# 

# 

# 

# 

# 

# **Kreator Konnect Project Report**

I22-0950 Azfar Bilal

I22-0920 Aashir Hameed

**1. Introduction**

Kreator Konnect is a creator-centric subscription platform designed to help digital creators monetize their content through tiered memberships and exclusive offerings. Inspired by Patreon, the app enables creators to build communities, interact with subscribers, and offer premium experiences in a secure and scalable way.

## **2. Rationale Behind the Project**

With the rise of independent content creation across platforms like YouTube, Instagram, and TikTok, many creators struggle to generate sustainable income due to algorithm changes or platform limitations. Kreator Konnect provides a reliable revenue stream by allowing fans to directly support creators. It also emphasizes privacy, two-factor authentication, and creator analytics, aligning with modern expectations of digital engagement.

## **3. Architecture of the Project**

Kreator Konnect follows a **Modular MVC-style Single Page Application (SPA)** architecture:

* **Frontend:** Built using React.js with vite app with modular components for each user-facing feature.
* **Backend:** Follows RESTful APIs likely powered by Node.js/Express.
* **State Management:** React context and hooks manage authentication and content loading.

### **Justification for the Architecture:**

* **Modularity** ensures maintainability and scalability as new features (like dark mode or subscription tiers) are added.
* **SPA architecture** provides seamless navigation and improved user experience without full page reloads.
* **Component-driven development** allows parallel work by multiple team members on different UI features.

## 

## **4. Complete Product Backlog**

Sprint 1 (March 3–March 14, 2025)

1. **Log In to the Platform**

**User Story**: As a user, I want to log in to the platform so that I can access my account and use its features.

**Tasks**:

Create user schema (email, hashed password). Develop API (POST /api/login) with JWT authentication. Design React login form with email/password fields. **Acceptance Criteria**:



Users can log in with valid credentials.



Invalid credentials show an error message.



**Rules**: Passwords must be hashed.

1. **Sign Up for a New Account**

**User Story**: As a user, I want to sign up for a new account so that I can join the platform and start engaging.

**Tasks**:

Extend user schema for username.



Develop API (POST /api/signup) with email verification.



Design React signup form with validation.



**Acceptance Criteria**:

Users can sign up with username, email, and password.



A confirmation email is sent.



**Rules**: Email must be unique.

1. **View My Profile**

**User Story**: As a user, I want to view my profile so that I can see my posts and personal information. **Tasks**:

Create profile schema (username, bio, profile\_picture).



Develop API (GET /api/profile).



Design React profile page.



**Acceptance Criteria**:

Users see their username, bio, and profile picture.



Social media links are clickable.



**Rules**: None.

1. **Edit My Profile**

**User Story**: As a user, I want to edit my profile so that I can update my bio, picture, and social links. **Tasks**:

Develop API (PUT /api/profile) for updates.



Implement file upload for profile pictures.



Design React edit profile form.



**Acceptance Criteria**:

Users can update bio, picture, and links.



Changes are reflected immediately.



**Rules**: Image size <2MB.

1. **Two-Factor Authentication**

**User Story**: As a user, I want to enable two-factor authentication so that my account is more secure. **Tasks**:

Integrate Speakeasy for 2FA codes.



Develop API (POST /api/2fa).



Design React 2FA toggle and code input UI.



**Acceptance Criteria**:

Users can enable/disable 2FA.



A verification code is required for login when enabled.



**Rules**: None.

Sprint 2 (April 1–April 14, 2025)

1. **View My Creator Dashboard**

**User Story**: As a creator, I want to view my dashboard so that I can monitor my earnings, subscribers, and engagement.

**Tasks**:

Create dashboard schema for metrics.



Develop API (GET /api/dashboard).



Design React dashboard with charts.



**Acceptance Criteria**:

Creators see earnings, subscriber count, and engagement metrics.



Metrics are displayed visually.



**Rules**: None.

1. **Toggle Between Light and Dark Modes**

**User Story**: As a user, I want to switch between light and dark modes so that I can choose a comfortable viewing experience.

**Tasks**:

Implement CSS for dark mode.



Develop API to save mode preference.



Design React toggle button.



**Acceptance Criteria**:

Users can toggle modes via a button.



Preference persists across sessions.



**Rules**: None.

1. **Request a Payout**

**User Story**: As a creator, I want to request a payout so that I can withdraw my earnings. **Tasks**:

Develop API (POST /api/payout) with validation.



Integrate Stripe for payouts.



Design React payout request form.



**Acceptance Criteria**:

Creators can request payouts above $50.



Requests are recorded in history.



**Rules**: Minimum payout $50.

1. **View My Payout History**

**User Story**: As a creator, I want to view my payout history so that I can track my withdrawal requests. **Tasks**:

Extend payout schema for history.



Develop API (GET /api/payout-history).



Design React history table.



**Acceptance Criteria**:

Creators see a table of past payouts.



Empty history shows a message.



**Rules**: None.

1. **Subscribe to a Creator**

**User Story**: As a user, I want to subscribe to a creator so that I can access their exclusive content. **Tasks**:

Create subscription schema.



Develop API (POST /api/subscribe).



Design React subscription form.



**Acceptance Criteria**:

Users can select and pay for a subscription tier.



Subscription is activated immediately.



**Rules**: None.

1. **Manage My Subscription**

**User Story**: As a user, I want to manage my subscription so that I can renew, cancel, or change my plan. **Tasks**:

Develop API (PUT /api/subscription).



Design React subscription management page.



**Acceptance Criteria**:

Users can cancel/renew subscriptions.



Changes take effect at the next billing cycle.



**Rules**: None.

1. **View Exclusive Content**

**User Story**: As a subscriber, I want to view exclusive content so that I can access posts available only to subscribers.

**Tasks**:

Extend content schema with is\_exclusive flag.



Develop API (GET /api/content).



Design React content display component.



**Acceptance Criteria**:

Subscribers see exclusive content.



Non-subscribers see a subscribe prompt.



**Rules**: None.

1. **Content Scheduling**

**User Story**: As a creator, I want to schedule posts so that my content is published automatically at a set time.

**Tasks**:

Create post schema with schedule fields.



Develop API (POST /api/schedule-post).



Design React scheduling form.



**Acceptance Criteria**:

Creators can schedule posts with date/time.



Posts are published automatically.



**Rules**: None.

1. **Gift Subscription**

**User Story**: As a user, I want to give a subscription to another user so that I can support a creator on their behalf.

**Tasks**:

Extend subscription schema for gifts.



Develop API (POST /api/gift-subscription).



Design React gift form.



**Acceptance Criteria**:

Users can gift a subscription.



Recipients receive a notification.



**Rules**: None.

1. **Dark Mode Feature**

**User Story**: As a user, I want to enable dark mode so that I can reduce eye strain. **Tasks**:

Implement dark mode CSS.



Develop API to persist preference.



Design React toggle UI.



**Acceptance Criteria**:

Users can enable dark mode.



UI adapts seamlessly.



**Rules**: None.

Sprint 3 (April 21–May 2, 2025)

1. **Create a New Post**

**User Story**: As a creator, I want to create a new post so that I can share content with my audience. **Tasks**:

Create post schema (title, content, media).



Develop API (POST /api/post).



Design React post creation form.



**Acceptance Criteria**:

Creators can create posts with text and media.



Posts appear in the feed.



**Rules**: Media size <5MB.

1. **Like a Post**

**User Story**: As a user, I want to like a post so that I can show appreciation for content I enjoy. **Tasks**:

Create like schema (user\_id, post\_id).



Develop API (POST /api/like).



Design React like button.



**Acceptance Criteria**:

Users can like/unlike posts.



Like count updates in real-time.



**Rules**: None.

1. **Comment on a Post**

**User Story**: As a user, I want to comment on a post so that I can engage with the creator and other users.

**Tasks**:

Create comment schema (user\_id, post\_id, text).



Develop API (POST /api/comment).



Design React comment section.



**Acceptance Criteria**:

Users can post comments.



Comments are displayed below posts.



**Rules**: Comment length <500 chars.

1. **Browse the Feed**

**User Story**: As a user, I want to browse a feed of posts so that I can discover content from creators I follow.

**Tasks**:

Develop API (GET /api/feed) with pagination.



Design React feed component.



**Acceptance Criteria**:

Users see posts from followed creators.



Feed supports infinite scrolling.



**Rules**: None.

1. **Send and Receive Messages**

**User Story**: As a user, I want to send and receive messages so that I can communicate with other users or creators.

**Tasks**:

Create message schema (sender\_id, receiver\_id, text). Develop API (POST /api/message, GET /api/messages). Design React messaging UI. **Acceptance Criteria**:



Users can send/receive messages.



Messages are displayed in a chat interface.



**Rules**: None.

## **5. Complete Project Planner and Gantt Chart**

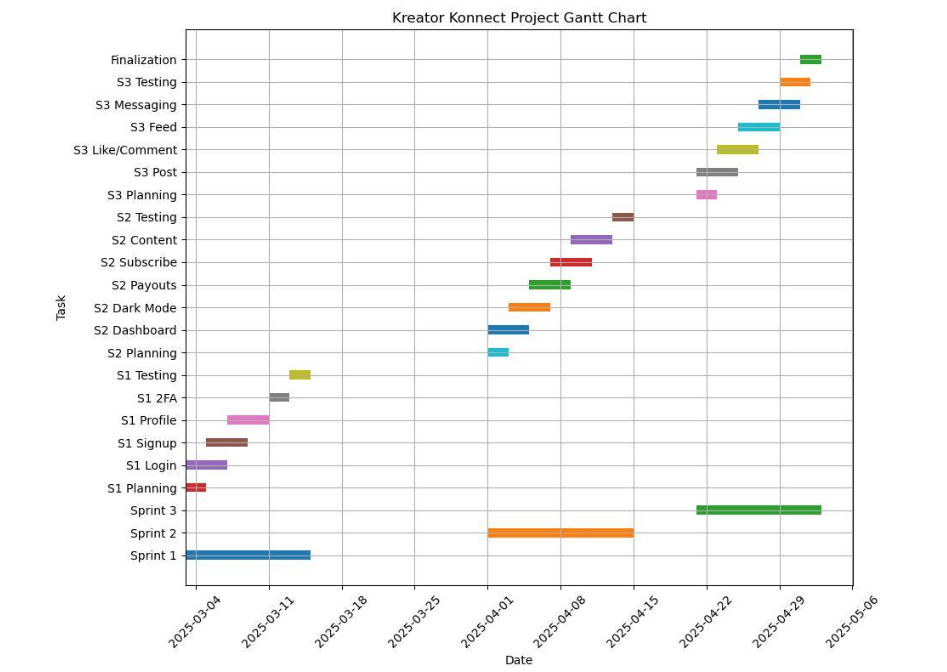
Project Planner

This document outlines the project timeline for Kreator Konnect, covering three sprints (March 3–May 2, 2025). Tasks are assigned to Aashir Hameed (frontend, UI/UX) and Azfar Bilal (backend, APIs).

Timeline and Tasks

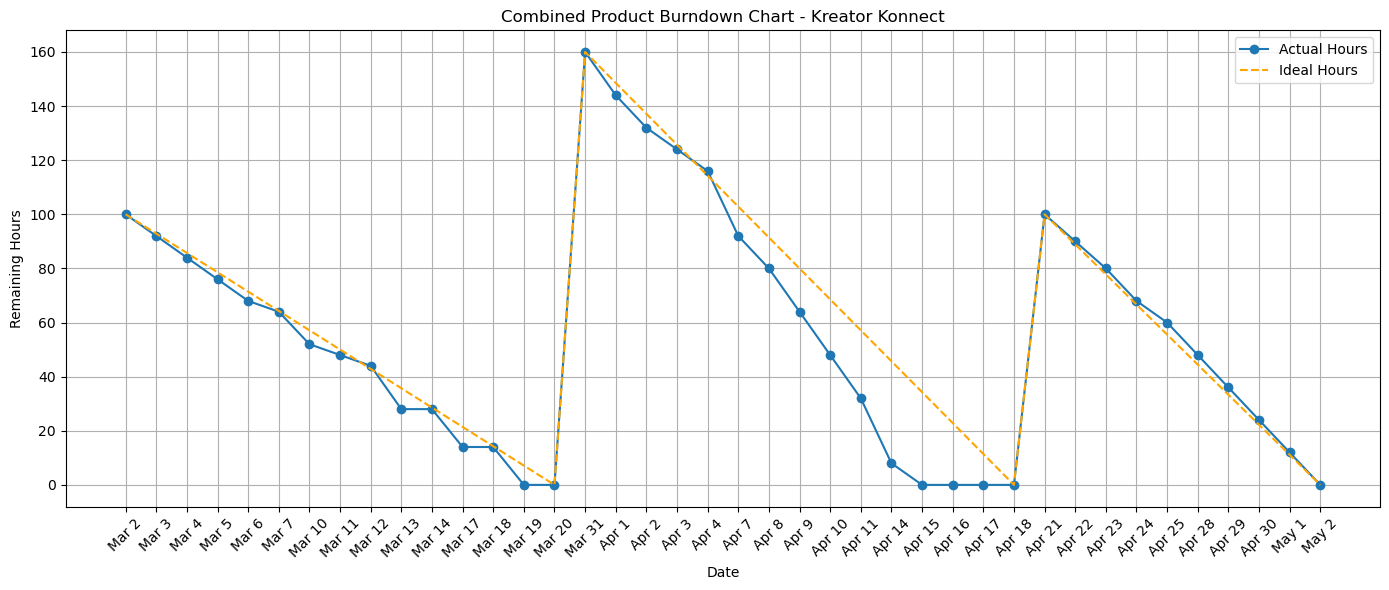
| **Date** | **Sprint** | **User Story/Task** | **Responsible** |
| --- | --- | --- | --- |
|  |  |  |  |
| Mar 3 | S1 | Sprint Planning: Assign 5 user stories | Aashir, Azfar |
|  |  |  |  |
| Mar 3–4 | S1 | Log In: Develop login API and form | Azfar (API), Aashir (UI) |
|  |  |  |  |
| Mar 5–6 | S1 | Sign Up: Develop signup API and form | Azfar (API), Aashir (UI) |
|  |  |  |  |
| Mar 7–8 | S1 | View Profile: Develop profile API and page | Azfar (API), Aashir (UI) |
|  |  |  |  |
| Mar 9–10 | S1 | Edit Profile: Develop update API and form | Azfar (API), Aashir (UI) |
|  |  |  |  |
| Mar 11–12 | S1 | 2FA: Develop 2FA API and UI | Azfar (API), Aashir (UI) |
|  |  |  |  |
| Mar 13 | S1 | Testing: Unit and integration tests | Aashir, Azfar |
|  |  |  |  |
| Mar 14 | S1 | Sprint Review/Retrospective: Demo and reflect | Aashir, Azfar |
|  |  |  |  |
| Apr 1 | S2 | Sprint Planning: Assign 10 user stories | Aashir, Azfar |
|  |  |  |  |
| Apr 1–2 | S2 | Dashboard: Develop metrics API and charts | Azfar (API), Aashir (UI) |
|  |  |  |  |
| Apr 3–4 | S2 | Dark Mode: Implement CSS and toggle | Aashir (UI) |
|  |  |  |  |
| Apr 5–6 | S2 | Payout Request/History: Develop APIs and UI | Azfar (API), Aashir (UI) |
|  |  |  |  |
| Apr 7–8 | S2 | Subscribe/Manage: Develop APIs and forms | Azfar (API), Aashir (UI) |
|  |  |  |  |
| Apr 9–10 | S2 | Exclusive Content/Scheduling: Develop APIs and UI | Azfar (API), Aashir (UI) |
|  |  |  |  |
| Apr 11–12 | S2 | Gift Subscription: Develop API and form | Azfar (API), Aashir (UI) |
|  |  |  |  |
| Apr 13 | S2 | Testing: Unit and integration tests | Aashir, Azfar |
|  |  |  |  |
| Apr 14 | S2 | Sprint Review/Retrospective: Demo and reflect | Aashir, Azfar |
|  |  |  |  |
| Apr 21 | S3 | Sprint Planning: Assign 5 user stories | Aashir, Azfar |
|  |  |  |  |
| Apr 21–22 | S3 | Create Post: Develop API and form | Azfar (API), Aashir (UI) |
|  |  |  |  |
| Apr 23–24 | S3 | Like/Comment: Develop APIs and UI | Azfar (API), Aashir (UI) |
|  |  |  |  |
| Apr 25–26 | S3 | Feed: Develop API and component | Azfar (API), Aashir (UI) |
|  |  |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
| Apr 27–28 | S3 | Messaging: Develop APIs and chat UI | Azfar (API), Aashir (UI) |
|  |  |  |  |
| Apr 29–30 | S3 | Testing: Unit and integration tests | Aashir, Azfar |
|  |  |  |  |
| May 1 | S3 | Sprint Review/Retrospective: Demo and reflect | Aashir, Azfar |
|  |  |  |  |
| May 2 | S3 | Project Finalization: Documentation and deployment | Aashir, Azfar |



## **6. Complete Product Burn-Down Chart**

Here’s a combined Sprint 1, 2, and 3 burndown chart:



## **7. Trello Snapshots**

## 

## 

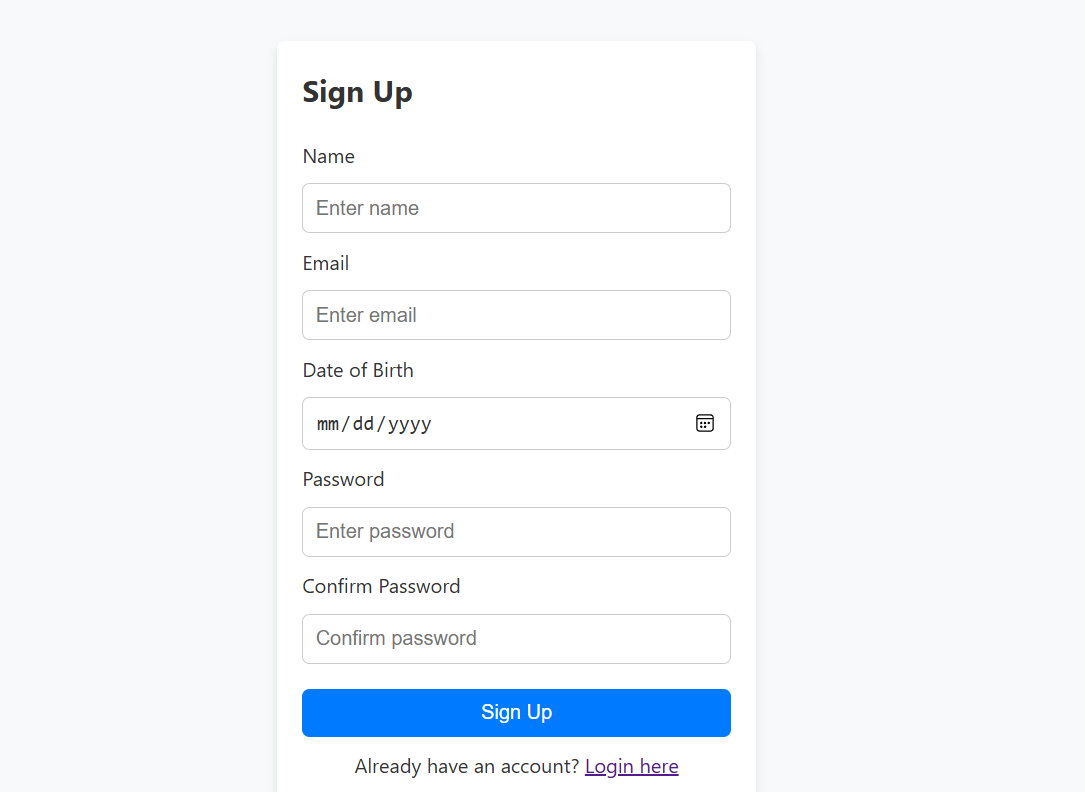
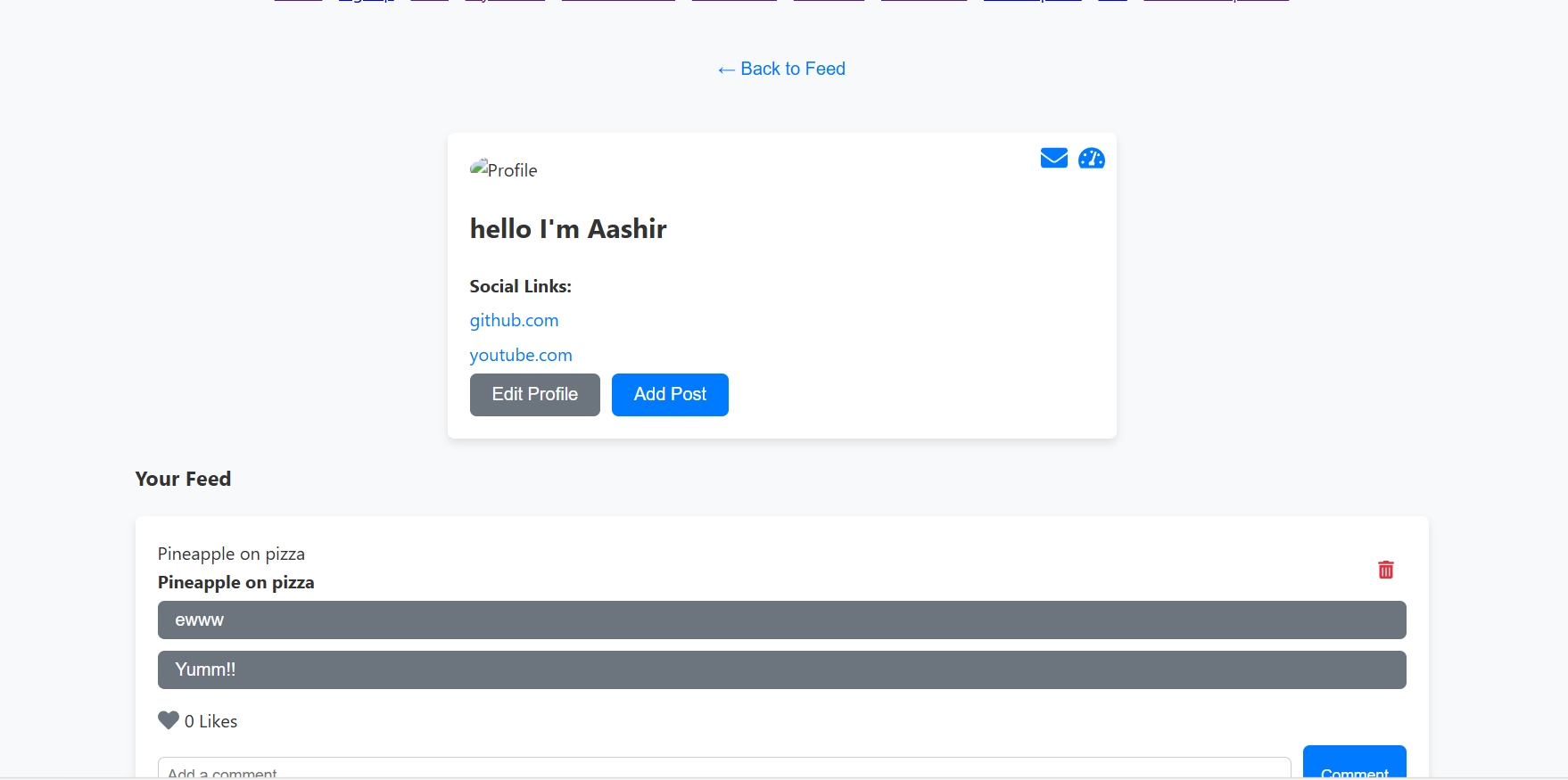
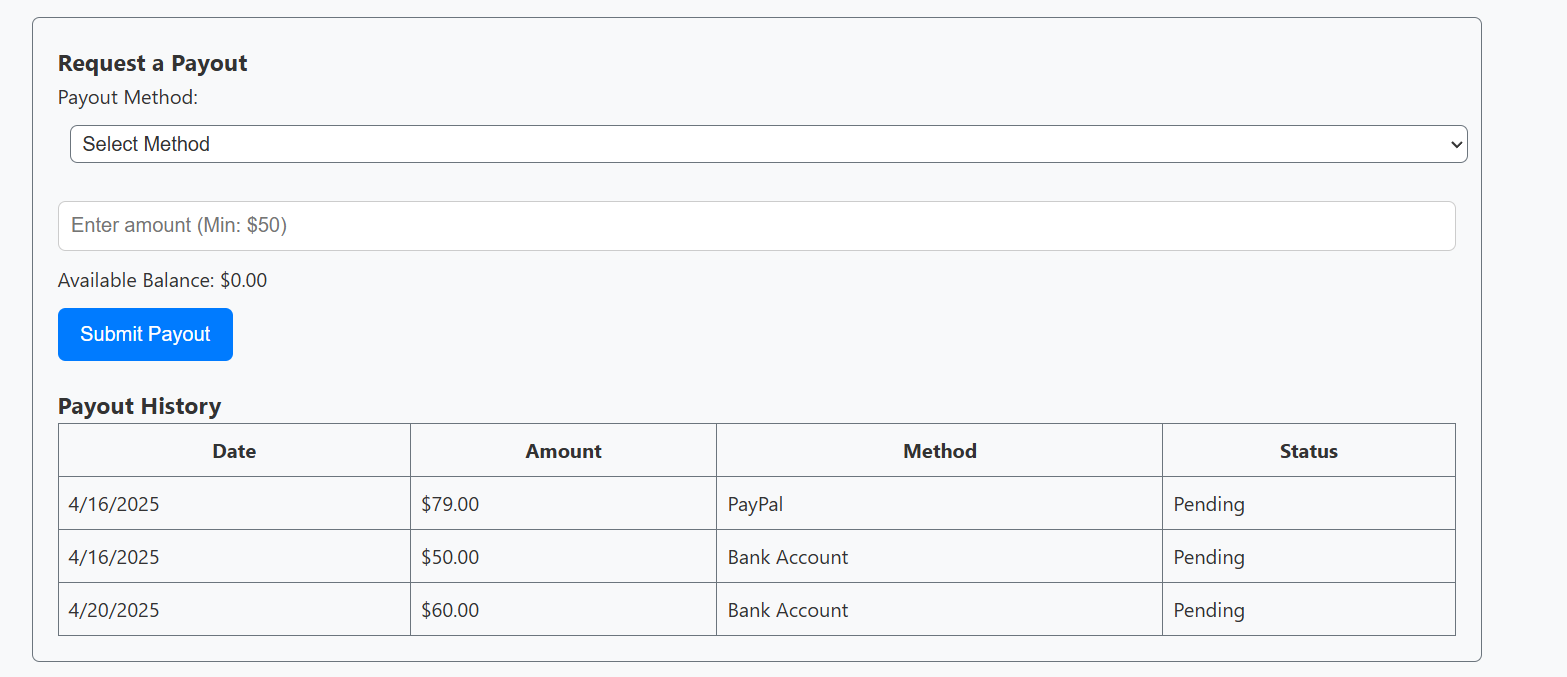
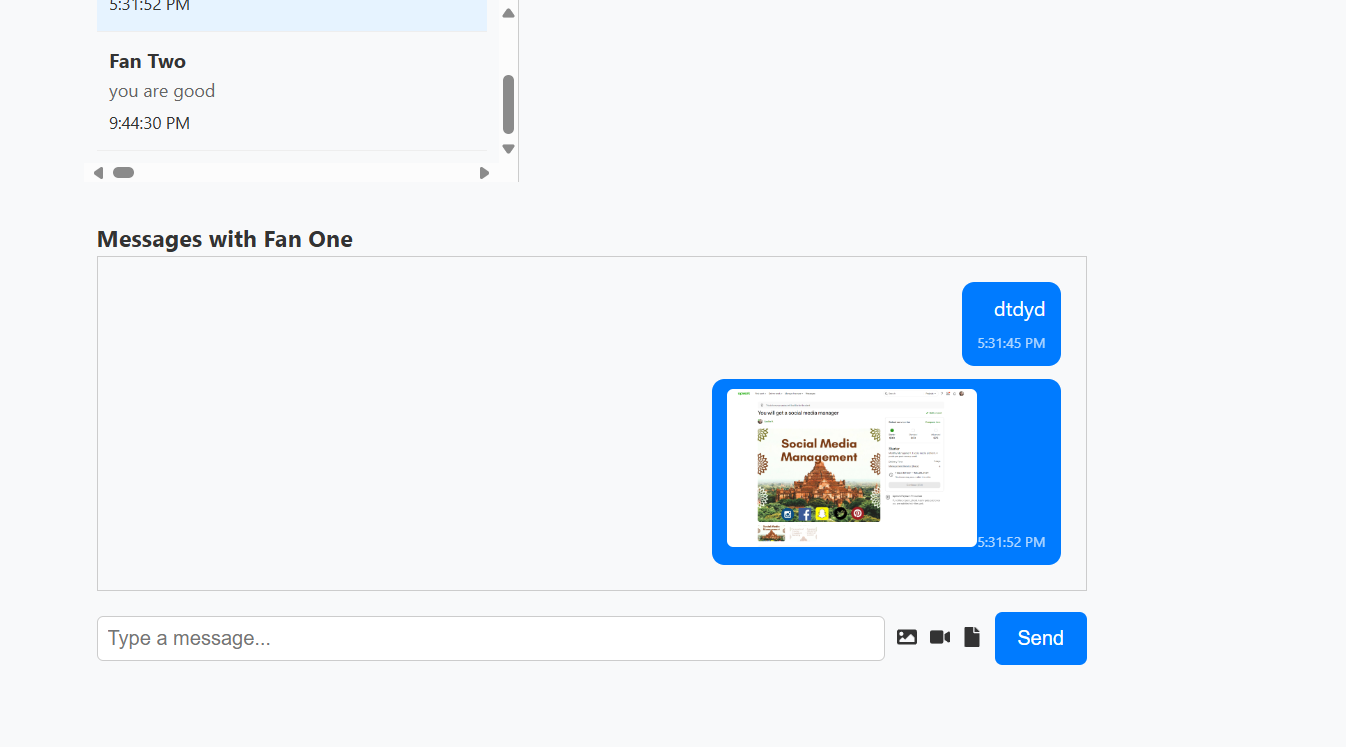
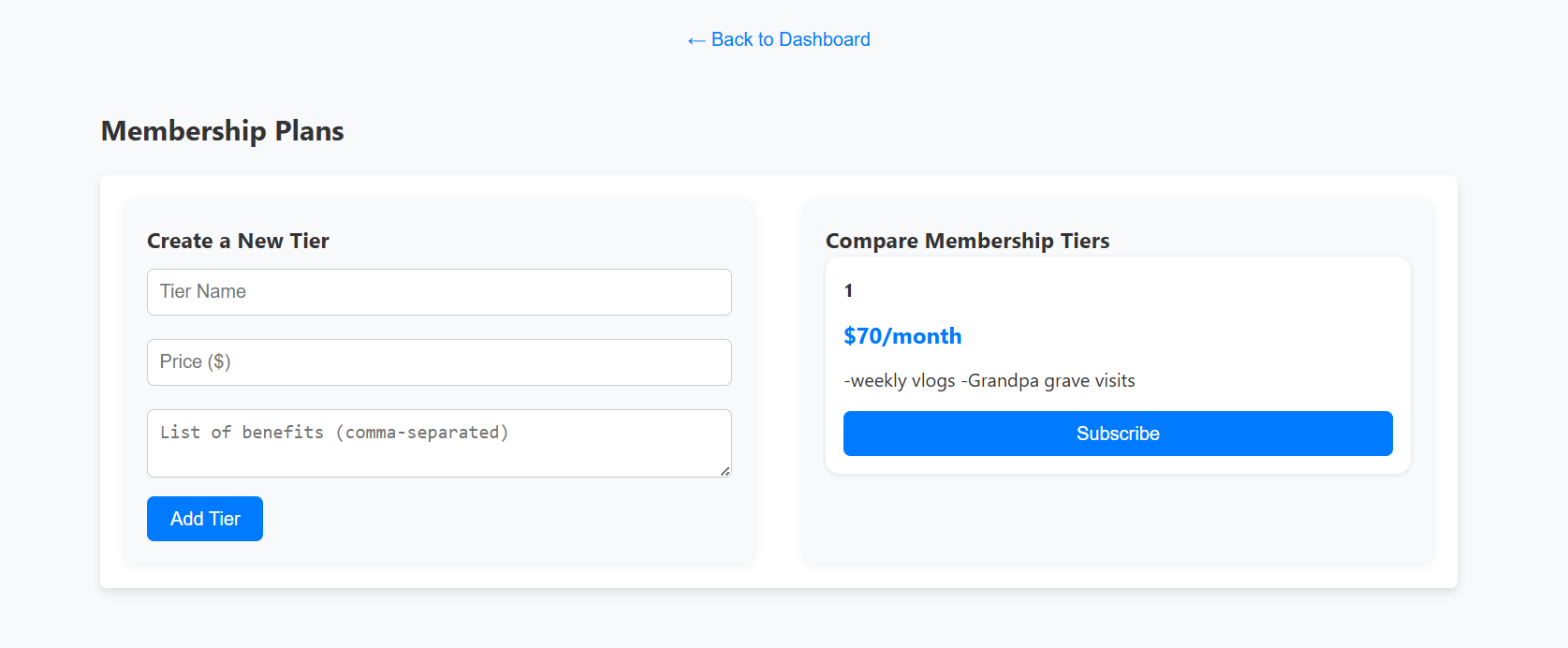
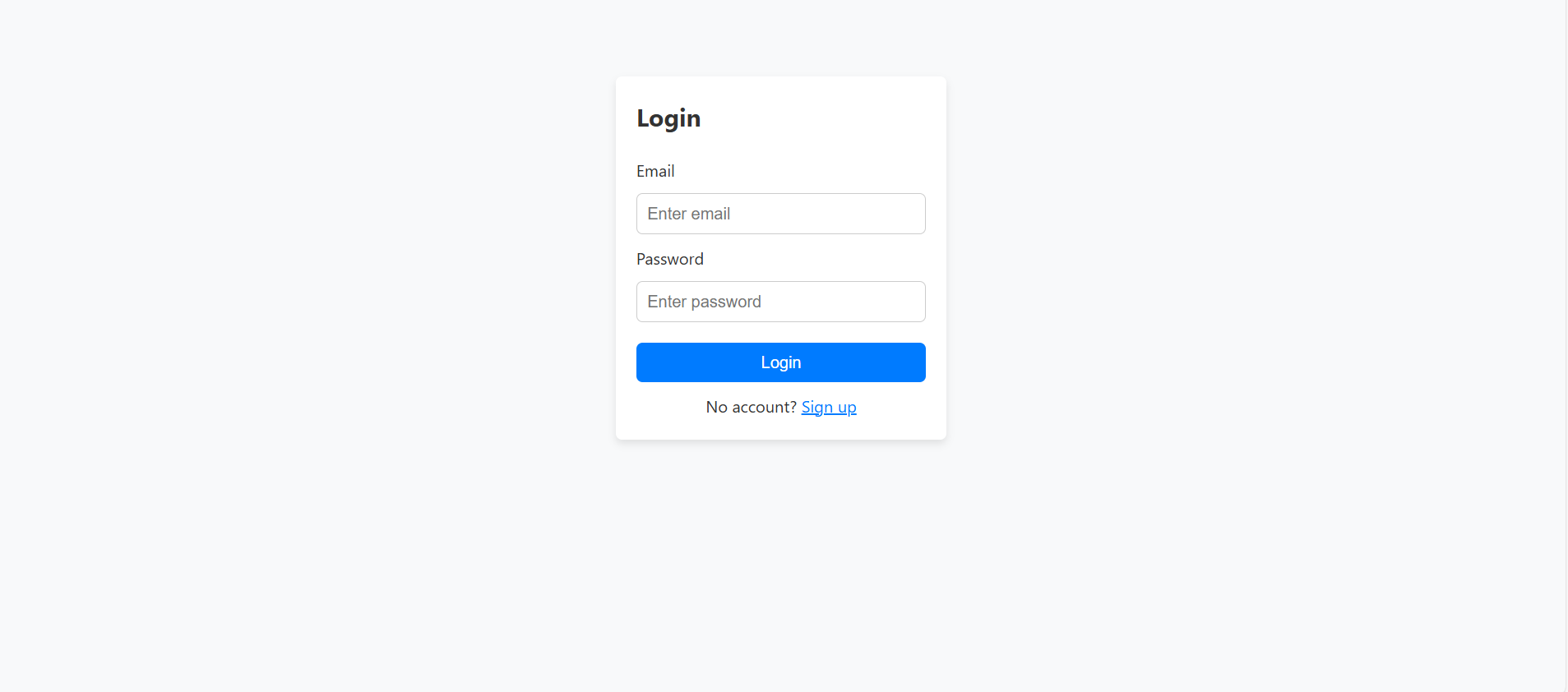
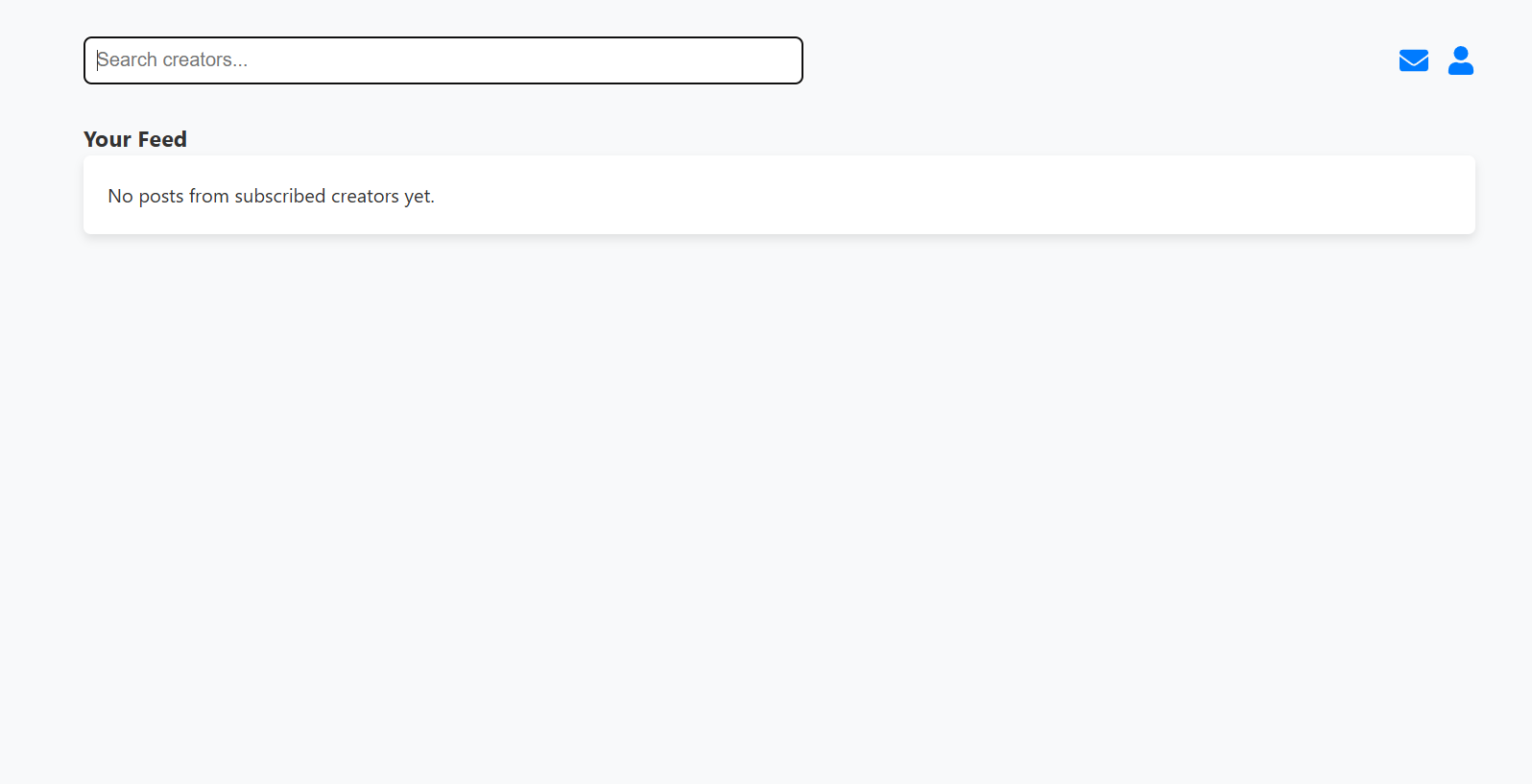
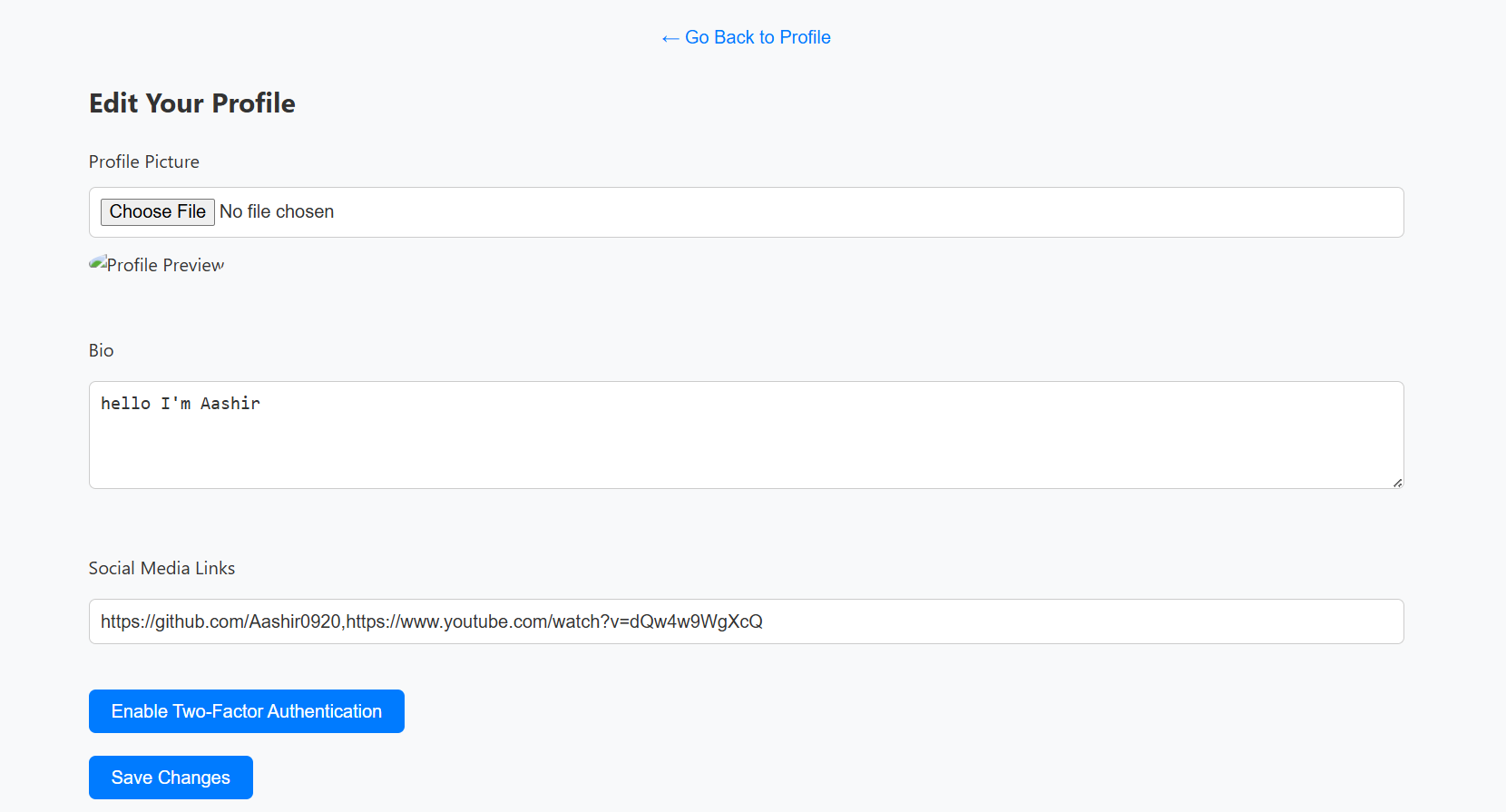
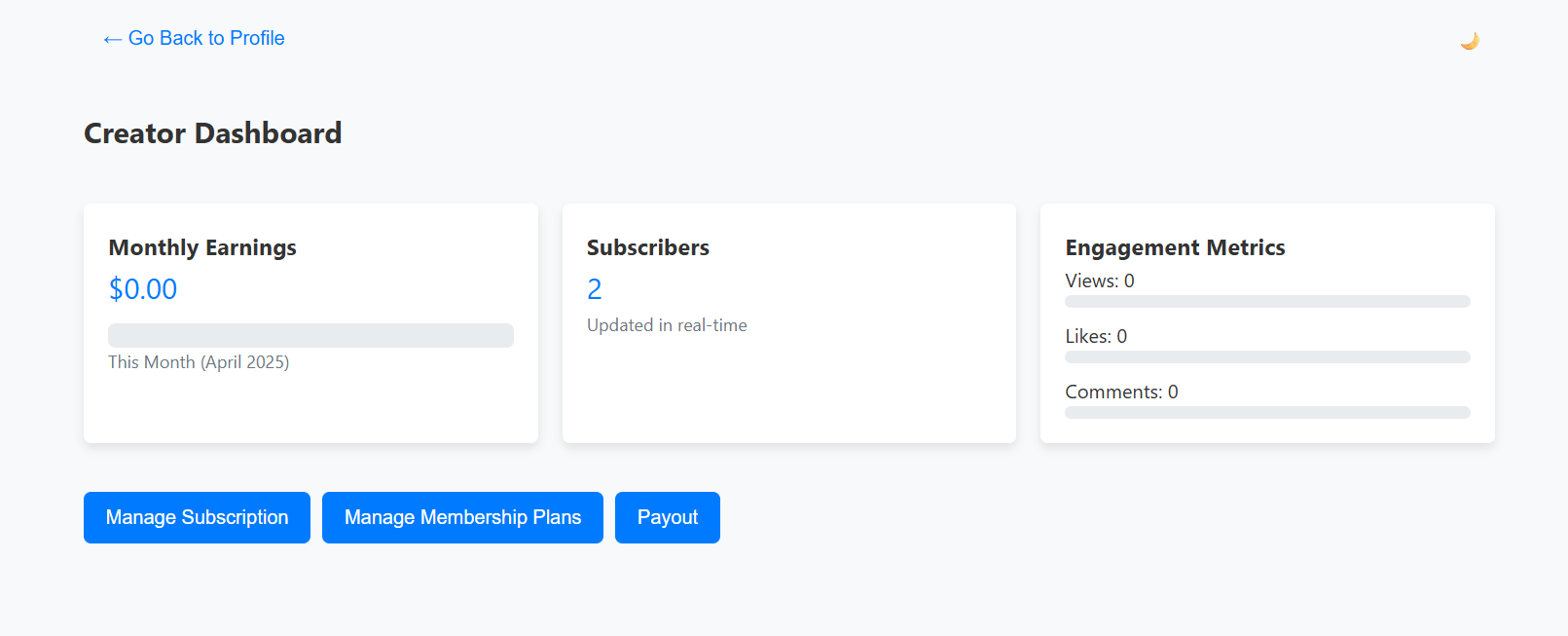
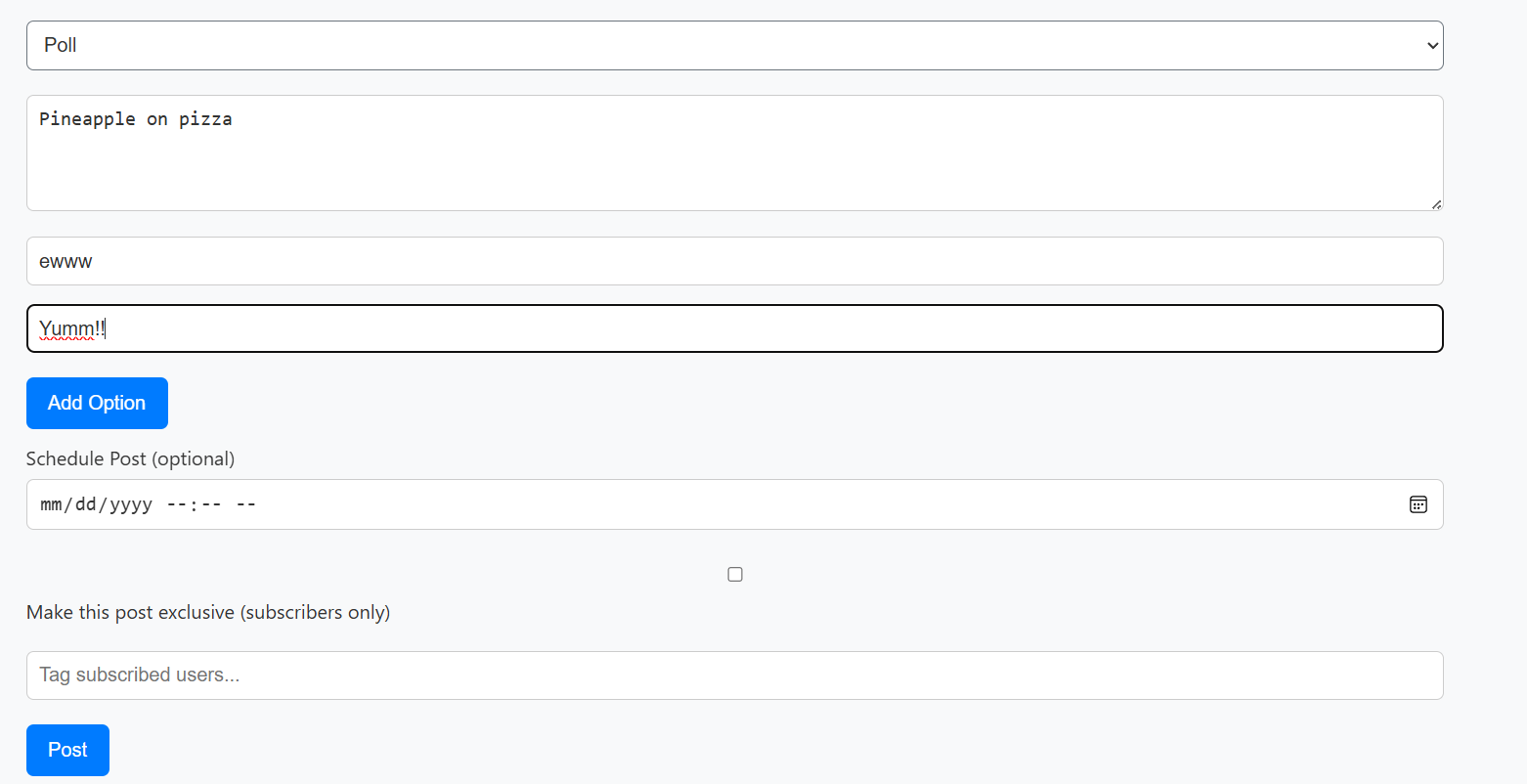
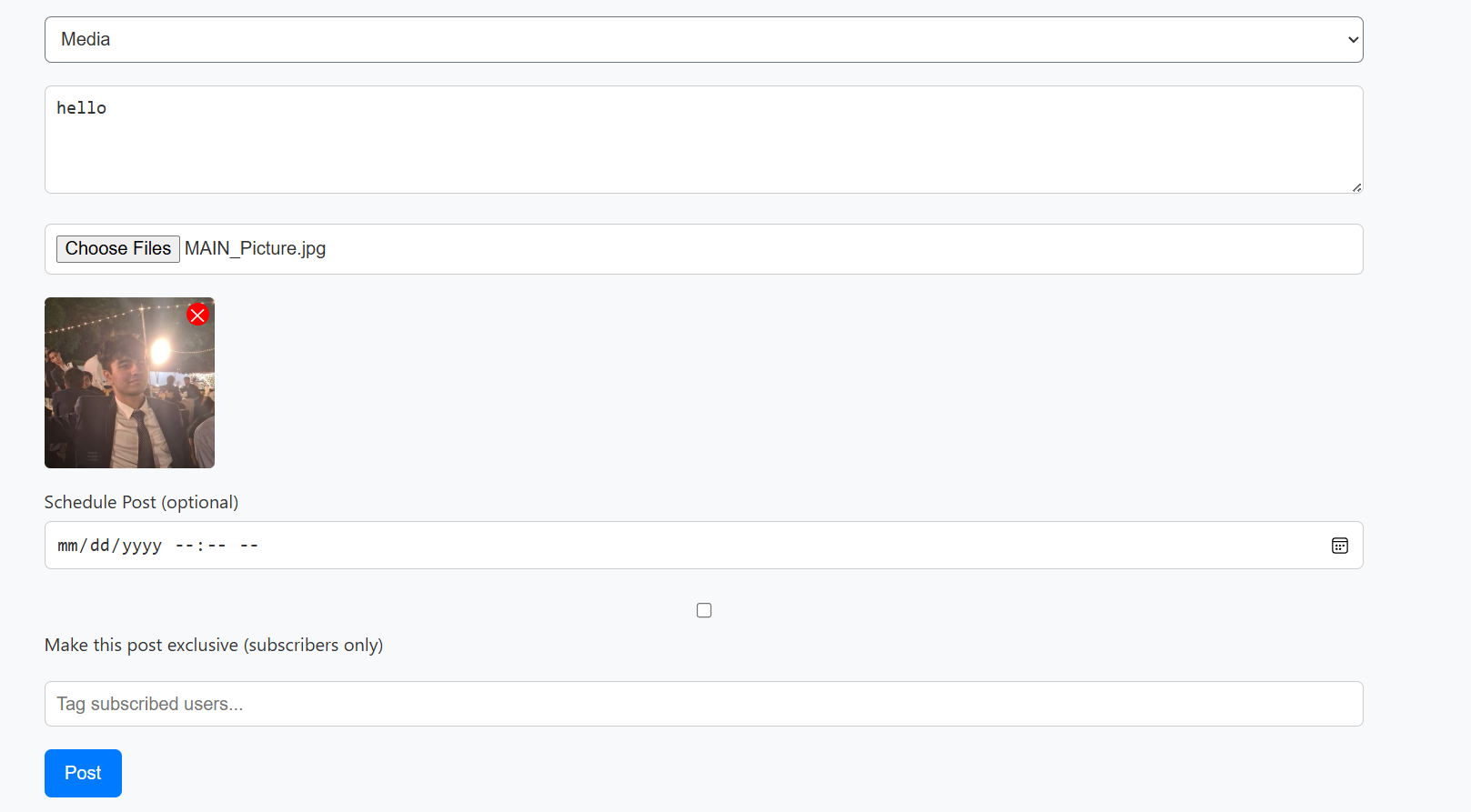
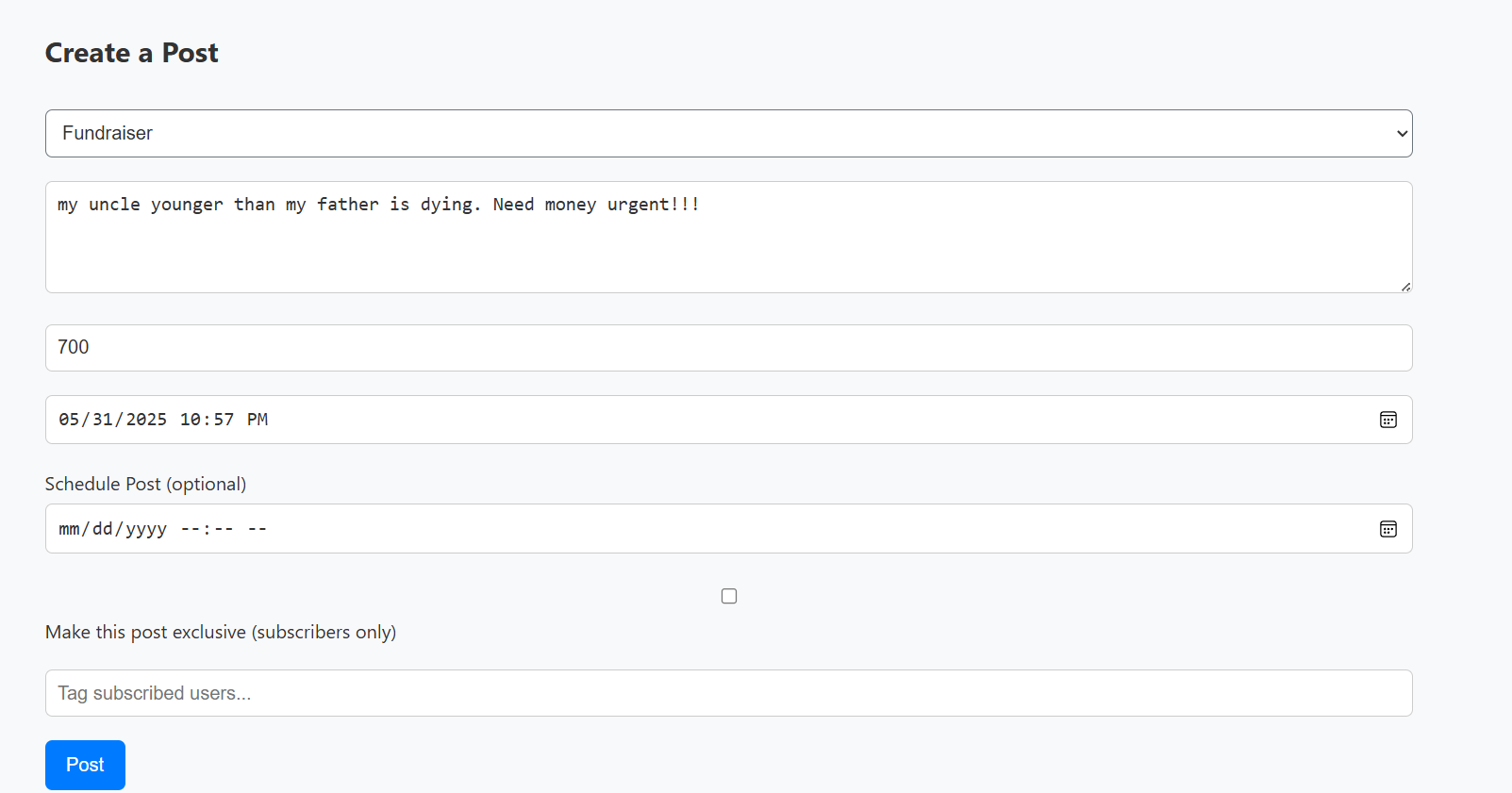
## 

## 

## 

## 

## **8. GUI Snapshots and Testing**



ECP Testing for Front-End Implementation

This document provides equivalence class partitioning (ECP), test case counts for weak and strong ECP testing, and boundary value analysis (BVA) for all GUI fields in the front-end implementation of Kreator Konnect across Sprint 1 (5 user stories), Sprint 2 (10 user stories), and Sprint 3 (5 user stories).

# Sprint 1: GUI Fields

1. **Login Form (Log In to the Platform) Field: Email**

| **ECP Table**:  **Partition** | **Description** | **Valid/Invalid** | **Example** |
| --- | --- | --- | --- |
| EC1 | Valid email format | Valid | [user@example.com](mailto:user@example.com) |
| EC2 | Invalid email (no @) | Invalid | userexample.com |
| EC3 | Invalid email (no domain) | Invalid | user@ |
| EC4 | Empty email | Invalid | "" |

 **Test Cases**:

 **Weak ECP**: 4 test cases (one per partition).

 **Strong ECP**: 4 test cases (single field, no combinations).

**BVA**:

 Min length: 1 char (invalid, e.g., "a").

 Max length: 254 chars (valid, per RFC 5321).  Edge cases: "@example.com", "user@.com".

**Field: Password**

| **ECP Table**:  **Partition** | **Description** | **Valid/Invalid** | **Example** |
| --- | --- | --- | --- |
| EC1 | Valid password (8–50 chars) | Valid | Password123 |
| EC2 | Too short (<8 chars) | Invalid | Pass123 |
| EC3 | Too long (>50 chars) | Invalid | [51-char string] |
| EC4 | Empty password | Invalid | "" |

 **Test Case**

* **Weak ECP**: 4 test cases.
* **Strong ECP**: 4 test cases.

**BVA**:

* Min: 7 chars (invalid), 8 chars (valid).
* Max: 50 chars (valid), 51 chars (invalid

1. **Signup Form (Sign Up for a New Account) Field: Username**

 **ECP Table**:

**Partition Description Valid/Invalid Example**



EC1 Valid username (3–20 chars, alphanumeric)

Valid user123



EC2 Too short ( chars) Invalid us

| EC3 | Too long (>20 chars) | Invalid | user123456789012345678 |
| --- | --- | --- | --- |
| EC4 | Invalid characters | Invalid | user@123 |
| EC5 | Empty username | Invalid | "" |

**Test Cases**:

 **Weak ECP**: 5 test cases.

 **Strong ECP**: 5 test cases.

**BVA**:

 Min: 2 chars (invalid), 3 chars (valid).

 Max: 20 chars (valid), 21 chars (invalid).

**Field: Email** (Same as Login Form)  See above for ECP and BVA.

**Field: Password** (Same as Login Form)  See above for ECP and BVA.

**Field: Confirm Password**

 **ECP Table**:

**Partition Description Valid/Invalid Example**



EC1 Matches password Valid Password123 (if password is

Password123)



EC2 Does not match password

Invalid Pass123 (if password is Password123)



EC3 Empty Invalid ""

**Test Cases**:

 **Weak ECP**: 3 test cases.

 **Strong ECP**: 12 test cases (combined with Password: 4 × 3).

 **BVA**: None (matching is boolean, no length constraints).

1. **Profile Edit Form (Edit My Profile) Field: Bio**

| **ECP Table**:  **Partition** | **Description** | **Valid/Invalid** | **Example** |
| --- | --- | --- | --- |
| EC1 | Valid bio (0–200 chars) | Valid | "Content creator" |
| EC2 | Too long (>200 chars) | Invalid | [201-char string] |
| EC3 | Empty bio | Valid | "" |

 **Test Cases**:

 **Weak ECP**: 3 test cases.

 **Strong ECP**: 3 test cases.

**BVA**:

 Min: 0 chars (valid).

 Max: 200 chars (valid), 201 chars (invalid).

**Field: Profile Picture (File Upload)**

| **ECP Table**:  **Partition** | **Description** | **Valid/Invalid** | **Example** |
| --- | --- | --- | --- |
| EC1 | Valid image (PNG/JPG, ≤2MB) | Valid | image.png (1MB) |
| EC2 | Invalid format (e.g., GIF) | Invalid | image.gif |
| EC3 | File too large (>2MB) | Invalid | image.png (3MB) |
| EC4 | No file | Valid | None |

 **Test Cases**:

 **Weak ECP**: 4 test cases.

 **Strong ECP**: 4 test cases.

**BVA**:

 Min size: 0 bytes (valid, no file).

 Max size: 2MB (valid), 2MB+1byte (invalid).

**Field: Social Links**

| **ECP Table**:  **Partition** | **Description** | **Valid/Invalid** | **Example** |
| --- | --- | --- | --- |
| EC1 | Valid URL (≤100 chars) | Valid | https://twitter.com/user |
| EC2 | Invalid URL format | Invalid | htt://invalid |
| EC3 | Too long (>100 chars) | Invalid | [101-char URL] |
| EC4 | Empty | Valid | "" |

**Test Cases**:

 **Weak ECP**: 4 test cases.

 **Strong ECP**: 4 test cases (per link, multiple links tested separately).

**BVA**:

 Min: 0 chars (valid).

 Max: 100 chars (valid), 101 chars (invalid).

1. **Two-Factor Authentication Field: 2FA Toggle (Checkbox)**

| **ECP Table**:  **Partition** | **Description** | **Valid/Invalid** | **Example** |
| --- | --- | --- | --- |
| EC1 | Enabled | Valid | Checked |
| EC2 | Disabled | Valid | Unchecked |

 **Test Cases**:

 **Weak ECP**: 2 test cases.

 **Strong ECP**: 2 test cases.

 **BVA**: None (binary input).

**Field: 2FA Code (Numeric)**

| **ECP Table**:  **Partition** | **Description** | **Valid/Invalid** | **Example** |
| --- | --- | --- | --- |
| EC1 | Valid 6-digit code | Valid | 123456 |
| EC2 | Non-6-digit (e.g., 5 or 7 digits) | Invalid | 12345, 1234567 |
| EC3 | Non-numeric | Invalid | abcdef |
| EC4 | Empty | Invalid | "" |

 **Test Cases**:

 **Weak ECP**: 4 test cases.

 **Strong ECP**: 4 test cases.

**BVA**:

 Min digits: 5 (invalid), 6 (valid).

 Max digits: 6 (valid), 7 (invalid).

# Sprint 2: GUI Fields

1. **Toggle Between Light and Dark Modes Field: Mode Toggle (Checkbox/Button)**

 **ECP Table**: Same as 2FA Toggle (Enabled/Disabled).

 **Test Cases**: Weak/Strong: 2 test cases.

 **BVA**: None.

1. **Request a Payout**

**Field: Payout Amount (Number)**

| **ECP Table**:  **Partition** | **Description** | **Valid/Invalid** | **Example** |
| --- | --- | --- | --- |
| EC1 | Valid amount (≥50, ≤balance) | Valid | 100 |
| EC2 | Too low (<50) | Invalid | 49 |
| EC3 | Exceeds balance | Invalid | 1000 (if balance is 500) |
| EC4 | Non-numeric | Invalid | abc |
| EC5 | Empty | Invalid | "" |

 **Test Cases**:

 **Weak ECP**: 5 test cases.

 **Strong ECP**: 5 test cases.

**BVA**:

 Min: 49 (invalid), 50 (valid).

 Max: Balance (valid), Balance+1 (invalid).

**Field: Payment Method (Dropdown)**

| **ECP Table**:  **Partition** | **Description** | **Valid/Invalid** | **Example** |
| --- | --- | --- | --- |
| EC1 | Valid option | Valid | Bank, PayPal |
| EC2 | No selection | Invalid | None |

 **Test Cases**:

 **Weak ECP**: 2 test cases.

 **Strong ECP**: 10 test cases (combined with Amount: 5 × 2).

 **BVA**: None (finite options).

1. **Subscribe to a Creator**

**Field: Subscription Tier (Dropdown)**

| **ECP Table**:  **Partition** | **Description** | **Valid/Invalid** | **Example** |
| --- | --- | --- | --- |
| EC1 | Valid tier | Valid | Basic, Premium |
| EC2 | No selection | Invalid | None |

 **Test Cases**:

 **Weak ECP**: 2 test cases.

 **Strong ECP**: 2 test cases.

 **BVA**: None.

**Field: Payment Details (Text, e.g., Card Number)**

| **ECP Table**:  **Partition** | **Description** | **Valid/Invalid** | **Example** |
| --- | --- | --- | --- |
| EC1 | Valid card number (16 digits) | Valid | 1234567890123456 |
| EC2 | Invalid length (<16 or >16) | Invalid | 123456789012345 |
| EC3 | Non-numeric | Invalid | abcd1234abcd1234 |
| EC4 | Empty | Invalid | "" |

**Test Cases**:

 **Weak ECP**: 4 test cases.

 **Strong ECP**: 8 test cases (combined with Tier: 4 × 2).

**BVA**:

 Min digits: 15 (invalid), 16 (valid).

 Max digits: 16 (valid), 17 (invalid).

1. **Manage My Subscription Field: Action (Dropdown)**

| **ECP Table**:  **Partition** | **Description** | **Valid/Invalid** | **Example** |
| --- | --- | --- | --- |
| EC1 | Valid action | Valid | Renew, Cancel |
| EC2 | No selection | Invalid | None |

 **Test Cases**:

 **Weak ECP**: 2 test cases.

 **Strong ECP**: 2 test cases.

 **BVA**: None.

1. **Content Scheduling**

**Field: Post Content (Textarea)**

| **ECP Table**:  **Partition** | **Description** | **Valid/Invalid** | **Example** |
| --- | --- | --- | --- |
| EC1 | Valid content (1–1000 chars) | Valid | "New post content" |
| EC2 | Too long (>1000 chars) | Invalid | [1001-char string] |
| EC3 | Empty | Invalid | "" |

 **Test Cases**:

 **Weak ECP**: 3 test cases.

 **Strong ECP**: 3 test cases.

**BVA**:

 Min: 0 chars (invalid), 1 char (valid).

 Max: 1000 chars (valid), 1001 chars (invalid).

**Field: Schedule Date (Date Input)**

| **ECP Table**:  **Partition** | **Description** | **Valid/Invalid** | **Example** |
| --- | --- | --- | --- |
| EC1 | Future date | Valid | 2025-05-01 |
| EC2 | Past date | Invalid | 2025-04-01 |
| EC3 | Invalid format | Invalid | 2025-13-01 |
| EC4 | Empty | Invalid | "" |

 **Test Cases**:

 **Weak ECP**: 4 test cases.

 **Strong ECP**: 12 test cases (combined with Content: 3 × 4).

**BVA**:

 Min: Current date (invalid), Current date+1 (valid).  Max: None (future dates valid).

**Field: Schedule Time (Time Input)**

| **ECP Table**:  **Partition** | **Description** | **Valid/Invalid** | **Example** |
| --- | --- | --- | --- |
| EC1 | Valid 24-hour time | Valid | 14:30 |
| EC2 | Invalid format | Invalid | 25:00 |
| EC3 | Empty | Invalid | "" |

 **Test Cases**:

 **Weak ECP**: 3 test cases.

 **Strong ECP**: 9 test cases (combined with Content: 3 × 3).

**BVA**:

 Min: 00:00 (valid).

 Max: 23:59 (valid), 24:00 (invalid).

# Sprint 3: GUI Fields

1. **Create a New Post**

**Field: Post Title (Text Input)**

| **ECP Table**:  **Partition** | **Description** | **Valid/Invalid** | **Example** |
| --- | --- | --- | --- |
| EC1 | Valid title (1–100 chars) | Valid | "New Post" |
| EC2 | Too long (>100 chars) | Invalid | [101-char string] |
| EC3 | Empty | Invalid | "" |

**Test Cases**:

 **Weak ECP**: 3 test cases.

 **Strong ECP**: 3 test cases.

**BVA**:

 Min: 0 chars (invalid), 1 char (valid).

 Max: 100 chars (valid), 101 chars (invalid).

**Field: Post Content (Textarea)** (Same as Content Scheduling)  See above for ECP and BVA.

**Field: Media Upload (File Upload)**

| **ECP Table**:  **Partition** | **Description** | **Valid/Invalid** | **Example** |
| --- | --- | --- | --- |
| EC1 | Valid media (PNG/JPG, ≤5MB) | Valid | image.png (4MB) |
| EC2 | Invalid format (e.g., MP4) | Invalid | video.mp4 |
| EC3 | File too large (>5MB) | Invalid | image.png (6MB) |
| EC4 | No file | Valid | None |

 **Test Cases**:

 **Weak ECP**: 4 test cases.

 **Strong ECP**: 12 test cases (combined with Content: 3 × 4).

**BVA**:

 Min size: 0 bytes (valid, no file).

 Max size: 5MB (valid), 5MB+1byte (invalid).

1. **Like a Post**

**Field: Like Button (Button)**

| **ECP Table**:  **Partition** | **Description** | **Valid/Invalid** | **Example** |
| --- | --- | --- | --- |
| EC1 | Liked | Valid | Clicked (liked) |
| EC2 | Unliked | Valid | Clicked (unliked) |

 **Test Cases**:

 **Weak ECP**: 2 test cases.

 **Strong ECP**: 2 test cases.

 **BVA**: None.

1. **Comment on a Post**

**Field: Comment Text (Textarea)**

| **ECP Table**:  **Partition** | **Description** | **Valid/Invalid** | **Example** |
| --- | --- | --- | --- |
| EC1 | Valid comment (1–500 chars) | Valid | "Great post!" |
| EC2 | Too long (>500 chars) | Invalid | [501-char string] |
| EC3 | Empty | Invalid | "" |

 **Test Cases**:

 **Weak ECP**: 3 test cases.

 **Strong ECP**: 3 test cases.

**BVA**:

 Min: 0 chars (invalid), 1 char (valid).

 Max: 500 chars (valid), 501 chars (invalid).

1. **Send and Receive Messages Field: Message Text (Textarea)**

| **ECP Table**:  **Partition** | **Description** | **Valid/Invalid** | **Example** |
| --- | --- | --- | --- |
| EC1 | Valid message (1–1000 chars) | Valid | "Hi, love your content!" |
| EC2 | Too long (>1000 chars) | Invalid | [1001-char string] |
| EC3 | Empty | Invalid | "" |

 **Test Cases**:

 **Weak ECP**: 3 test cases.

 **Strong ECP**: 3 test cases.

**BVA**:

 Min: 0 chars (invalid), 1 char (valid).

 Max: 1000 chars (valid), 1001 chars (invalid).

**Field: Recipient ID/Email** (Same as Login Form Email)  See above for ECP and BVA.

# Notes

 **Non-Tested Fields**: User stories like View My Profile, View My Creator Dashboard, View My Payout

History, View Exclusive Content, and Browse the Feed are display-only (no input fields), so they have no ECP/BVA.

 **Consolidated Fields**: Fields like Email, Subscription Tier, and Content Text are reused across user stories to avoid redundancy, with sprint-specific notes where needed.

 **Test Case Counts**: Strong ECP counts increase when fields are interdependent (e.g., Password and Confirm Password in Signup). For single fields, weak and strong counts are often identical.

Katalon Studio: Use these ECP/BVA details to write test scripts in Katalon Studio, automating tests for each partition and boundary value (e.g., test login with valid/invalid email).

## **9. Product Review**

### **✅ Improvements Made:**

* Added Two-Factor Authentication for security.
* Introduced Dark Mode for accessibility.
* Enabled revenue tracking for creators through dashboard analytics.

### **⚠️ Limitations:**

* No in-app payment integration yet.
* No mobile application.
* Basic search/filtering for content discovery — could be enhanced.

### **💡 Future Enhancements:**

* Add payment gateway like Stripe.
* Integrate push notifications.
* Introduce role-based access control for admin/moderator functions.

**DEMO video**

https://drive.google.com/drive/folders/1\_PsFnWJrDQO34cQ2KZejIUGD8w2VCZ1x?usp=sharing **Presentation**https://www.canva.com/design/DAGmgqpg0E4/uzqDe\_lz9ZuEeIh\_xRGqnQ/edit?utm\_content=DAGmgqpg0E4&utm\_campaign=designshare&utm\_medium=link2&utm\_source=sharebutton