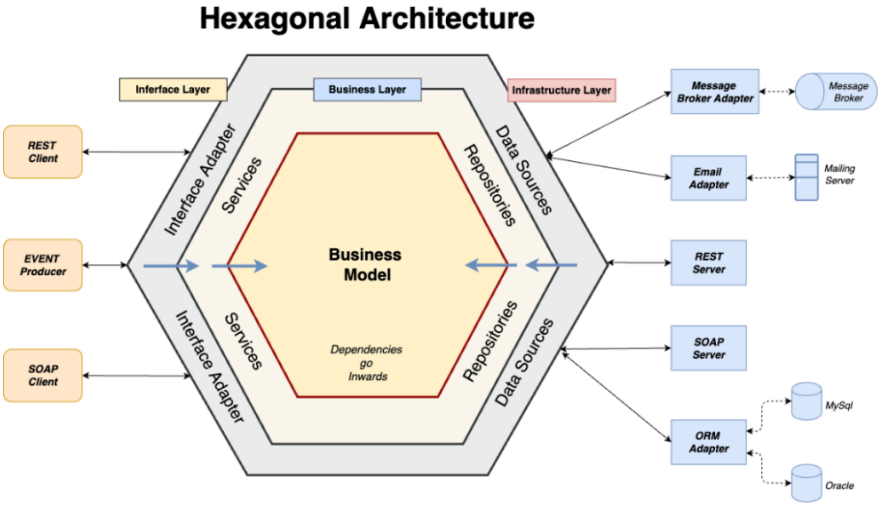

Figure 7 Hexagonal Architecture

This further integrates into Decast's vision of developing a Decentralized Physical Infrastructure (DePIN).

Funded by the European Union

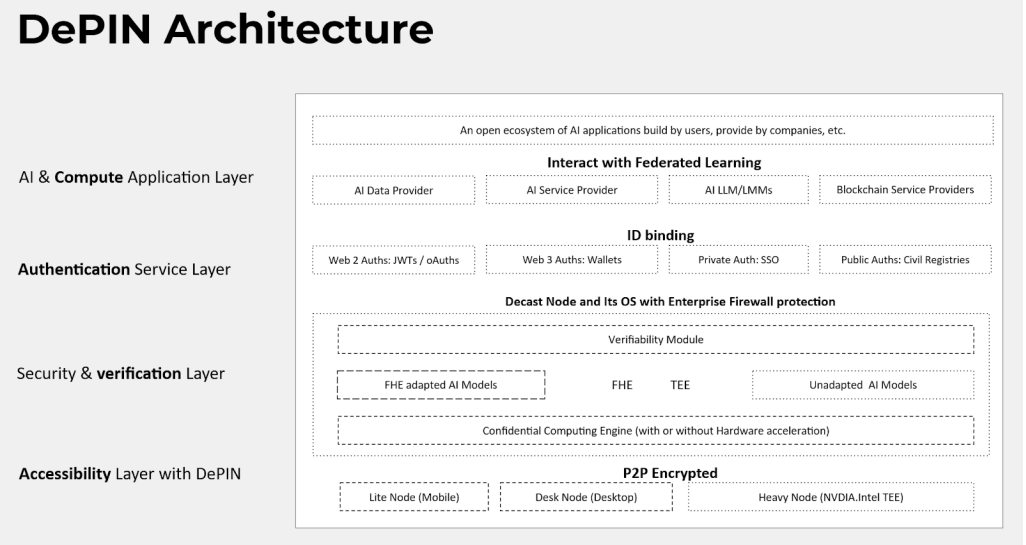

Figure 8 DEPIN Architecture

Decentralized Identifier (DID) Design

Funded by the European Union

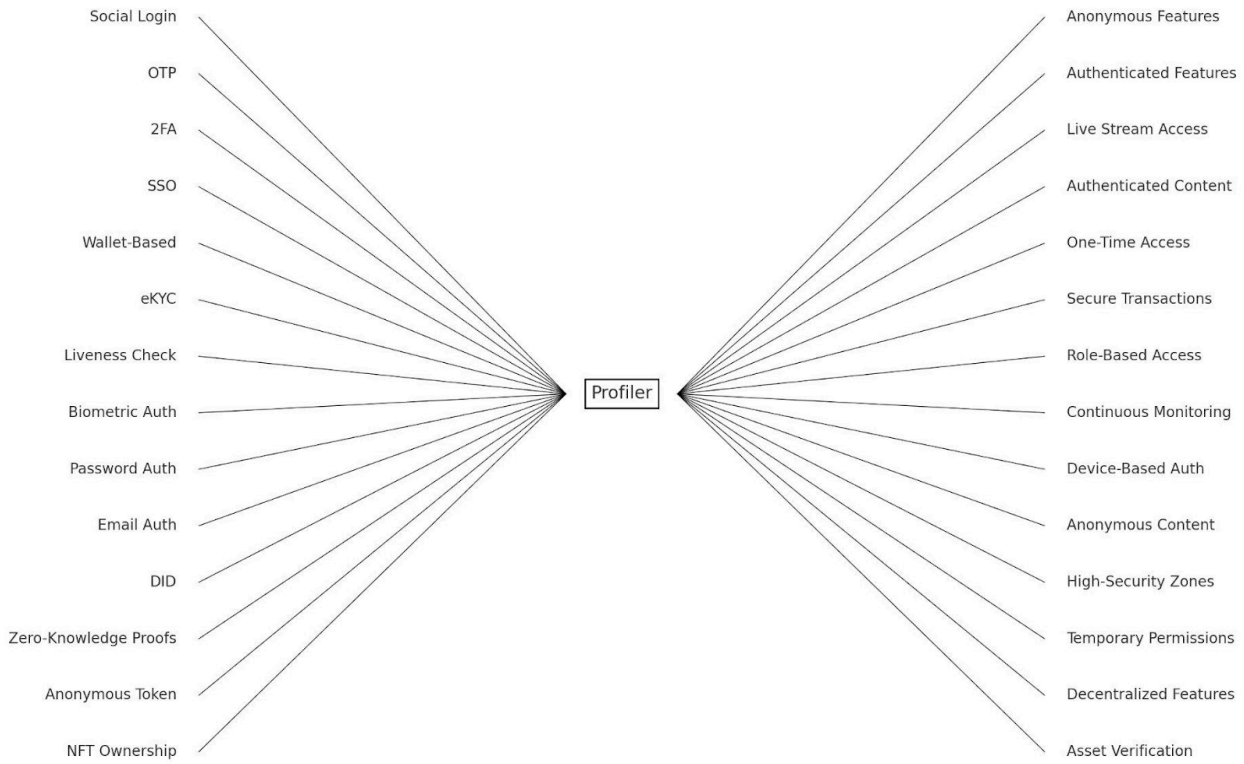

Figure 9 DID Profiler

A **Profiler** as a central entity connects various authentication mechanisms (on the left) with different access control and security features (on the right).

DID-Based Authentication

DIDs enable self-sovereign identity (SSI), allowing users to authenticate without relying on centralized authorities. The left side lists authentication methods that can be tied to a DID-based identity:

- DID (Decentralized Identifier) – A unique, blockchain-based identifier for a user.
- Zero-Knowledge Proofs (ZKP) – Enables privacy-preserving authentication.
- Wallet-Based & Anonymous Tokens – Users can prove identity or ownership using digital wallets without revealing private information.

Funded by the European Union

- NFT Ownership – Proves access rights based on NFT-based credentials.
- Biometric & Liveness Check – Strengthens DID authentication with biometrics.
- eKYC & Social Logins – Links traditional identity verification methods to decentralized IDs.

DID-Enabled Access Control

DIDs allow fine-grained, decentralized access control for various use cases:

- Anonymous & Authenticated Features – Users can interact either anonymously or with verified credentials.
- Live Stream & Content Access – Access can be gated via DID-based permissions.
- Secure Transactions & Role-Based Access – Decentralized identity enables secure, permissioned actions.
- Device-Based & Temporary Permissions – Grant access only to trusted devices and for limited timeframes.
- Decentralized Features – Ensures that access management is distributed and not controlled by a single entity.

A DID-based Profiler can act as a bridge between diverse authentication mechanisms and access control policies, providing privacy, security, and decentralization. Users control their credentials without relying on centralized authorities while enabling seamless access to various digital services.

Application Components

The application follows a microservices architecture implemented using the NestJS framework and a mono-repo structure for the Authentication module (Foreground). The system includes the following key components:

Foreground:

Funded by the European Union

1. **Authentication Module** – Handles authentication, including login via wallet, social etc. authentications.

2. **Shared Library** – Contains reusable utilities, common types, and shared business logic.

3. **API Gateway** – Routes requests and provides a unified interface for clients.

Background

1. **Storage Service** – Manages file storage, retrieval, and access control.
2. **Video Infrastructure** – Works as the interaction layer for all events and casts.

4. SOFTWARE MODULES

Authentication Module

Purpose:

The **Authentication Module** is responsible for handling user authentication using decentralized identifiers (DIDs), social login, and traditional email/password methods. It eliminates the need for traditional username/password authentication by enabling users to log in with their wallets while still supporting conventional authentication methods.

Authorization Strategies (Passportjs.org)

Our authentication system utilizes various JWT-based strategies to manage user authorization and session handling effectively.

- **JWT (DID-JWT) Access Token**: Verify user authentication. Stores userId, role, and sessionId.
- **JWT (DID-JWT) Refresh Token**: Used to obtain a new access token while checking for any user changes in storage. It stores the sessionId and userId.
- **JWT (DID-JWT) 2FA Token**: Used for two-factor authentication (2FA) during login or sensitive actions. It stores the userId.

Funded by the European Union

- **Anonymous Token**: Used for anonymous sessions with predefined, limited permissions. It stores the sessionId.

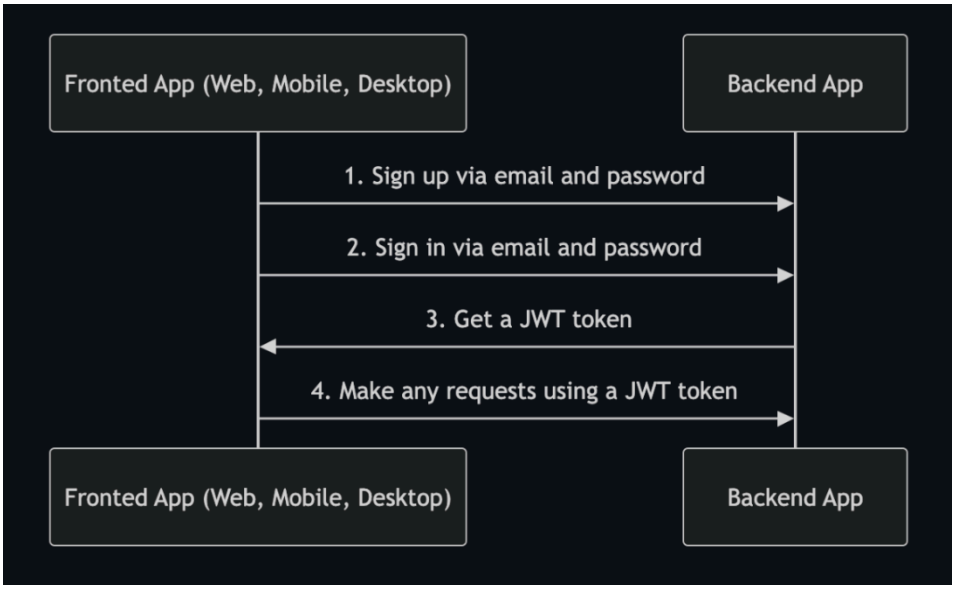

Figure 10 Simple Login Sequence

Planned Development

- **Wallet Authentication**: Allows users to sign authentication requests with their wallets.

Funded by the European Union

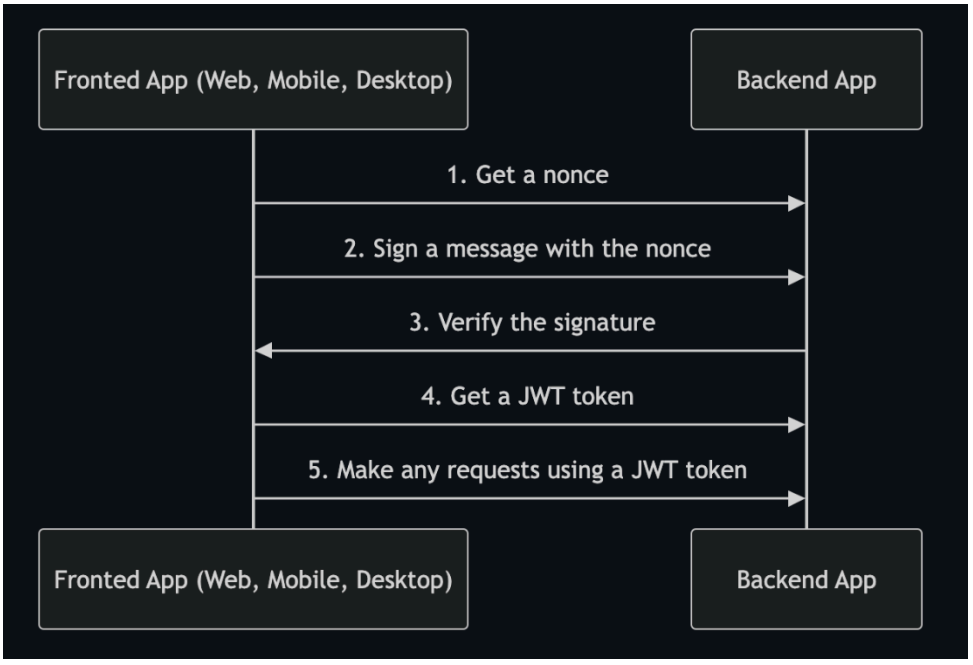

Figure 11 Wallet JWT Login sequence

- **DID Verification**: Ensures the validity of the user's decentralized identity using ethr-did.
- **JWT Issuance**: Generates and signs JSON Web Tokens (JWTs) for authenticated sessions.
- **Session Management**: Manages user sessions, token refresh mechanisms, and expiration policies.
- **Two-Factor Authentication (2FA)**: Provides an additional layer of security using OTP.
- **Social Login**: Supports OAuth 2.0 authentication for Google, Facebook, and other providers.

Funded by the European Union

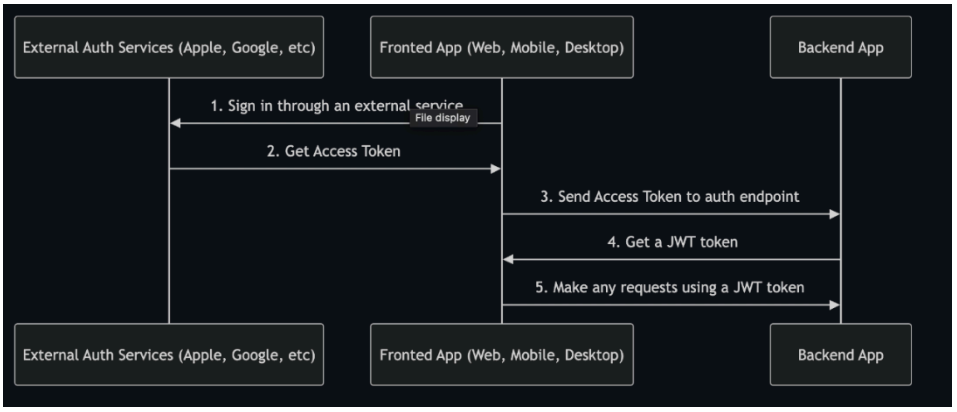

Figure 12 Social JWT Login sequence

- **Email/Password Authentication**: Allows users to register and authenticate using email and passwords.

Contribution:
This module will ensure that only authorized participants can access live streams, thereby meeting the core security and privacy objectives of the platform.

Access Control Module

Purpose:

To establish a plug-and-play authentication framework that supports multiple login methods (wallet-based, social, and traditional) and dynamically enforces access controls based on user roles.

Planned Development:

We intend to develop a microservice using NestJS (within a mono-repo structure) that will manage JWT issuance, session handling, and role-based permissions. This layer will also integrate a lightweight DID management component to support secure user identification for live streaming events.

Contribution:

This module will ensure that only authorized participants can access live streams, thereby meeting the core security and privacy objectives of the platform.

Funded by the European Union

Shared Library & API Gateway

Purpose:

To provide a unified set of reusable utilities, common business logic, and a single entry point for all client requests.

Planned Development:

A standardized library will be created to handle data validation, cryptographic operations, and logging. The API Gateway will be developed to route incoming requests to the appropriate microservices, abstracting internal complexities from external developers.

Contribution:

It will simplify future integrations, accelerate new feature development, and offer a consistent API interface across the platform.

DID Management and Profiler Package

Purpose:

Although not the primary focus, this component is designed to support secure access by managing decentralized identifiers (DIDs) and associated user profiles.

Planned Development:

A dedicated package will be built to interface with the authentication module, allowing for the resolution and verification of user credentials with minimal overhead on the core video infrastructure.

Contribution:

This package will enhance overall security by linking user identities to streaming sessions, while allowing customizable privacy settings without impacting video performance.

Funded by the European Union

4.1 ARCHITECTURE DIAGRAM

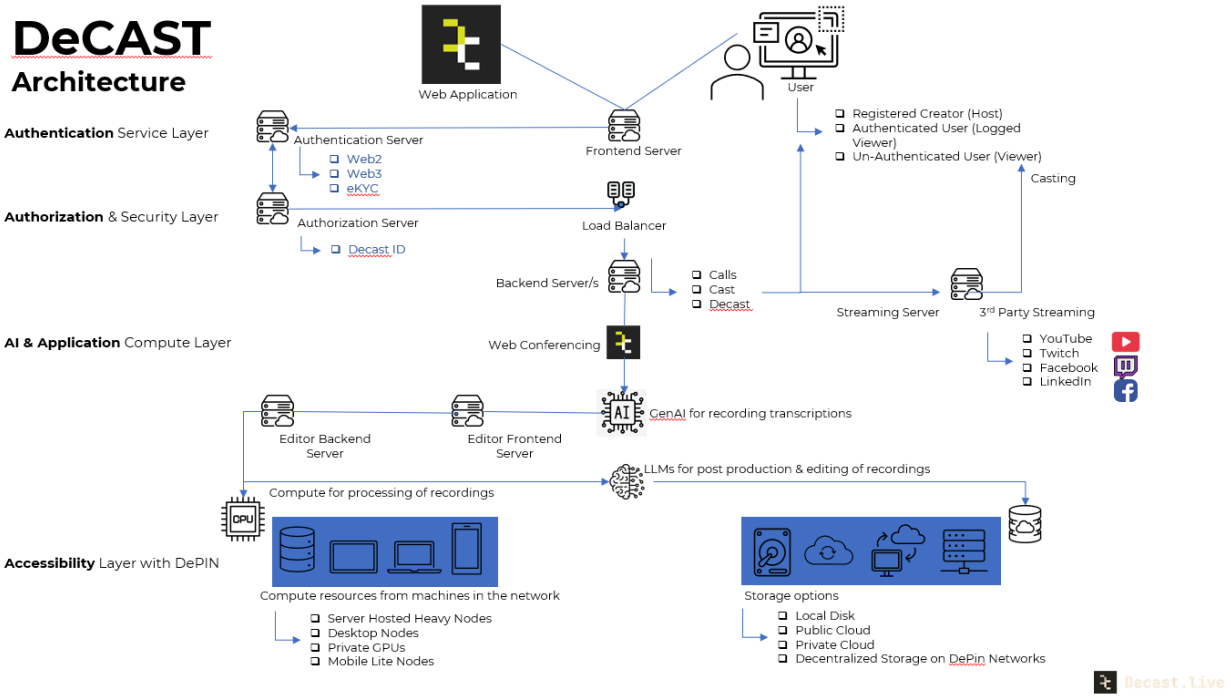

Figure 13 Application Architecture

The architecture of the video infrastructure integrates multiple interconnected components to deliver a seamless and secure solution.

Foreground

Modular Authentication Layer: We designed a plug-and-play authentication module that interfaces directly with the live streaming engine. This module is responsible for dynamically selecting authentication mechanisms—such as social logins, OTPs, biometric checks, and wallet-based verifications—based on the event context and user role.

DID Management and Profiler Package: At the core of our approach is the integration of Decentralized Identifiers (DIDs). We developed a dedicated DID management system that leverages self-sovereign identity principles, enabling users to control their credentials via a distributed ledger.

Funded by the European Union

Background

Web Application serves as the user interface, connecting to **Backend Servers** that handle core functionalities like calls, casting, and streaming.

Streaming Server supports live casting to platforms like YouTube and Twitch, enabling broad content distribution. Together, these layers ensure scalability, decentralized infrastructure, and secure interactions, providing a robust alternative to centralized solutions.

Foreground + Background

Storage module leverages decentralized storage options like Sia, ethSwarm etc. to ensure data privacy, tamper-proof storage, and autonomy for its users while also providing traditional storage options.

Future Work

Advanced features, such as real-time transcription and post-production editing, are powered by the **AI & Compute Application Layer**, leveraging decentralized compute resources from the **Accessibility Layer with DePIN**, which utilizes server nodes, GPUs, and decentralized storage networks.

Funded by the European Union

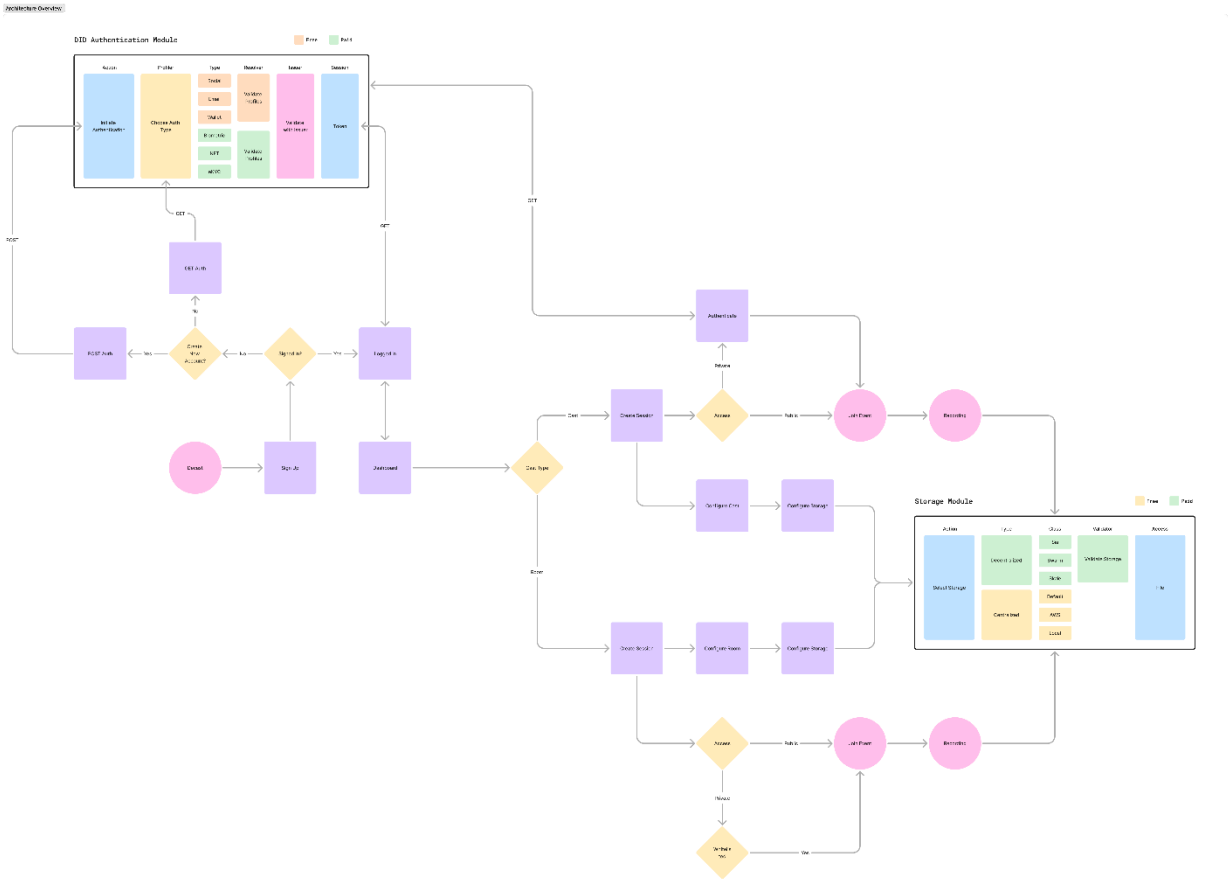

Figure 14 Component Interaction

System Interaction Flow

1. A user attempts to log in via their wallet.
2. The request is processed by the Auth Microservice, verifying the DID JWT.
3. Upon successful authentication, an access token is issued.
4. The user interacts with the Storage Microservice to upload or retrieve files.
5. The Storage Microservice verifies user permissions and processes the request accordingly.

Funded by the European Union

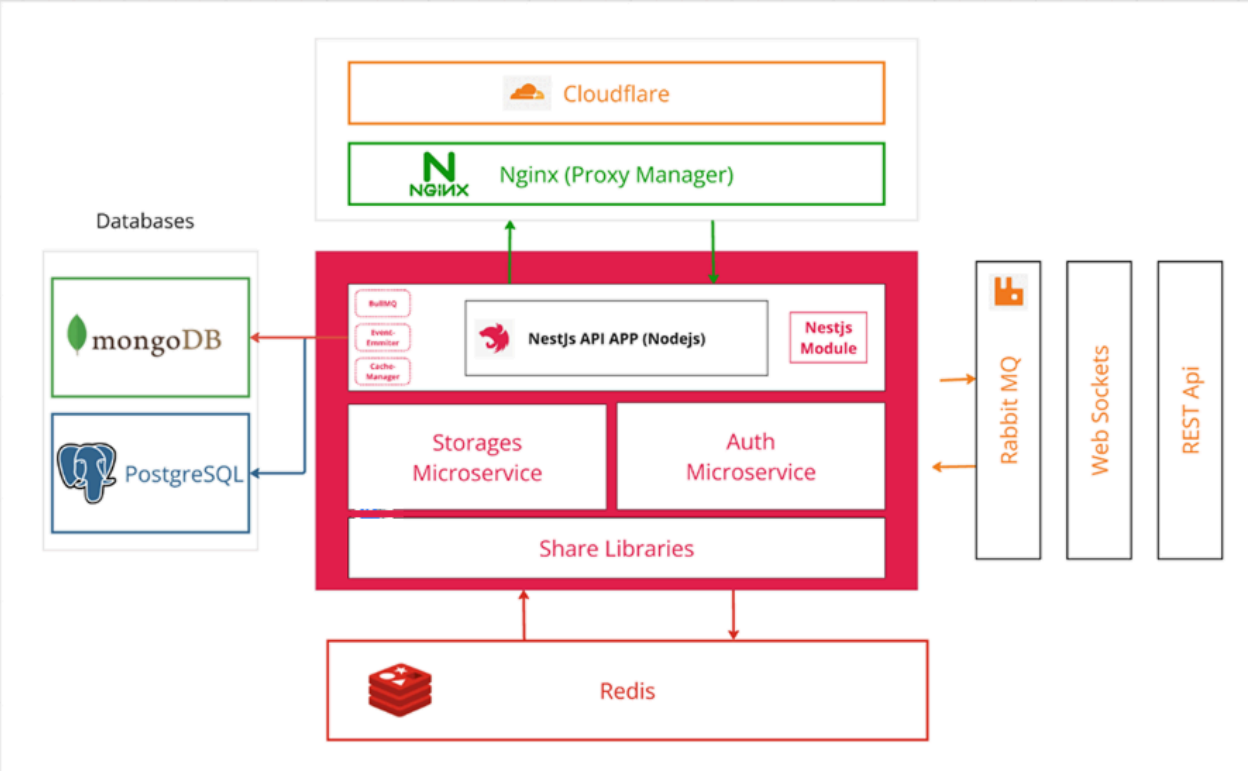

Figure 15 System Interaction Flow

5. DETAILED API SPECIFICATION (PRELIMINARY)

5.1 API SPECIFICATION FOR SDK MODULES

- **Decast DID Resolver API:** Handles DID resolution and signature verification.
- **Decast DID Wallet SDK:** Provides functions for DID creation, credential management, and cross-chain synchronization.
- **Decast Web Wallet:** Manages web-based blockchain interactions and handles credential operations.
- **Decast Mobile Wallet:** Enables secure mobile authentication, including

Funded by the European Union

biometric and QR-based methods, and supports credential issuance.

5.1.1 DECAST DID RESOLVER API

Communication Protocols & Security

- **DID Resolution Protocol:** Uses decentralized identity resolvers compatible with did:decast.
- **Cryptographic Security:** Verifies signatures using ECDSA-based public-private key cryptography.
- **Data Integrity:** Ensures immutability by leveraging blockchain or decentralized storage systems.
- **Scalability:** Designed to work efficiently across multiple nodes in a decentralized network.

Table 1 DECAST DID RESOLVER API

Function	Arguments	Return value	Description
resolveDid(did: string)	- did: (string) The DID to be resolved, in the format did:decast:publicKey.	(object) JSON representation of the DID document.	Resolves a did:decast identifier and returns its corresponding DID document, including public keys, authentication methods, and service endpoints.
verifyDidSignature(did: string, signature: string, message: string)	- did: (string) The DID whose signature is to be verified.	(boolean) True if the signature is valid; false otherwise.	Verifies that the given signature was generated by the public key

Funded by the European Union

	- signature: (string) The cryptographic signature to verify. - message: (string) The signed message.		associated with the provided DID for the specified message.
getPublicKey(did: string)	- did: (string) The DID whose public key is to be retrieved.	(string) The public key associated with the DID.	Extracts and returns the public key from the resolved DID document.
registerDid(did: string, publicKey: string, serviceEndpoints?: object)	- did: (string) The DID to register. - publicKey: (string) The public key associated with the DID. - serviceEndpoints (optional): (object) Additional endpoints linked to the DID.	(boolean) True if registration is successful; false otherwise.	Registers a new did:decast identifier in the decentralized storage system by associating it with a public key and optional service endpoints.
updateDidDocument(did: string, newPublicKey?: string, serviceEndpoints?: object)	- did: (string) The DID whose document is updated. - newPublicKey (optional): (string) New public key. - serviceEndpoints (optional): (object) New service endpoints.	(boolean) True if the update is successful; false otherwise.	Updates the DID document with a new public key or service endpoints, ensuring security and immutability constraints.

Funded by the European Union

deleteDid(did: string)	- did: (string) The DID to be removed.	(boolean) True if deletion is successful; false otherwise.	Deletes the specified did:decast identifier from the decentralized storage system.

Example Usage

```
const { resolveDid, verifyDidSignature } = require('decast-did-resolver');


(async () => {
  const didDocument = await resolveDid("did:decast:0x123abc...");
  console.log("DID Document:", didDocument);


   const isValid = verifyDidSignature("did:decast:0x123abc...", "0xsignature...", "message to verify");
  console.log("Signature Valid:", isValid);
})();
```

This specification provides a robust and secure DID resolution mechanism for did:decast, ensuring compatibility with decentralized applications and blockchain-based authentication systems.

5.1.2 DECAST DID WALLET SDK

DID Interoperability: Supports did:decast resolution across multiple chains.

Security Mechanisms:

Funded by the European Union

- Uses ECDSA-based cryptographic verification.
- Secure storage for private keys and credentials.

Scalability:

- Designed for multi-chain environments (Ethereum, Polygon, SKALE, etc.).

Table 2 DECAST DID WALLET SDK

Function	Arguments	Return value	Description
createDID(publicKey: string)	- publicKey: (string) The public key to associate with the DID.	(string) The generated DID in the format did:decast:publicKey.	Generates a new DID for a user based on the provided public key.
resolveDID(did: string)	- did: (string) The DID to be resolved.	(object) The DID document associated with the given DID.	Resolves the given DID and returns its corresponding document.
signMessage(did: string, message: string)	- did: (string) The DID of the signer. - message: (string) The message to sign.	(string) A cryptographic signature.	Signs a message using the private key associated with the specified DID.
verifySignature(did: string, signature: string, message: string)	- did: (string) The DID of the signer. - signature: (string) The cryptographic signature to verify. - message: (string) The original message.	(boolean) True if the signature is valid; false otherwise.	Verifies that the signature corresponds to the given DID's public key for the specified message.

Funded by the European Union

storeCredential(did: string, credential: object)	- did: (string) The DID to associate with the credential. - credential: (object) The Verifiable Credential to store.	(boolean) True if stored successfully; false otherwise.	Stores a Verifiable Credential (VC) in decentralized storage associated with the given DID.
retrieveCredential(did: string)	- did: (string) The DID whose credentials are to be retrieved.	(array) A list of stored Verifiable Credentials.	Retrieves all Verifiable Credentials associated with the specified DID.
syncDIDAcrossChains(did: string, targetChain: string)	- did: (string) The DID to synchronize. - targetChain: (string) The blockchain network to sync with.	(boolean) True if synchronization is successful; false otherwise.	Synchronizes the specified DID and its associated roles across multiple blockchain networks.

Example Usage

```
const { createDID, resolveDID } = require('decast-did-wallet-sdk');


const did = createDID("0x123abc...");

console.log("New DID:", did);


const didDocument = resolveDID(did);
```


Funded by the European Union

```
console.log("DID Document:", didDocument);
```

5.1.3 DECAST WEB WALLET

Table 3 DECAST WEB WALLET

Function	Arguments	Return value	Description
connectWallet(provider: object)	- provider: (object) The web3 provider for blockchain interactions.	(boolean) True if connection is successful.	Establishes a connection between the Web Wallet and the blockchain.
getUserDID()	None.	(string) The DID linked to the user's blockchain account.	Retrieves the DID associated with the connected user's blockchain account.
manageCredentials(action: string, credential?: object)	- action: (string) The operation to perform (store, retrieve, delete). - credential (optional): (object) The credential data.	(object array)	Credential data or confirmation.
signAndVerifyTransaction(data: object)	- data: (object) The transaction details.	(boolean) True if the transaction is verified; false otherwise.	Signs a transaction and verifies its integrity within the Web Wallet.

Funded by the European Union

Example Usage

```
const { getUserDID } = require('decast-web-wallet');


const userDid = getUserDID();

console.log("User DID:", userDid);
```

5.1.4 DECAST MOBILE WALLET

Table 4 DECAST MOBILE WALLET

Function	Arguments	Return value	Description
generateNewDID()	None.	(string) The newly created DID.	Generates a new did:decast identifier for the user via the mobile wallet.
authenticateWithBiometrics()	None.	(boolean) True if authentication is successful; false otherwise.	Enables biometric authentication for secure DID access on the mobile platform.
scanQRForDIDAuth(qrData: string)	- qrData: (string) The content of the QR code scanned.	(object) The authentication response.	Scans a QR code to authenticate the user and retrieve associated DID details.
issueCredential(credential: object)	- credential: (object) The	(boolean) True if the credential is	Issues a new Verifiable

Funded by the European Union

	credential to be issued.	issued successfully; false otherwise.	Credential via the mobile app.
upgradeDIDRole(did: string, nftId: string)	- did: (string) The DID to upgrade. - nftId: (string) The NFT representing the new role.	(boolean) True if the upgrade is successful; false otherwise.	Upgrades the user's DID role using NFT-based permissions.

Example Usage

```
const { authenticateWithBiometrics } = require('decast-mobile-wallet');


const authSuccess = authenticateWithBiometrics();

console.log("Biometric Authentication Successful:", authSuccess);
```

5.2 API SPECIFICATION FOR REST SERVICES

This section describes the REST API endpoints for Decast's services. Each endpoint is defined with its HTTP method, URI, arguments, return value, and a brief description. All endpoints require an Authorization header with a bearer token (except for public endpoints).

5.2.1 AUTH MODULE

Description: Endpoint: https://<base_uri>/api/v1/

Headers:

Header Name: "**Authorization**"

Header Value: `**bearer <token>**`

Funded by the European Union

startegies

https://www.passportjs.org/
https://jwt.io/

Strategy	payload type	generate on:	validate by:

2FA	{
	id: string | numebr;
	twoFactorAuthentication: true;
	iat: number;
	exp: number;
	}
		• logged in & user.twoFactorAuthentication == true	payload.id && payload.twoFactorAuthentication

Refresh Token	{
	sessionId: Session['id'];
	hash: Session['hash'];
	iat: number;
	exp: number;
	}
		• logged in & user.twoFactorAuthentication != true
		• 2fa verified
			!payload.id && payload.sessionId

Token	{
	id: string | numebr;
	role: Role { id: number; name: string; };
	sessionId: Session['id'];
	iat: number;
	exp: number;
	}
		• logged in & user.twoFactorAuthentication != true
		• 2fa verified
			payload.id && payload.sessionId

- **Token:** Successful Authentication token that used for all apis. Expires in 15m (jwt, did-jwt). Renew by refresh token
- **2FA**: need for 2nd step login (if 2FA enabled). Expires in 5m. (jwt, did-jwt).
- **RefreshToken**: to renew the token and refresh token. Expires in 30d. (jwt, did-jwt).

JSON Objects

- User
  - Id
  - FirstName
  - LastName
  - …
- Login Response
  - Token
  - refreshToken
- Login 2FA Response (if 2FA enabled)
  - TwoFactorEnabled
  - 2faToken

Funded by the European Union

Table 5 AUTH Module API

HTTP method	URI	Arguments	Return value	Description
POST	auth/google/login	(string) **idToken**	(JSON) Returns the **login response**, including tokens and user details.	Authenticates a user using their Google account. **idToken** is from Google callback.
POST	auth/google/connect	(string) **idToken**	(JSON) Returns **the user details** after connecting their Google account.	Connects a user's Google account to their existing account.
POST	auth/facebook/login	(string) **accessToken**	(JSON) Returns the **login response**, including tokens and user details.	Authenticates a user using their Facebook account. **accessToken** is from Facebook callback.
POST	auth/facebook/connect	(string) **accessToken**	(JSON) Returns **the user details** after connecting their Facebook account.	Connects a user's Facebook account to their existing account.
POST	auth/apple/login	(string) **idToken** (string) **firstName?** (string) **lastName?**	(JSON) Returns the **login response**, including tokens and user details.	Authenticates a user using their Apple account. **idToken** is from Apple callback.
POST	auth/apple/connect	(string) **idToken** (string) **firstName?** (string) **lastName?**	(JSON) Returns **the user details** after connecting their Apple account.	Connects a user's Apple account to their existing account.

Funded by the European Union

POST	auth/did/login	(JSON) **didDocument**	(JSON) Returns the **login response**, including tokens and user details.	Validate the document by related resolver and verify it. Login user or create a Did Document. Check issuer for more options
POST	auth/wallet/nonce	(string) **walletAddress**	(string) **nonce**	Generates a nonce for wallet-based authentication.
POST	auth/wallet/login	(string) **nonce** (string) **signature**	(JSON) Returns the **login response**, including tokens and user details.	Authenticates a user using their wallet
POST	auth/wallet/connect	(string) **nonce** (string) **signature**	(JSON) Returns **the user details.**	Connects a wallet to their existing account.
POST	auth/2fa/mail/send-otp		(boolean) Return true or raise err (ex: method not activated by user found)	Sends a one-time password (OTP) to the user's email for 2FA.
POST	auth/2fa/mail/verify-otp	(string) **code**	(JSON) Returns the **login response** after verifying the OTP.	Login 2nd factor login when 2FA enabled y user. Using email component for verification.
POST	auth/2fa/sms/send-otp		(boolean) Return true or raise err (ex: method not activated by user found)	Sends a one-time password (OTP) to the user's mobile for 2FA.
POST	auth/2fa/sms/verif	(string) **code**	(JSON) Returns the **login response**	Login 2nd step login when 2FA enabled y

Funded by the European Union

	y-otp		after verifying the OTP.	user. Using sms component for verification.
POST	auth/2fa/totp/verify-otp	(string) **code**	(JSON) Returns the **login response** after verifying the OTP.	Login 2nd step login when 2FA enabled y user. Using TOTP (Google Authenticator, ...) component for verification.
GET	Auth/2fa/method-types		(JSON) Returns array of available method: (string) **methodId** (string) **methodName** (string) **inputTitle** (mobile, email, TOTP secret, ...) (string) **triggerActionToSend** (Send SMS, Send Email, ...)	Get all available 2FA methods can be activated for user.
GET	Auth/2fa/method		(JSON) Returns array of active method: (string) **id** (string) **methodId**(string) **inputValue** (mobile, email,)	Get User all activated 2FA methods
POST	auth/2fa/add	(string) **type** (string) **inputValue** (mob. No, ...)	(JSON) Returns the new method that added.	Enable 2FA on login and add a new 2FA method.

Funded by the European Union

DELETE	Auth/2fa/delete/<id>			Deletes the method and deactivate it for user.
GET	auth/me		(JSON) Returns the authenticated **user's details.**	Retrieves the details of the currently authenticated user.
PATCH	auth/me	(JSON) user partial editable data	(JSON) Returns the updated authenticated **user's details.**	Updates the details of the currently authenticated user
POST	auth/refresh		(JSON) Returns new access and refresh tokens.	Refreshes the user's access token using a valid refresh token.
POST	auth/logout			Logs out the currently authenticated user by invalidating their tokens.

This comprehensive API specification for both SDK modules and REST services ensures that developers have clear, consistent instructions for integrating with the Decast.live platform. All functions, endpoints, and expected behaviors have been detailed using the provided template, enabling robust, scalable, and secure interactions within our decentralized video and identity management ecosystem.

6. DETAILED PLAN FOR IMPLEMENTATION AND DEPLOYMENT (FINAL)

6.1 PLAN FOR IMPLEMENTATION

Each phase focuses on technical tasks, resource allocation, and tech integration.

Funded by the European Union

Phases:

- **Design Finalization:**
  - Lock down architecture, update design diagrams, and assign tasks.
  - Finalize API specs for microservices (Auth, Storage, Shared Library).
- **Modular Prototyping:**
  - Develop individual microservices using NestJS.
  - Integrate DID-based authentication (using did-jwt, ethr-did-resolver, did-key.js) and JWT management (access, refresh, 2FA).
  - Implement cryptographic components (e.g., ZKPs) for privacy.
- **Integration & Testing:**
  - Perform unit, integration, and load tests (simulate high concurrency on AWS EC2/S3/RDS).
  - Use Docker for containerization; automate tests via CI/CD pipelines.
- **Documentation & Validation:**
  - Update API docs and technical specs.
  - Validate through pilot runs; iterate based on feedback.

Key Outputs:

- Prototype demonstrating core functionalities.
- Initial API specifications and integration tests.
- Verified system performance and security benchmarks.
- Updated documentation reflecting testing outcomes and user feedback.
- Complete technical documentation.
- A validated solution ready for staged deployment.

Tools & Technologies:

- **Development Framework:** NestJS, TypeScript
- **Authentication Libraries:** did-jwt, ethr-did-resolver, did-key.js
- **Infrastructure:** Docker containerization, AWS EC2/S3/RDS for hosting storage
- **Testing:** Automated testing frameworks, CI/CD pipelines

Key milestones, tasks, and timelines are outlined below:

Funded by the European Union

Table 6 WORK PLAN TASKS AND TIMELINE

Work plan tasks	Description	Starting Month	Ending Month
Project Initiation and Research	Requirement Gathering, Market Analysis, Technical Feasibility Analysis, Team Formulation	Nov 2024	Jan 2025
Prototyping and Development	Development of the Prototype, Blockchain Integration, DID Implementation and Testing, Security Testing, User Interface Design and Integration	Jan 2025	March 2025
Testing and Iterative Refinement	Beta Release Usability Testing Scalability Assessment, Identity Integration, Video Integration Security Audit	March 2025	April 2025
Deployment and Marketing	Full Release Marketing Campaigns User Acquisition and Promotion Documentation and Support	May 2025	June 2025
Ongoing Optimization and Future Planning	Continuous Improvement Future Roadmap User Training and Deployment	June 2025	July 2025

Funded by the European Union

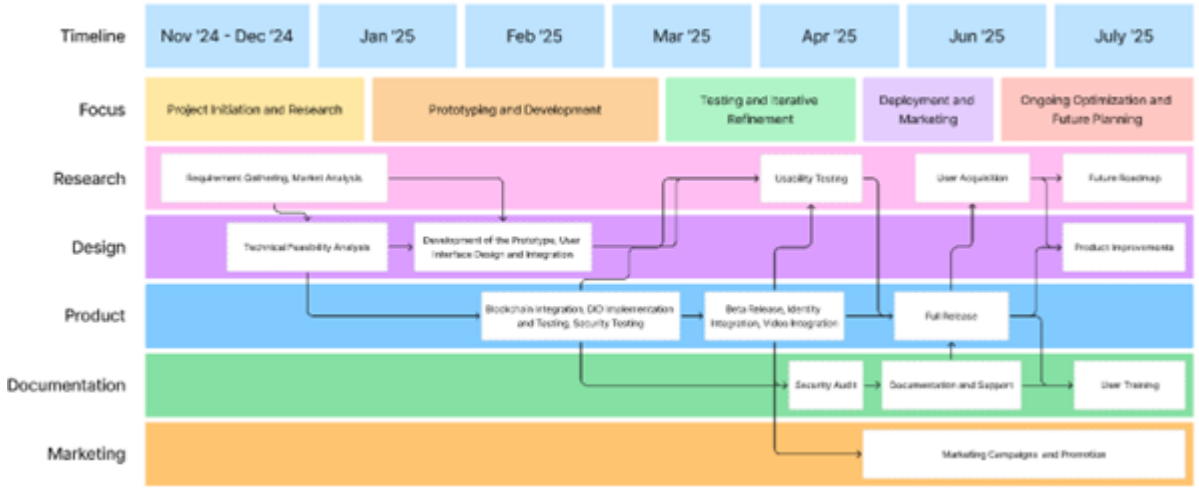

Figure 16 Gantt Chart of Workplan

Resources and Role Allocation

Table 7 RESOURCES AND ROLE ALLOCATION

Name	Role	Social Profile
Shivam Dhawan	Project Coordinator	https://www.linkedin.com/in/shivamdhawan/
Peyman Pourjafar	Development Lead & Manager	https://www.linkedin.com/in/aman-j-bishnoi/
Mohammed Yasrab	Product Manager & Designer	https://www.linkedin.com/in/mohammed-yasrab/
Anish Jha	Application Developer	https://www.linkedin.com/in/anish-jha7/

Funded by the European Union

M. Ajmal Azad	Blockchain Researcher	https://www.linkedin.com/in/sushil-kumar-gupta/
Sushil Gupta	Application Developer	https://www.linkedin.com/in/muhammad-ajmal-azad-25b34717/

Deliverables and Milestones

Table 8 DELIVERABLES AND MILESTONES

No.	Deliverable or milestone name	Description	Type	Delivery Month	TRL level delivered
1	Requirement Analysis	A requirement analysis and design document will be released	Document	Jan 2025	7
2	Feasibility and Market Analysis	A Market analysis report will be released after studying major competitors and performing requirement and need analysis	Report	April 2025	7
3	Implementation, Testing and Analysis	A detailed report will be released that will cover all functional, non-functional, and security use cases. The report	Report	June 2025	8

Funded by the European Union

		will cover all implementation and testing details/scenarios.			
4	Product Release	An open-source codebase will be released	Tool/code repository	June-July 2025	7
5	Research paper and final report	Project documentation will be released.	Documentation	July 2025 (Document) and July-August (Research Paper)	10

6.2 PLAN FOR DEPLOYMENT

We adopt a phased deployment approach that minimizes risks and facilitates continuous feedback.

Phases:

- **Pilot Deployment:**
  - Roll out in a controlled AWS test environment.
  - Monitor system performance with AWS CloudWatch or Sentry.
- **Scaling & Orchestration:**
  - Containerize services with Docker; manage via Kubernetes if needed.
  - Set up CI/CD for continuous integration, deployment, and rollbacks.
- **User Onboarding & Monitoring:**
  - Provide concise user guides; integrate feedback loops.
  - Track performance and errors in real-time; plan quick fixes.

Deployment Steps

Funded by the European Union

Proposed DECAST Live platform will be released into the production environment during deployment phase involving initial deployment, integration and configuration, feature activation and user onboarding activities.

- **Initial Deployment:** Firstly, core components of the DECAST Live e.g. identity Verification, Blockchain integration, Implementation of Zero Knowledge proofs and integration with online video sharing platform. Following by the setup of software on servers, integration with blockchain and correct configuration of platforms modules

- **Configuration and Integration:** After this initial deployment of system components, configuration of system setting and integration of components will be performed. This involves building connection between ZK proof modules and blockchain based identity wallet and storage of Video.

- **Feature Activation:** Key functionalities of the system, such as wallet management, video broadcasting, user interaction will be activated. Correct working of these functionalities and their compliance with benchmark performance will be ensured.

- **User Onboarding:** Initial users will be onboard and necessary access credentials will be provided to them. After creating user's accounts, system's initial data will be imported to test the system's functionalities.

Key Outputs:

- Successfully deployed pilot followed by a full-scale production release.
- Comprehensive operational documentation and user training materials.

Challenges and Solutions

- **Dynamic Integration:** Selecting the appropriate authentication method on-the-fly without disrupting the user experience will be a major challenge. We will address this by implementing a middleware profiler that will classify events and user roles using nested segmentation, ensuring seamless and context-aware authentication.

- **System Interoperability:** Integrating the new authentication layer with existing decentralized storage systems (ethSwarm, Sia) and live casting components will require careful design to avoid disruption. We will solve this by developing a well-defined API that will allow the authentication module to

Funded by the European Union

operate independently while interfacing smoothly with the core platform.

- **Balancing User Experience and Security:** Implementing stringent security measures without overburdening users will be a delicate balance. Our adaptive framework will ensure that low-risk scenarios offer frictionless entry while higher-risk events trigger additional security layers, maintaining an optimal balance between usability and protection.

6.3 RISK ANALYSIS

1. **Integration Complexity:**
   - **Risk:** Challenges in seamlessly integrating multiple authentication methods and blockchain components may delay progress.
   - **Mitigation**: Modular testing, parallel development, team sync-ups.

2. **System Interoperability:**
   - **Risk:** Integrating the new authentication layer with existing decentralized storage systems (EthSwarm, Sia) and live casting components will require careful design to avoid disruption.
   - **Mitigation:** By developing a well-defined API that will allow the authentication module to operate independently while interfacing smoothly with the core platform.

3. **Performance Bottlenecks:**
   - **Risk:** High concurrent access during live streaming events could lead to latency or system overload.
   - **Mitigation:** Load testing, auto-scaling (AWS), Redis for caching.

4. **Security Vulnerabilities:**
   - **Risk:** Potential breaches in authentication mechanisms, including wallet/DID integrations and token issuance.
   - **Mitigation:** Penetration tests, vulnerability scans, access controls.

5. **User Adoption & Feedback:**
   - **Risk:** Low initial adoption or negative feedback could hinder the scaling process.

Funded by the European Union

- **Mitigation:** Early pilot feedback, iterative UI/UX improvements, and support documentation.

6. **Third-Party Dependencies:**
   - **Risk:** Reliance on external APIs or blockchain components might introduce unforeseen challenges.
   - **Mitigation:** Build fallback mechanisms, redundancy plans, and monitor external API updates.

7. BUSINESS MODEL AND EXPLOITATION PLAN (PRELIMINARY)

7.1 BUSINESS MODEL DESCRIPTION

Cost Structure

Fixed Costs

- **Platform Development & Testing**
  - Ongoing software engineering to maintain a stable, feature-rich video platform.
  - Rigorous QA to ensure performance under high-concurrency live streaming.
- **Deployment & Operations**
  - Cloud infrastructure costs (e.g., AWS) for hosting and scaling live streams.
  - DevOps overhead for containerization, orchestration, and system maintenance.
- **Marketing & Outreach**
  - Promotional activities to attract event organizers, educators, and enterprise clients.
  - Partnership-building with relevant communities (e.g., educational institutions, corporate networks).

Variable Costs

Funded by the European Union

- **Scalability & Support**
  - Bandwidth and storage expenses grow with user volume and streaming demands.
  - Customer support costs scale with the number of live events and participants.
- **Third-Party Integrations**
  - Fees for optional decentralized storage or blockchain-based features (e.g., NFT minting, advanced identity verification).

Revenue Streams

Subscription Tiers

- **Explorer (Free):** Limited concurrency (up to 50 viewers), basic storage (3-day retention), no AI or monetization tools.
- **Creator (€35/month):** Ideal for mid-size events (up to 150 viewers), basic AI-driven post-production, NFT ticketing, and moderate monetization.
- **Developer (€89/month):** Suited for advanced use cases (up to 1,000 viewers), full AI tools, custom integrations, and priority support.
- **Enterprise (Custom):** Large-scale deployments, white-labeling, extended concurrency, and dedicated account management.

Pay-Per-Event or Credit Packages

- Event organizers purchase credits for occasional, high-impact events without committing to a monthly plan.

NFT-Based Access (Optional)

- Users can mint or trade Access NFTs for premium or long-term usage, creating secondary market possibilities.

Key Partnerships

- **Decentralized Storage & Web3 Providers:** For optional off-chain or blockchain-based content hosting.
- **Educational & Corporate Alliances:** Institutions and businesses seeking secure, large-scale streaming solutions.
- **Open-Source Community & Tech Partners:** Collaboration on feature enhancements, shared R&D, and plugin-based integrations.

Funded by the European Union

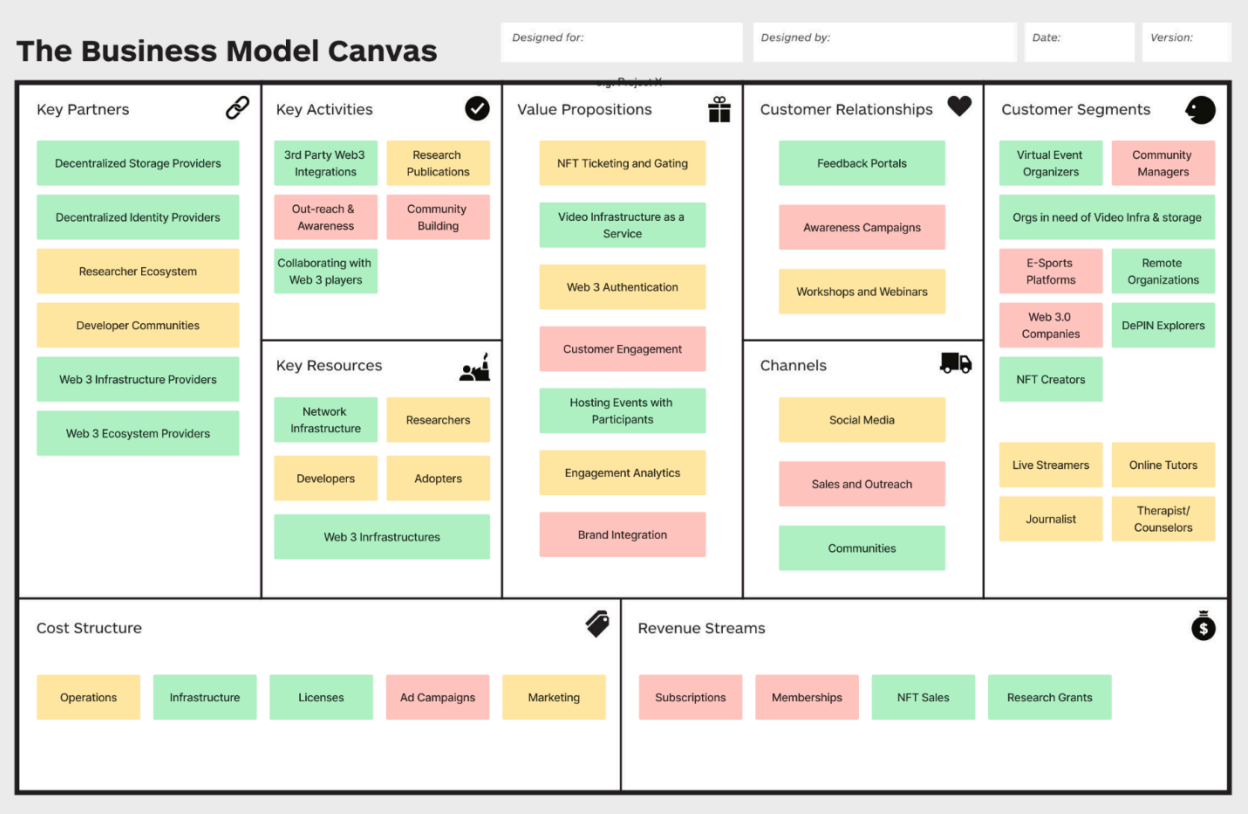

Figure 17 Business Model Canvas

7.2 BUSINESS VALUE FOR THE BLOCKCHAIN AND SSI DOMAIN IN GENERAL

Tangible Contributions

- **Improved Efficiency:** Streamlined video workflows reduce operational overhead for large-scale events.
- **Reduced Costs:** Decentralized options (e.g., P2P or distributed storage) lower reliance on centralized CDN services.
- **Increased Adoption Metrics:** By merging video infrastructure with optional blockchain-based features (e.g., NFTs), we encourage mainstream use of decentralized technologies.

Qualitative Impact

Funded by the European Union

- **User-Centric Innovation:** Flexible streaming models, robust moderation tools, and straightforward event monetization.
- **Enhanced Interoperability:** Supports a range of third-party identity and storage modules, aligning with the broader decentralized ecosystem.
- **Solving Scalability & Privacy Challenges:** Demonstrates a real-world approach to hosting large events securely, with or without advanced cryptographic identity layers.

Alignment with Trends

- **Decentralized Media Hosting:** Follows the rise of distributed content delivery, ensuring resilience and reducing single points of failure.
- **Tokenized Content & Access:** Integrates seamlessly with NFT gating and user-owned digital assets for next-gen event monetization.

Funded by the European Union

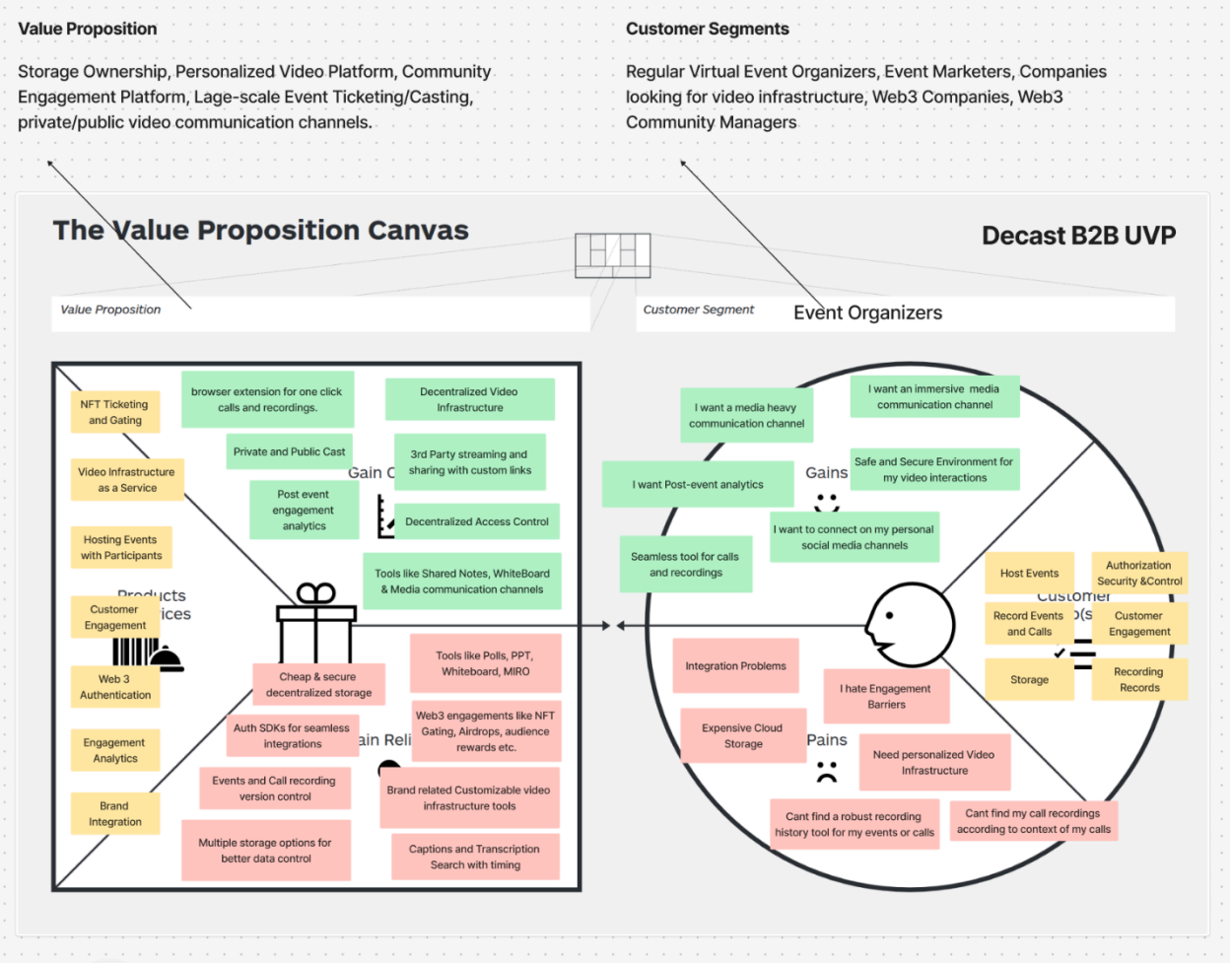

Figure 18 UVP of Decast

7.3 BUSINESS VALUE AND RELEVANCE FOR TRUSTCHAIN

Mutual Benefits

- **Leveraging TrustChain Resources:** Our open infrastructure can incorporate TrustChain's frameworks for identity, storage, and security, enhancing reliability.
- **Adding Value to the Ecosystem:** Decast.live offers a reusable, modular approach to decentralized video hosting, benefiting other TrustChain projects seeking similar functionalities.

Funded by the European Union

Value Exchange & Ecosystem Fit

- **Interoperable Features:** The platform's APIs and plugin architecture make it easy for TrustChain projects to integrate or build upon our streaming capabilities.
- **Shared Goals:** Decast.live and TrustChain both emphasize transparency, user empowerment, and decentralized governance, driving adoption of open technologies.

7.4 ANY OTHER IMPACT

Technological Impact

- **Next-Gen Video Infrastructure:** Showcases how decentralized hosting and streaming can handle high user concurrency without compromising performance.
- **Toolkits & Methodologies:** Encourages open-source contributions to improve encoding, distribution, and monetization modules.

Socio-Economic Impact

- **Job Creation:** Growing need for developers, content creators, and community managers around decentralized events and media.
- **Educational Accessibility:** Schools, universities, and training programs benefit from robust, cost-effective streaming solutions.

Environmental Impact

- **Efficient Resource Usage:** Containerized deployments and load balancing reduce idle resource consumption.
- **Reduced Reliance on Centralized CDNs:** Decentralized or edge-based streaming can minimize data transit and energy usage.

7.5 SAMPLE

Profiles

- **Educators & Trainers:** Need reliable, user-friendly live sessions with replay options.

Funded by the European Union

- **Event Organizers & Broadcasters:** Require stable, scalable streaming for conferences, product launches, and community events.
- **Corporate Clients:** Look for secure, branded solutions that integrate with existing workflows.
- **Blockchain Enthusiasts & Developers:** Interested in optional NFT gating and decentralized storage features.
- **General Users:** Assess everyday usability, performance, and potential friction points.

Sample Size & Criteria

- **100–200 Participants:** Offers a broad cross-section—technical and non-technical—to refine the platform's usability and feature set.

- **Selection Criteria:**
  - **Demographics:** Range of age groups and geographic locations.
  - **Professional Background:** Education, corporate, entertainment, and blockchain.
  - **Usage Context:** Frequency of streaming, required concurrency, interest in monetization or open-source collaboration.

Insights Utilization

- **Feedback Loop:** Iterative improvements to user flows, cost structures, and feature priorities.
- **Validating the Value Proposition:** Confirms that decentralized video hosting can address real-world pain points (e.g., scalability, user control, monetization).

8. CONCLUSIONS

This project tackles the challenges of decentralized identity management and secure video casting by eliminating centralized vulnerabilities and enhancing privacy and user autonomy. As online interactions surge, risks like data breaches and identity theft demand innovative solutions. Our initiative leverages blockchain, decentralized identifiers (DIDs), and zero-knowledge proofs to provide robust identity verification, fine-grained access control, secure streaming, NFT gating for exclusive content, and customizable branding.

Funded by the European Union

Our design is guided by extensive user feedback from educators, legal professionals, event organizers, and IT experts, ensuring an intuitive interface, adaptive streaming, and secure storage that blends decentralized networks with traditional cloud services. Technically, we have established a decentralized architecture that uses blockchain for tamper-proof data and encryption for privacy, with modular components for scalability. The system also integrates oracles to connect legacy systems and advanced cryptographic methods to guarantee interoperability and robust security.

In summary, this pioneering effort redefines identity management and video casting in a decentralized context, setting a new standard for secure, transparent, and efficient digital interactions. Decast is poised to empower users and organizations with a seamless, privacy-preserving, and highly adaptable digital solution for the modern era.

Funded by the European Union

REFERENCES

[1] https://www.passportjs.org
[2] Sporny, M., Longley, D., Sabadello, M., Reed, D., & Steele, O. (2022). Decentralized Identifiers (DIDs) v1.0. Retrieved from https://www.w3.org/TR/did-core/
[3] W3C. (2022). Credentials Verification Data Model. Retrieved from https://www.w3.org/TR/vc-data-model/
[4] Xu, J., et al. Anonymous Credential Schemes.
[5] Garman, C., et al. Anonymous Credential Schemes.
[6] Coconut Project. Decentralized Credential Issuance Framework.
[7] PrivIdEx Project. Privacy-Preserving Identity Exchange.
[8] CanDID Framework. A Blockchain-Based Framework for Decentralized Identifiers.
[9] U-Port. Self-Sovereign Identity Management.
[10] Serto. Decentralized Identity Solutions.
[11] Hyperledger Indy. Distributed Identity.
[12] Microsoft. Decentralized Identifier Solutions.
[13] https://www.passportjs.org/packages/passport-jwt/

Funded by the European Union

APPENDIX A.

Anything that is related but not core to the deliverable can go into appendix.

Funded by the European Union