# **User Requirements Specification for YansTribe**

By Petar Yankov

Contents

[**User Requirements Specification for YansTribe** 1](#_Toc198860030)

[1. Introduction 3](#_Toc198860031)

[2. Functional Requirements (FR) 3](#_Toc198860032)

[FR-01: User Registration & Authentication ✅ 3](#_Toc198860033)

[FR-02: User Profile Management 3](#_Toc198860034)

[FR-03: User Profile Viewing ✅ 3](#_Toc198860035)

[FR-04: Post Creation ✅ 3](#_Toc198860036)

[FR-05: Post Viewing in profile ✅ 3](#_Toc198860037)

[FR-06: Post Viewing in home page ✅ 4](#_Toc198860038)

[FR-07: Post Hiding ✅ 4](#_Toc198860039)

[FR-08: Post Editing 4](#_Toc198860040)

[FR-09: Post Deletion 4](#_Toc198860041)

[FR-10: Post Reactions ✅ 4](#_Toc198860042)

[FR-11: Comment Section 4](#_Toc198860043)

[FR-12: Comment Creation 4](#_Toc198860044)

[FR-13: Comment Editing 4](#_Toc198860045)

[FR-14: Comment Deletion 4](#_Toc198860046)

[FR-15: Comment Reactions 5](#_Toc198860047)

[FR-16: Post Stories 5](#_Toc198860048)

[FR-17: View Stories 5](#_Toc198860049)

[FR-18: Friend Management ✅ 5](#_Toc198860050)

[FR-19: Messaging System 5](#_Toc198860051)

[FR-20: Privacy & Security 5](#_Toc198860052)

[FR-21: Admin Controls 5](#_Toc198860053)

[3. Non-Functional Requirements (NFRs) 5](#_Toc198860054)

[NFR-01: Performance 5](#_Toc198860055)

[NFR-02: Security 6](#_Toc198860056)

[NFR-03: Scalability 6](#_Toc198860057)

[NFR-04: Usability 6](#_Toc198860058)

[NFR-05: Reliability 6](#_Toc198860059)

[NFR-06: Maintainability 6](#_Toc198860060)

[NFR-07: Compatibility 6](#_Toc198860061)

[NFR-08: Compliance 6](#_Toc198860062)

[4. Use Cases (UC) 7](#_Toc198860063)

[UC-001: User Registration 7](#_Toc198860064)

[UC-002: Create Post 8](#_Toc198860065)

[UC-003: Send Friend Request 8](#_Toc198860066)

[UC-004: Send Message 9](#_Toc198860067)

[UC-005: Delete Comment (Admin) 9](#_Toc198860068)

[UC-006: Edit Post 10](#_Toc198860069)

[UC-007: React to Comment 10](#_Toc198860070)

[UC-008: View Stories 11](#_Toc198860071)

[UC-009: View Post 11](#_Toc198860072)

[UC-010: View Message 12](#_Toc198860073)

[UC-011: View Chat Interface 12](#_Toc198860074)

[UC-012: View Profile Page 12](#_Toc198860075)

[UC-013: View Settings Page 13](#_Toc198860076)

[5. Conclusion 13](#_Toc198860077)

## Introduction

This web application is a social media platform that integrates features from booth Facebook and Instagram, allowing users to interact through stories, posts, comments, messages, and friendships. Users can personalize their experience by modifying their settings, privacy options, and profile details.

## Functional Requirements (FR)

### FR-01: User Registration & Authentication ✅

* Users must be able to sign up, log in, and log out securely.
* Implement password encryption and account verification.
* Provide a "Forgot Password" feature to allow users to reset their passwords securely.

### FR-02: User Profile Management

* Users can view and edit personal details, including name, bio, profile picture, and password.
* Users can set their account visibility as public or private.

### FR-03: User Profile Viewing ✅

* Allow users to view other users' profiles, respecting privacy settings.
* View all the stories and posts for the current public or friend user profile

### FR-04: Post Creation ✅

* Users can create posts containing text, images, or videos.
* Users must be able to preview the post before publishing.
* After previewing, users can edit or confirm the post.
* Once confirmed, the post is published based on the user's privacy settings.

### FR-05: Post Viewing in profile ✅

* Users can view posts from their own profile, friends, and public accounts.
* Posts should be displayed in chronological order or sorted by relevance (e.g., most recent, most liked).

### FR-06: Post Viewing in home page ✅

* Users can view related to them posts including popular posts, posts from friends and random posts

### FR-07: Post Hiding ✅

* Users can hide unwanted posts from their feed.

### FR-08: Post Editing

* Users can edit their own posts.
* The system must maintain a record of the last modified timestamp.
* Edited content should be updated without creating duplicates.

### FR-09: Post Deletion

* Users can delete their own posts.
* Deleted posts must be permanently removed.
* Admins can delete posts that violate community guidelines.

### FR-10: Post Reactions ✅

* Users can like or unlike any post they have access to.
* The system must prevent duplicate likes by the same user.

### FR-11: Comment Section

* Users must be able to see the comment section after a post.
* Users must be able to hide unwanted comments.

### FR-12: Comment Creation

* Users can add comments to any post they have access to.

### FR-13: Comment Editing

* Users can edit their own comments.
* An “Edit” option must be visible for the user's own comments within the modal.

### FR-14: Comment Deletion

* Users can delete their own comments.
* Admins can delete any comment that violates platform rules.
* A “Delete” option must be available for the comment owner (and admins).

### FR-15: Comment Reactions

* Users can like or unlike any comment they have access to.

### FR-16: Post Stories

* Users can upload stories that disappear after 24 hours.

### FR-17: View Stories

* Users can view stories from their friends and public accounts.

### FR-18: Friend Management ✅

* Users can send, accept, and decline friend requests.
* Users can remove friends from their list.

### FR-19: Messaging System

* Users can send private messages to their friends.
* Users can view their message history.
* Provide a chat interface for real-time communication.

### FR-20: Privacy & Security

* Users can block other users.
* Users can change their password.
* Implement role-based access control to restrict access to certain features based on user roles.

### FR-21: Admin Controls

* Admins can manage users, posts, and comments.
* Admins can delete accounts violating platform rules.
* Admins can view reports and analytics related to user activity.

## Non-Functional Requirements (NFRs)

### NFR-01: Performance

* The system must handle at least 10,000 concurrent users without performance degradation.
* Page load times should not exceed 2 seconds under normal operating conditions.

### NFR-02: Security

* Implement secure authentication and data encryption.
* All sensitive user data must be encrypted using industry-standard encryption algorithms.
* The system should undergo regular security assessments to address potential vulnerabilities.

### NFR-03: Scalability

* The application must be scalable to accommodate growth in user base and data volume.
* The system should support horizontal scaling to handle increased load.

### NFR-04: Usability

* The UI must be intuitive and responsive.
* The system should be accessible to users with disabilities, adhering to accessibility standards.

### NFR-05: Reliability

* The system should have an uptime of at least 99.9%.
* The system should be able to recover gracefully from failures without data loss.

### NFR-06: Maintainability

* The codebase should be modular and well-documented to facilitate maintenance and updates.
* The system should support automated testing to ensure stability during updates.

### NFR-07: Compatibility

* The application should be compatible with major browsers (Chrome, Firefox, Safari, Edge).
* The application should be responsive and function correctly on various devices (desktops, tablets, smartphones).

### NFR-08: Compliance

* The system must comply with relevant data protection regulations (e.g., GDPR).
* User data handling practices should be transparent and allow users to manage their data preferences.

## Use Cases (UC)

### UC-001: User Registration

**Actor**: Visitor  
**Trigger**: User initiates registration.  
**Preconditions:**

* User is not logged in.
* Valid email is available.

**Postconditions:**

* A verified account is created.

**Main Success Scenario (MSS):**

* 1. User accesses the registration page.
  2. System prompts for email, username, password.
  3. User submits the form.
  4. System validates input and checks uniqueness.
  5. System hashes the password.
  6. System sends a verification email.
  7. Confirmation message is shown to user.

**Extensions:**

* 3a. User submits incomplete form:
  1. System highlights missing fields. Return to step 2.
* 4a. Email or username already exists
  1. System shows error. Return to step 2.
* 6a. Email server down
  1. Notify delay. Account created. Return to homepage.

### UC-002: Create Post

**Actor**: Logged-in User  
**Trigger**: User clicks “Create Post.”  
**Preconditions**: User is authenticated.

**Postconditions:** Post is saved and displayed per privacy settings.

**MSS:**

1. System opens post creation modal.
2. User enters content.
3. User previews.
4. User confirms and submits.
5. System saves and displays post.

**Extensions:**

* 2a. Unsupported format uploaded
  1. Show error. Return to step 2.
* 4a. User cancels
  1. Discard content. Return to feed.

### UC-003: Send Friend Request

**Actor**: Logged-in User  
**Trigger**: User clicks “Send Friend Request.”  
**Preconditions**: Logged-in and target user exists.  
**Postconditions**: Friend request saved.

**MSS:**

1. User clicks “Send Friend Request.” Button.
2. System saves request and notifies recipient.

**Extensions:**

* 2a. Already requested
  1. Show message. End.
* 3a. User blocked
  1. Show error. End.

### UC-004: Send Message

**Actor**: Logged-in User  
**Trigger**: User selects friend and sends message.  
**Preconditions**: Users can message.  
**Postconditions**: Message delivered or queued.

**MSS:**

1. User opens chat.
2. User types message.
3. User clicks send button.
4. System delivers or queues.

**Extensions:**

* 3a. Message too long
  1. Show error. Return to step 2.

### UC-005: Delete Comment (Admin)

**Actor**: Administrator  
**Trigger**: Admin selects “Delete.”  
**Preconditions**: Admin logged in. Comment exists.  
**Postconditions**: Comment deleted.

**MSS:**

1. User can view comment list.
2. User selects comment.
3. User clicks delete button.
4. System removes the comment.

**Extensions:**

* 2a. Already deleted
  1. Show error. End.
* 3a. Cancel deletion
  1. No changes. Return to list.

### UC-006: Edit Post

**Actor:** Logged-in User  
**Trigger:** Clicks “Edit Post.”  
**Preconditions:** User owns the post.  
**Postconditions:** Post updated.

**MSS:**

1. System loads post content.
2. User edits the contents.
3. User clicks save.
4. System updates the content and the timestamp.

**Extensions:**

* 3a. Cancel edit
  1. Discard changes. Return.
* 4a. Save fails
  1. Show error. Retry option.

### UC-007: React to Comment

**Actor:** Logged-in User  
**Trigger:** Clicks “Like” on comment.  
**Preconditions:** Comment visible.  
**Postconditions:** Like state toggled.

**MSS:**

1. User clicks “Like.”
2. System checks existing like.
3. System toggles state.

**Extensions:**

* 3a. Database error
  1. Show error. Retry.

### UC-008: View Stories

**Actor**: Logged-in User  
**Trigger**: Clicks stories bar.  
**Preconditions**: Active stories exist.  
**Postconditions**: Story viewed.

**MSS**:

1. User taps profile in story bar.
2. System loads and plays story.
3. Story expires after 24h.

**Extensions:**

* 2a. Load fails
  1. Show error. Retry.
* 1a. No stories
  1. Show message. End.

### UC-009: View Post

**Actor**: Logged-in User  
**Trigger**: Scrolls or clicks post.  
**Preconditions**: Post visible.  
**Postconditions**: Post displayed.

**MSS**:

1. System loads feed/posts.
2. User clicks post.
3. System enables interaction.

**Extensions**:

* 1a. No posts
  1. Show message. End.
* 3a. Load fails
  1. Show error. Retry.

### UC-010: View Message

**Actor**: Logged-in User  
**Trigger**: Opens thread.  
**Preconditions**: Conversation exists.  
**Postconditions**: Messages displayed.

**MSS**:

1. User opens messages.
2. User selects one chat.
3. System loads history.

**Extensions**:

* 4a. Load fails
  1. Show error. Retry.

### UC-011: View Chat Interface

**Actor**: Logged-in User  
**Trigger**: Clicks chat tab/friend.  
**Preconditions**: Chat allowed.  
**Postconditions**: Chat UI shown.

**MSS**:

1. User opens chat panel.
2. User sees friends.
3. User clicks on friend username.
4. System loads chat.

**Extensions**:

* 3a. Not available
  1. Show restriction. End.

### UC-012: View Profile Page

**Actor**: Logged-in User  
**Trigger**: Clicks username/pic.  
**Preconditions**: Profile visible.  
**Postconditions**: Profile shown.

**MSS**:

1. User clicks profile.
2. System loads info.
3. System shows posts and friends.
4. System displays actions.

**Extensions**:

* 2a. Private profile
  1. Show restriction. End.

### UC-013: View Settings Page

**Actor**: Logged-in User  
**Trigger**: Clicks “Settings.”  
**Preconditions**: User is logged in.  
**Postconditions**: Settings UI shown.

**MSS**:

1. User click settings.
2. System loads preferences.
3. User views or updates.

**Extensions**:

* 2a. Load fails
  1. Show error. Retry or return.

## Conclusion

This document outlines the key functional and non-functional requirements for the social media web app, ensuring a structured approach to development and implementation.