Postman Testing Guide — Instagram-style API

Base URL: http://127.0.0.1:8000/api/

Use header "X-Session-ID" with the session_id returned from login/signup for authenticated endpoints.

Step	Endpoint (path)	Method	Headers	Body (example)	Expected response / Notes
1	auth/signup/	POST	None	{ "username": "alice", "email":"alice@example.com", "password":"pass123", "full_name":"Alice" }	201 Created, JSON with "session_id" and user object. Save session_id for later.
2	auth/login/	POST	None	{ "username":"alice", "password":"pass123" }	200 OK, JSON with "session_id". Use this value in X-Session-ID header for subsequent requests.
3	auth/me/	GET	Header: X-Session-ID: <session_id>	None	200 OK, returns the authenticated user object. Validates session.
4	posts/ (create)	POST	Header: X-Session-ID	Form-data: caption="Hello", media: (file(s))	201 Created (or 202 Accepted if files). Response contains created post object with media_urls.
5	posts/feed/	GET	Header: X-Session-ID	None	200 OK, paginated list of posts from users you follow + your posts.
6	posts/user/{user_id}/	GET	Header: X-Session-ID	None	200 OK, paginated list of posts by user.
7	posts/{post_id}/ (delete)	DELETE	Header: X-Session-ID	None	204 No Content or 200 OK depending on implementation. Removes post and media cleanup triggered.
8	comments/ (create)	POST	Header: X-Session-ID	{ "post": "<post_id>", "text":"Nice photo!" }	201 Created, comment object (user set from session).
9	comments/?post_id={post_id}	GET	Header: X-Session-ID	None	200 OK, list top-level comments with replies for the post.
10	likes/{post_id}/toggle/	POST	Header: X-Session-ID	None	{ "liked": true } or { "liked": false } — toggles like.
11	likes/{post_id}/list_likes/	GET	Header: X-Session-ID	None	200 OK, list of users who liked the post.
12	followers/{user_id}/follow/	POST	Header: X-Session-ID	None	If target is public → follow immediately (200). If private → creates friend-request (pending).
13	followers/{user_id}/unfollow/	POST	Header: X-Session-ID	None	200 OK, unfollows user.
14	followers/{user_id}/followers/	GET	Header: X-Session-ID	None	200 OK, list of users following the target user (paginated).
15	followers/{user_id}/following/	GET	Header: X-Session-ID	None	200 OK, list of users the target user follows (paginated).
16	friend-requests/send/	POST	Header: X-Session-ID	{ "receiver": <user_id> }	201/200 OK, creates a friend (follow) request for private accounts. If existing pending/accepted, message returned.
17	friend-requests/{id}/accept/	POST	Header: X-Session-ID	None	200 OK, status accepted; creates follower relationship so sender follows receiver.
18	friend-requests/{id}/reject/	POST	Header: X-Session-ID	None	200 OK, status rejected.
19	friend-requests/pending/	GET	Header: X-Session-ID	None	200 OK, list of incoming pending requests for authenticated user (receiver).
20	friend-requests/sent/	GET	Header: X-Session-ID	None	200 OK, list of pending requests sent by the authenticated user.
21	friend-requests/friends/	GET	Header: X-Session-ID	None	200 OK, list of accepted friends (users with accepted requests).
22	messages/ (create)	POST	Header: X-Session-ID	Form-data: receiver=<user_id>, text="hi", media: (optional file)	201 Created, message object with media_url if uploaded.
23	messages/chat/{user_id}/	GET	Header: X-Session-ID	None	200 OK, list of messages between you and the given user ordered by created_at.
24	messages/{message_id}/mark_read/	POST	Header: X-Session-ID	None	200 OK, marks message as read (only receiver allowed).
25	stories/ (create)	POST	Header: X-Session-ID	Form-data: media (file), optional media_type	201 Created, story object with expires_at ~24h from creation.
26	stories/list_active/	GET	Header: X-Session-ID	None	200 OK, list active stories from users you follow and yourself.
27	stories/{id}/mark_viewed/	POST	Header: X-Session-ID	None	200 OK, marks story viewed for current user.
28	stories/{id}/viewers/	GET	Header: X-Session-ID	None	200 OK, returns list of viewers (only owner allowed).
29	stories/{id}/delete_story/	DELETE	Header: X-Session-ID	None	200 OK, deletes story (owner only).
30	search/users/?q={query}	GET	Header: optional X-Session-ID	None	200 OK, up to 20 user results matching username or full_name (min 2 chars).
31	auth/logout/	POST	Header: X-Session-ID	None	200 OK, deletes session on server; subsequent requests with same session_id will be unauthorized.

Testing Tips & Order of Operations

1. 1. Always start with signup -> login -> me to get a valid X-Session-ID.
2. 2. Use the same session_id header for authenticated requests: X-Session-ID: <session_id>.
3. 3. Test public vs private account flows: set profile.is_private True to test friend-request flow.
4. 4. For file uploads use multipart/form-data in Postman (key type: File).
5. 5. When accepting a friend request the backend creates a Follower entry so the sender will be following the receiver.
6. 6. Use the 'sent' and 'pending' endpoints to debug request state before/after accept/reject.
7. 7. If an endpoint requires pagination, check 'results' and use query params ?page=2.
8. 8. After logout, me and other authenticated endpoints should return 401.