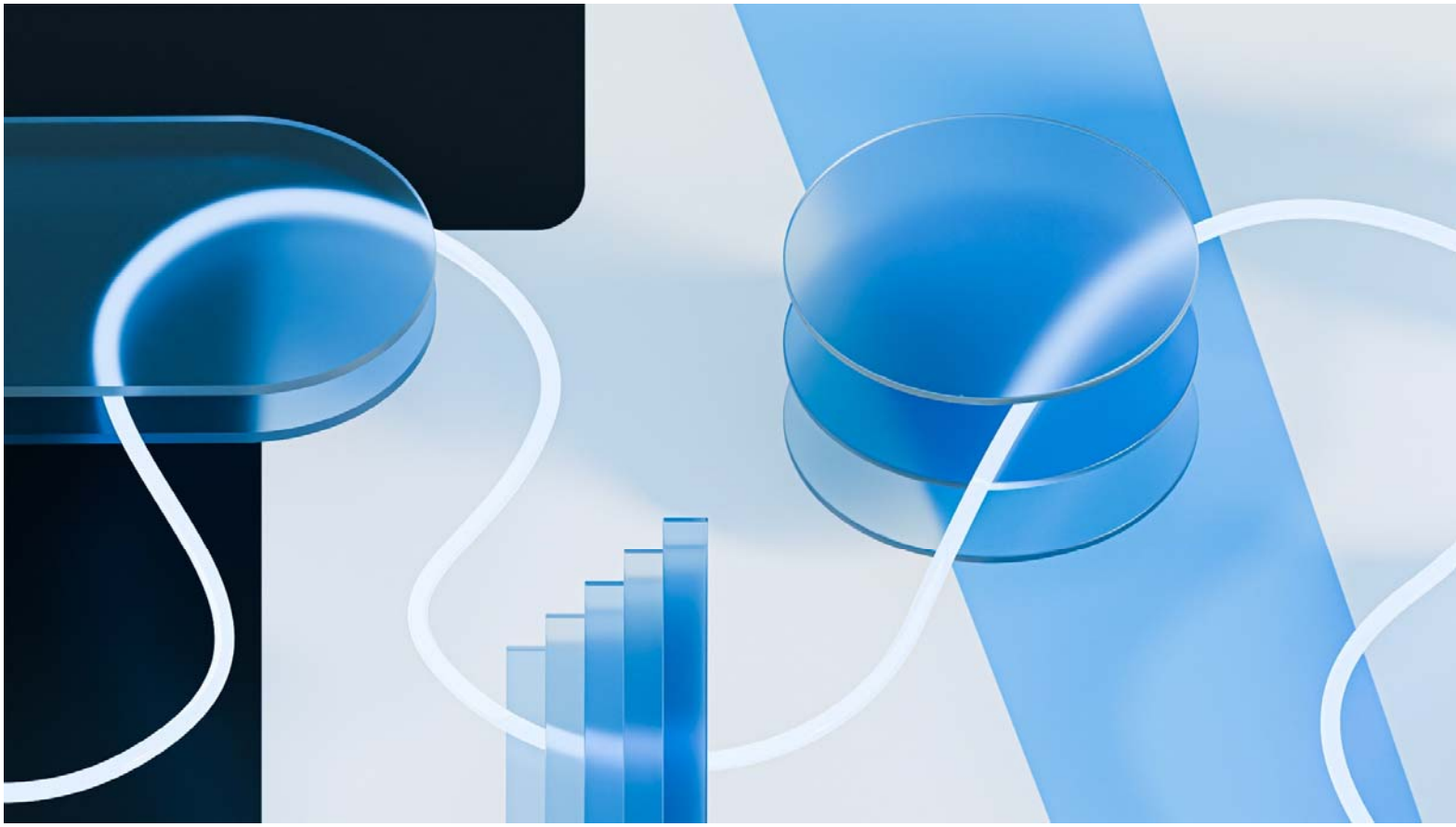
twon

twin of online social networks



TWin of Online Social Networks

Deliverable D4.3

TWON Software Platform

Main Authors: Alenka Guček, Abdul Sittar

Funded by the European Union

twon
twin of online social networks

About TWON

TWON (project number 101095095) is a research project, fully funded by the European Union, under the Horizon Europe framework (HORIZON-CL2-2022-DEMOCRACY-01, topic 07). TWON started on 1 April 2023 and will run until 31 March 2026. The project is coordinated by the Universiteit van Amsterdam (the Netherlands) and implemented together with partners from Universität Trier (Germany), Institut Jozef Stefan (Slovenia), FZI Forschungszentrum Informatik (Germany), Karlsruher Institut für Technologie (Germany), Robert Koch Institute (Germany), Univerzitet u Begogradu - Institut za Filozofiju I Drustvenu (Serbia), Slovenska Tiskovna Agencija (Slovenia) and Dialogue Perspectives e.V (Germany).

Funded by the European Union. Views and opinions expressed are those of the author(s) only and do not necessarily reflect those of the European Union. Neither the European Union nor the granting authority can be held responsible for them.

Funded by the European Union

Project Name	Twin of Online Social Networks
Project Acronym	TWON
Project Number	101095095
Deliverable Number	D4.3
Deliverable Name	TWON Software Platform
Due Date	30.09.2025
Submission Date	30.09.2025
Type	Other
Dissemination Level	PU - Public
Work Package	WP 4
Lead beneficiary	6-JSI
Contributing beneficiaries and associated partners	Institut Jozef Stefan (JSI), Universiteit van Amsterdam (UvA), Universität Trier (UT)

Executive Summary

To investigate the impact of online social networks (OSNs) on democratic debates, the project has developed a suite of software platforms that enable experimentation, analysis, and demonstration of simulated digital replicas of OSNs, referred to as TWONs. These platforms are designed to support different dimensions of the research, ranging from empirical case studies to public engagement. In total, five platforms have been implemented, each with a distinct focus:

1. A use case–oriented platform validated through interaction with actual users;
2. Large Scale Simulation (LSS) platform optimized for straightforward configuration;
3. LSS platform with UI for human interaction with agents;
4. TWON Base Simulation platform incorporating plug-in agents to enable flexible experimentation;
5. A demonstrator platform designed to communicate the project's results to a general audience.

The deliverable provides an overview of the design, scope, and functionality of each platform and discusses their role within the broader research and dissemination activities of the project. Collectively, these platforms constitute a versatile software infrastructure that will underpin the project's empirical investigations, methodological developments, and outreach efforts in subsequent phases.

twon
twin of online social networks

Contents

List of Figures

List of Abbreviations

DS Data Source

LLM Large Language Model

LSS Large Scale Simulation

OSN Online Social Network

SFT Supervised Fine Tuning

TWON Twin of an Online Social Network

1 Introduction

Online social networks (OSNs) have become central arenas for public discourse and democratic engagement, yet their scale and complexity make them difficult to study under controlled conditions. To address this challenge, the project has developed a set of experimental software platforms—collectively referred to as TWONs—which serve as simulated digital replicas of OSNs. These platforms provide controlled environments for investigating information flows, user behaviour, and the dynamics of public debate, while also supporting methodological development and public outreach.

This deliverable documents the design, scope, and functionality of the five TWON software platforms developed to date. It focuses on their technical architectures, implementation choices, and intended roles within the project's broader research and dissemination activities. The platforms are:

- A use case–oriented platform, validated through interaction with real users to support applied empirical studies;
- An LSS platform optimised for straightforward configuration, enabling rapid experimental setup and execution;
- An LSS platform with a user interface for human–agent interaction, supporting mixed human–simulation experiments;
- The TWON Base Simulation platform, a modular framework with plug-in agents to enable flexible experimentation;
- A demonstrator platform, designed to communicate project results to a general audience in an accessible way.

This deliverable builds on top of deliverables that were submitted from WP2, WP3 and WP4, and is preceding D4.4, which will provide Computation report on the ran simulations.

2 A use case–oriented platform validated through interaction with actual users

We have developed a web application for human participant studies, designed with scalability, privacy, and secure data handling. The platform is hosted separately from the database and runs independently. It enables multiple participants to join simultaneously through unique links generated per study. This ensures that each participant is tracked individually and prevents duplicate entries, maintaining the integrity of the research data. We presented the platform for simulating debates on social media using digital twin

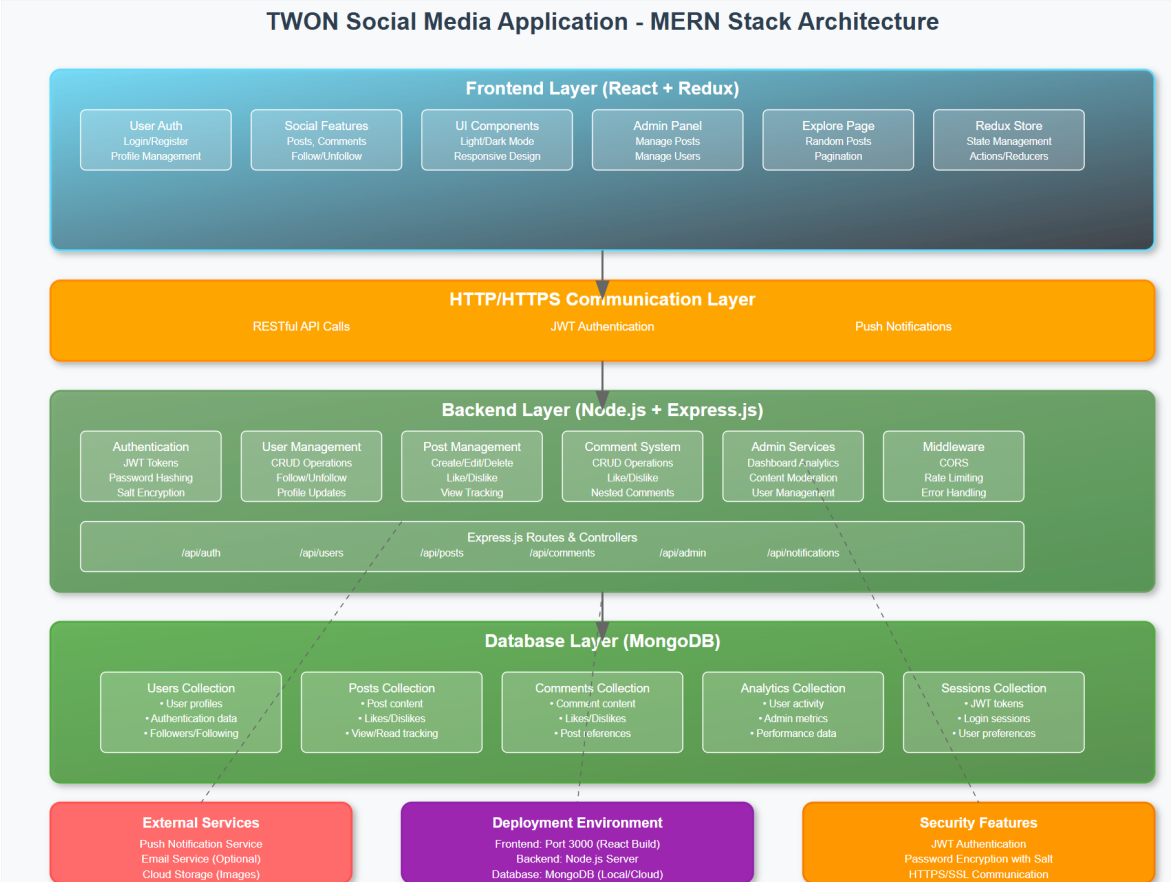


Figure 1: How TWON Social Media Works: A visual breakdown of the application's architecture from user interface to database storage

methodology at the European Data Science Day, part of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (Sittar et al. (2024)).

2.1 Architecture

The TWON social media application follows a classic MERN stack architecture. As shown in Figure 1, the platform's architecture is composed of several interconnected layers and components, which are described in detail in the following sections. User interacts with the Frontend layer, which is connected with the Communication Layer. Backend layer is built with Node.js and communicates both with Communication Layer and the Database Layer (MongoDB) for storing the captured data. The platform is open source and is available at: `https://github.com/abdulsittar/TWON_STA_Case_Study`.

2.2 Implementation details

Frontend Layer - User Experience and Interface

The frontend layer is built using React.js with Redux for state management, creating a dynamic and re-

sponsive user interface to handle all user interactions. The application includes several key components: user authentication modules for registration and login, social interaction features for creating and managing posts and comments, and a sophisticated UI system that supports both light and dark themes. The frontend also incorporates an admin panel that provides administrative users with comprehensive control over the platform, including the ability to manage posts, users, and comments. Additionally, an explore page allows users to discover content from other users through a paginated interface, while profile pages display user information, posts, and social connections like followers and following lists. The Redux store centralizes state management, ensuring consistent data flow throughout the application and maintaining user preferences and session information. See Figure 2 for example posts.

Communication Layer - API and Data Transfer

Between the frontend and backend lies the HTTP communication layer, which facilitates all data exchange through RESTful API calls. This layer implements JWT (JSON Web Token) authentication to ensure secure communication between the client and server. The system supports real-time features through push notifications, which require public and private keys configured in the environment variables. All communication is designed to be stateless and follows REST principles, making the API predictable and easy to maintain. The communication layer also handles error responses, data validation, and ensures that sensitive information is properly encrypted during transmission.

Backend Layer - Business Logic and Processing

The backend layer, built with Node.js and Express.js, serves as the application's core processing engine. This layer contains multiple specialized services that handle different aspects of the application's functionality. The authentication service manages user registration, login, and JWT token generation, while implementing secure password hashing with salt encryption to protect user credentials. User management services handle CRUD operations for user profiles, following and unfollowing functionality, and profile updates. The post management system allows users to create, edit, and delete posts while tracking views and reads, and managing likes and dislikes. A comprehensive comment system supports nested comments with their own like and dislike functionality. The admin services provide dashboard analytics showing total counts of posts, users, and comments, along with content moderation capabilities. Various middle-ware components handle cross-origin requests (CORS), rate limiting to prevent abuse, and comprehensive error handling to ensure application stability.

Database Layer - Data Persistence and Management

MongoDB serves as the application's primary database, organizing data into several specialized collections. The Users collection stores user profiles, authentication credentials, and social connection data including followers and following relationships. The Posts collection maintains all user-generated content, associated metadata like creation dates and author information, engagement metrics such as likes and dislikes,

and tracking data for views and reads. The Comments collection stores comment content with references to their parent posts, engagement metrics, and supports the hierarchical structure needed for nested comments. An Analytics collection captures user activity patterns, performance metrics, and data required for the admin dashboard's statistical displays. Finally, a Sessions collection manages JWT tokens, active login sessions, and user preferences like theme selection. This database structure supports efficient querying and maintains referential integrity across related data.

Security and Deployment Considerations

The application implements robust security measures throughout all layers. JWT authentication provides stateless session management, while password encryption with salt ensures that user credentials remain secure even if the database is compromised. The system is designed to communicate over HTTPS to prevent man-in-the-middle attacks and data interception. For deployment, the frontend is built and served from port 3000, while the backend runs on a separate Node.js server instance. The MongoDB database can be hosted locally during development or deployed to cloud services for production use. The application also integrates with external services for enhanced functionality, including push notification services for real-time user engagement and potentially cloud storage services for handling media uploads and static assets. This comprehensive architecture ensures that the TWON social media application can scale effectively while maintaining security and performance standards expected of modern web applications.

2.3 User interaction

This platform has already been successfully used in two case studies, for which details will be found in deliverable 5.3 (M36). Users are first presented with a sign up page and pre survey questions. Upon completing those, they access the platform (see Figures 2 and 3 for example). In the platform, interactions are carefully monitored and managed through the frontend. Users can like, dislike, or comment on posts, with comments requiring navigation to the post's detail view. The platform enforces a minimum interaction threshold: participants must perform a predefined number of actions before proceeding to the post-survey, which collects study-specific responses. Once the threshold is met, a pop-up guides the user to the post-survey, and completing this survey marks the study as finished. An info button is available at all times to provide context or guidance.

3 LSS platform optimized for straightforward configuration

LSS is a Node.js-based research platform designed for conducting large-scale social media simulations using Large Language Model (LLM) agents. This platform enables researchers and developers to study agent behaviours, interaction patterns, and emergent phenomena in controlled social media environments. Im-

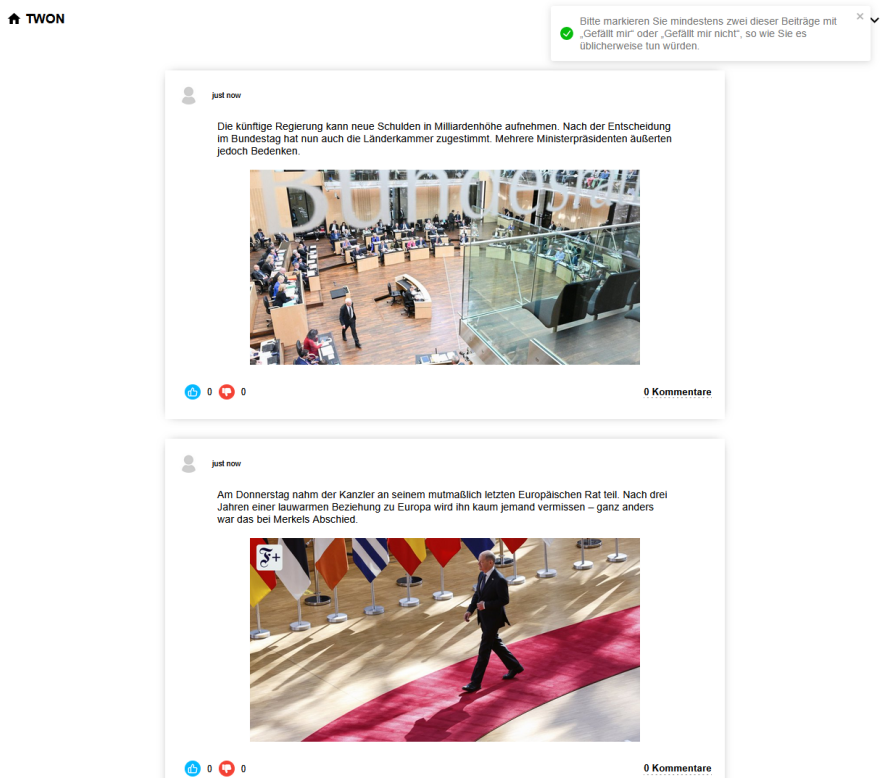


Figure 2: Screenshot example of the platform used in the RKI study

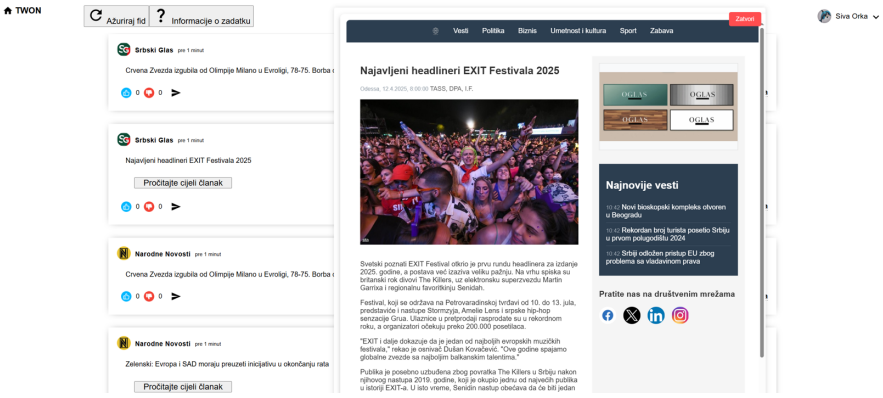


Figure 3: Screenshot example of the platform used in the STA study

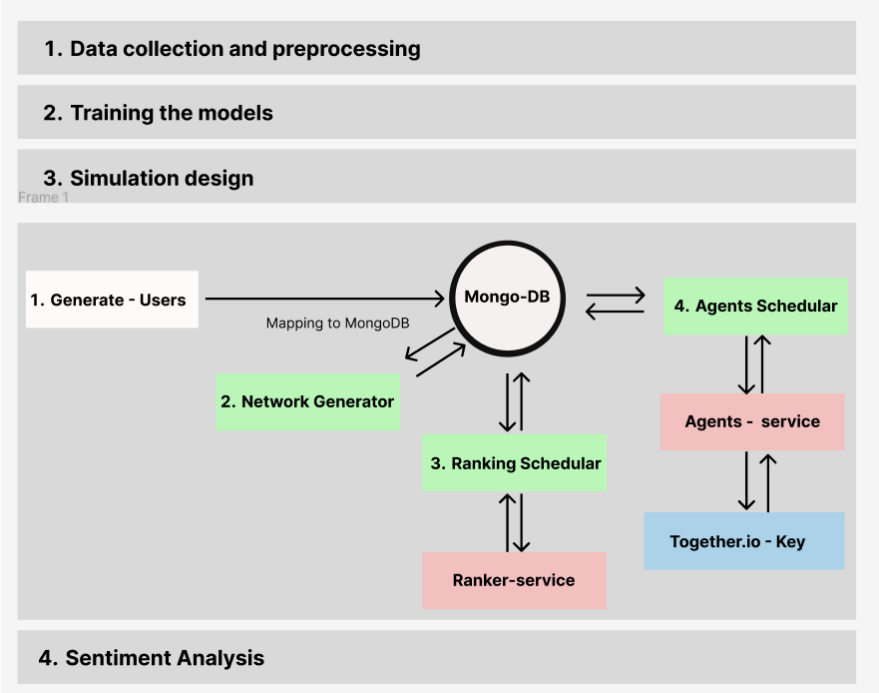


Figure 4: LSS platform schema with architecture

plementation approach is visualized in Figure 5. This platform is open source and is available at: `https://github.com/abdulsittar/LSS`.

3.1 Technical Architecture and Components

The LSS platform aims to create realistic social media simulations. The simulation uses the Barabási–Albert (BA) model, which grows the network over time and connects newcomers preferentially to already well-connected nodes. This creates realistic social network structures with hub nodes (highly connected users) and peripheral nodes (less connected users), yielding a power-law–like degree pattern where most have few links and a few have many. It's a strong baseline for social-media-style systems, though real platforms can diverge due to directionality, clustering, recommendation algorithms, and user heterogeneity. The platform operates on several key technical layers. At its core, it uses LLM agents that can generate posts and comments based on configurable topics and models. The system implements a sophisticated behavioural model where each agent has dynamic attributes including time budgets and motivation levels. The time budget formula is calculated as: Ti = 100 - 20 * (number of posts) - 20 * (number of comments) - 5 * (number of likes + dislikes), representing how agent activity decreases their available time. Similarly, motivation is tracked using: Mj = initial motivation + 5 * (likes received) - 5 * (dislikes received), creating realistic feedback loops where positive engagement increases agent motivation while negative engagement decreases it.

Key Features and Capabilities The LSS platforms functions include agent-based social media interactions, network visualization, sentiment analysis, and data export functionality. The system can run social

media simulations with LLM agents, generate network (Barabasi), visualize agent interactions and social graphs, perform sentiment analysis of posts and comments from top users, and export data and visual insights. The integration with Together AI or other LLM APIs (from WP3 or others) enables agents to create contextually appropriate posts and responses.

3.1.1 Architecture based on probabilistic model

It employs a two-stage hierarchical approach combining probabilistic scheduling with domain-specialized fine-tuned language model agents to simulate realistic social media interactions. The framework consists of two primary components: (1) a **Timeline-Based Probabilistic Model** that serves as an environmental scheduler, and (2) **Domain-Specialized Fine-tuned Agents** that generate contextually appropriate content based on the scheduler's decisions.

The probabilistic scheduler is implemented as a multi-output neural network that simultaneously predicts four key dimensions of social media behaviour: agent selection (which agent should act next), action classification (post vs. reply), temporal prediction (timing of next action), and context setting (emotional tone and topical focus for content generation). The model is trained on 88,330 conversation items spanning April 2019 to April 2020, focusing on AI and cryptocurrency discussions. Our Timeline-Based approach generates 93,440 chronological training pairs—18.7× more than baseline methods—through complete conversation sequence learning rather than isolated post-reply pairs. Given the current state $S(t)$ at time t, the model computes probability distributions over the action space. The timeline manager determines which agent should act next based on the current time, agent, context, and action. The selected fine-tuned model then generates a new post or reply for the chosen agent, creating realistic conversation flow (see Figure 5).

We implement a single fine-tuned language model that serves as both AI and cryptocurrency agents. The model is trained on conversations from both domains (AI technology and cryptocurrency discussions) to capture the vocabulary, argumentation patterns, and discourse styles across both topic areas.

- **Agent A (AI Focus)**: The same fine-tuned model called when the probabilistic scheduler determines AI-related content is needed.
- **Agent B (Crypto Focus)**: The identical fine-tuned model called when cryptocurrency-related content generation is required.

When called by the probabilistic scheduler, the fine-tuned model generates content based on provided context including action type (post/reply), emotional context, topical focus, temporal context, and conversation history. The model's training on both domains enables it to produce contextually appropriate responses regardless of which agent role it is fulfilling. We presented our work on designing AI agents for social media at the 28th International Multiconference Information Society SiKDD, highlighting methods

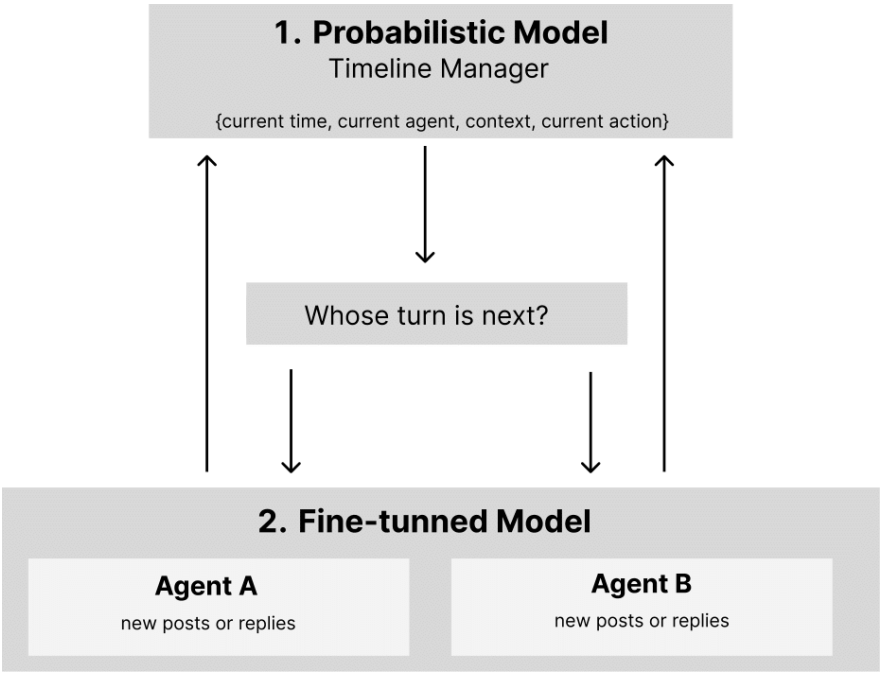


Figure 5: Overview of the proposed methodology for conversation simulation

for intelligent interaction and content moderation Sittar et al. (2025b).

4 LSS platform with UI for human interaction with agents

4.1 Architecture

We next developed a TWON platform to simulate social media debates by integrating human interaction within a controlled network framework. TWON simulations is a typescript-based research platform designed for conducting large-scale multi-agent social media simulations using LLM agents. This platform enables researchers and developers to study agent behaviours, interaction patterns, and emergent phenomena in controlled social media environments, e.g. explore how agents perceive information, adjust their emotions and stances, and provide insights into social media dynamics. The platform is open source and available at: `https://github.com/abdulsittar/TWON_Simulations`. Agent-based simulations have been effectively used to model online political discussions, providing insights into voter behaviour during elections in Germany, as presented at the ESWC 2025 Workshops and Tutorials (Sittar et al. (2025a)).

4.2 Technical Architecture and Design

TWON simulations follows a modern full-stack architecture with clear separation between frontend and backend components. The system is built using typescript throughout, ensuring type safety and better code maintainability. The platform implements a multi-agent framework where LLM agents within a sim-

Figure 6: Configurations to setup a simulation

ulated social network specifically look at how information is spread, utilizing generative agents to control behaviour patterns. The backend serves as the simulation engine, built on Node.js with typescript, handling all computational aspects including agent life cycle management, LLM integration, data persistence, and real-time analytics. It uses MongoDB for data storage and implements RESTful APIs for communication with the frontend. The frontend provides a modern React-based user interface with tailwind CSS styling, featuring dark mode support and real-time visualization of simulation metrics.

5 TWON Base Simulation platform with plug-in agents for flexible experimentation

TWON Base Simulation platform is a modular framework for experimenting with how conversations and content spread on social platforms at scale. The platform is configured in a way one can mix and match building blocks—agents, ranking logic, and simulation drivers—and choose between built-in models like a Bounded Confidence Model or an LLM-powered "TWON-base." It supports different platform styles (e.g., X/Twitter-like or Reddit-like feeds), lets agents read, react, and post, and then saves results in structured files for analysis. The design is lightweight and API-driven, so teams can plug in their own components or connect external services without committing to a specific deployment setup. The platform was developed by University of Trier and is open-source (Apache 2.0 license), the code is available at: `https://github.com/cl-trier/TWON-LSS`.

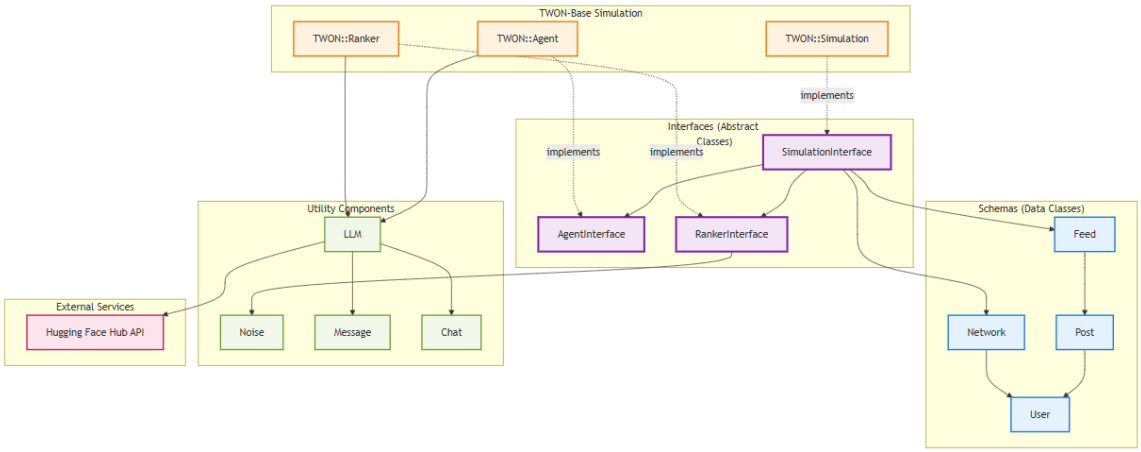


Figure 7: Schematic workflow of the TWON Base Simulation platform

5.1 Architecture

The platform is organised into four cohesive areas. A set of interfaces defines the plug-in points for agent logic, content ranking, and simulation orchestration. A collection of schemas captures the core data models—users, posts, per-user feeds, and the social graph. Reference simulations include both the bounded-confidence setup and the LLM-driven TWON-Base. A utility layer provides common services such as language-model integration and noise injection. See Figure 7 for details.

At runtime, a simulation composes a ranker that scores and orders content, a population of agents that read, react, and generate content, and a network-and-feed substrate that handles message routing, timelines, and notification-like mechanics. The design is deliberately lightweight and API-driven so teams can introduce custom components or connect external services without committing to a specific deployment model. Scalability is achieved through modular components and step-wise execution with progress tracking, enabling large-scale experiments.

5.2 Implementation details

The implementation is in Python. The platform uses a network-analysis library to model social graphs and support neighbour discovery. Simulations produce JSON artefacts—network state, feeds, and individual trajectories—and expose provider-agnostic hooks for LLM integration. It requires no stateful database, is packaged as a Python library (interfaces, schemas, utilities, simulation modules) without a frontend, and integrates with other tools via exported data.

5.3 User interaction

The platform is API-first. Users compose experiments by selecting a simulation (for example, TWON-Base), choosing or implementing an agent and a ranker that satisfy the corresponding interfaces, and configuring

network size, interaction limits, and step count. Execution proceeds in discrete steps; upon completion, the platform emits JSON files that capture the network, feed histories, and final agent states, which can be analysed in external tools. See Figure 8.

6 TWONy demonstrator for general audience

microTWONy is a browser-based micro-simulation that showcases how generative agents and recommendation policies shape the emotional tone of social-media feeds. Users seed the system with a post, select a language model, and choose between a chronological baseline or an emotion-prioritising ranker; autonomous personas then populate the feed while the interface tracks positive and negative valence at feed- and user-level over time. Visual cues on each post surface emotion categories, and compact dashboards display trajectories of emotionality and emerging polarisation. Ranking parameters (including the weighting of positive versus negative affect) and agent prompts/personas are directly editable, enabling rapid what-if experiments without real-user data. In sum, microTWONy provides a concise, configurable demonstration of TWON's core ideas—recommendation effects, emotional contagion, and discourse dynamics—suitable for inclusion alongside the main simulation platform. The platform was developed by University of Trier and is available at: https://simon-muenker.github.io/TWONy-micro/. See deliverables from WP3 for more details on TWONy.

7 Conclusions

In conclusion, this deliverable documents how the project has progressed from the idea of building a single, general-purpose simulation platform to a coherent suite of purpose-built TWON platforms that address distinct research and outreach needs. Together, the five implemented platforms provide an end-to-end capability: from configuration-friendly large-scale simulation and human-in-the-loop interaction, through flexible, plug-in experimentation in TWON Base, to concise demonstrations for public communication. In parallel to the demonstrator presented here, additional public-facing demonstrators (e.g. macroTWONy, ethicsTWONy) are being developed to broaden engagement and transparency around OSN dynamics and recommendation effects (https://www.twon-project.eu/twony/). This diverse tools support empirical case studies, methodological advances, and stakeholder dialogue with tools, each optimized for their specific tasks. Concretely, project partners are directly benefiting from the platforms: UvA has used the Use case-oriented platform for two case studies; JSI (with KIT and UT) has used LSS platforms for straightforward configuration and human interaction to run the midscale simulation; UT and KIT are running the LSS with the TWON Base Simulation platform, and the project dissemination (FZI, DIA) heavily relies on the demonstrators, such as microTWONy.

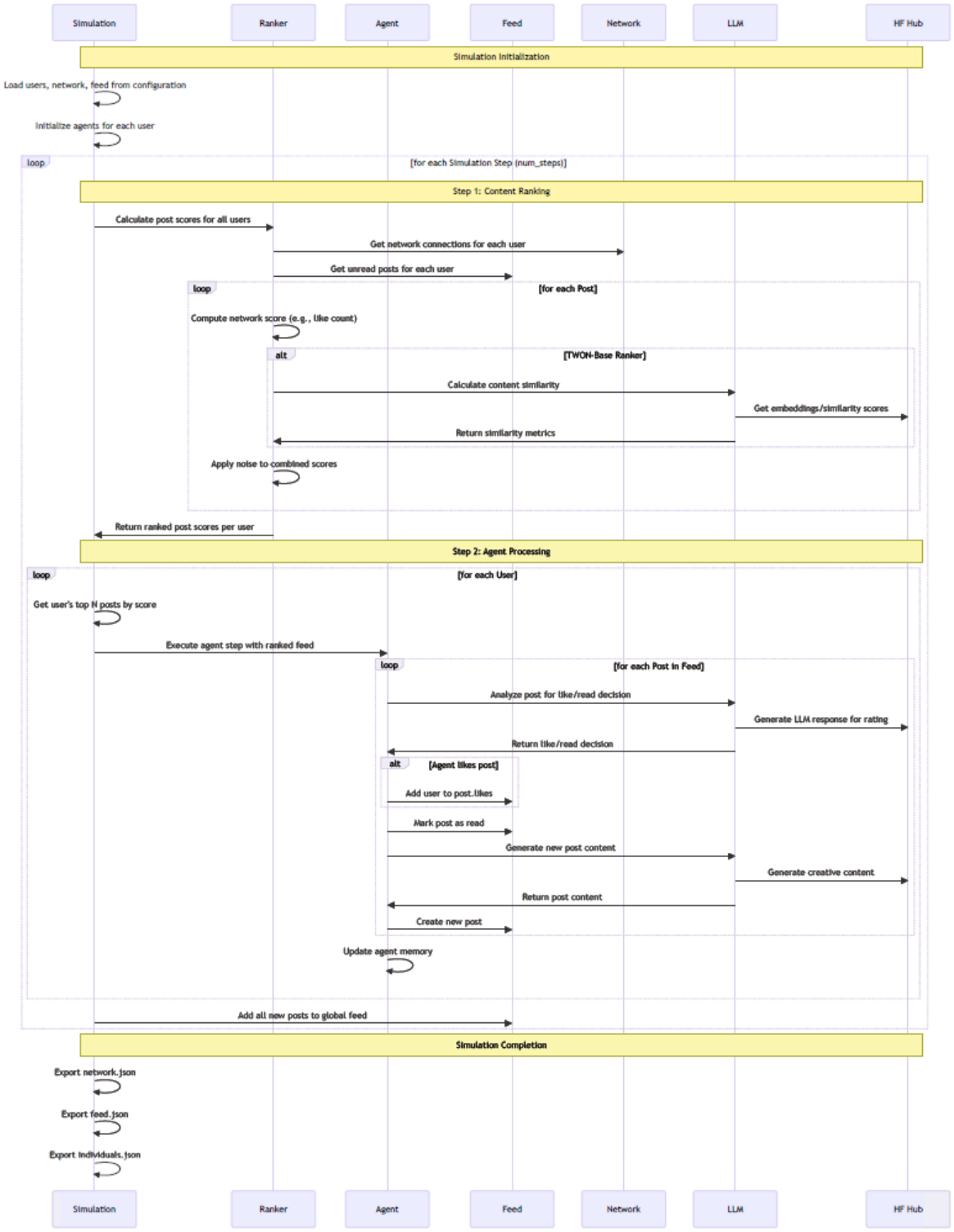


Figure 8: Step-by-step simulation workflow with the TWON Base Simulation platform

Looking forward, we are planning to connect the TWON social media platform with the large-scale simulation (LSS) platform to enable hybrid studies that incorporate both real human participants and AI agents. This integration will allow us to conduct richer case studies and larger-scale simulations, combining empirical human behaviour with controlled agent-based interactions. By merging the strengths of the TWON platform's human-in-the-loop capabilities with the scalability and flexibility of the LSS platform, we aim to investigate complex social dynamics, information propagation, and emergent behaviours in social media ecosystems at unprecedented scale. This platform is open source and is available at: `https://github.com/abdulsittar/hybrid-social-platform`.

In the last phase of the project we will run the LSS platforms: large, at orders of magnitude greater scale, simulation campaigns driven by different network dynamics, schedulers, and domain-specialised (fine-tuned) LLM agents. We will vary network size and seeding, agent time-budget/motivation parameters, topics, and ranking services to answer concrete research questions about information flow, engagement dynamics, and the effect of alternative ranking choices. These efforts will enable systematic exploration of democratic-debate dynamics under varied recommendation regimes, provide evidence for policy-relevant assessments, and feed lessons learned back into the platforms' design, ultimately maturing the TWON ecosystem into a scalable, reliable infrastructure for research and public engagement.

References

Abdul Sittar, Simon Münker, Alenka Guček, and Marko Grobelnik. Simulating debates on social media with digital twin – twon. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2024. European Data Science Day: KDD-2024 Special Day. 6

Abdul Sittar, Alenka Guček, and Marko Grobelnik. Agent-based simulations of online political discussions: A case study on elections in germany. In *Joint Proceedings of the ESWC 2025 Workshops and Tutorials (SemGenAge)*, volume 3977 of *CEUR Workshop Proceedings*, Portorož, Slovenia, 2025a. URL `https://ceur-ws.org/Vol-3977/SemGenAge-6.pdf`. 12

Abdul Sittar, Mateja Smiljanić, and Alenka Guček. Designing ai agents for social media. In *Proceedings of the 28th International Multiconference Information Society SiKDD*, volume 100, 2025b. 12

twon

twin of online social networks

Contact us

Damian Trilling
Project Coordinator

+31 62 782 7904

d.c.trilling@uva.nl

University of Amsterdam
Postbus 15791
1001 NG Amsterdam

Funded by
the European Union