**🔌 ReAlign API Routes Specification**

**Version:** 1.6

**Date:** May 21, 2025

**Stack:** Express.js + Drizzle ORM + Supabase (Auth)

**Base URL:** /api/v1

**General Conventions**

* **Authentication:** All protected routes expect a JWT in the Authorization header: Authorization: Bearer <YOUR\_JWT\_TOKEN>.
* **Request Body Format:** JSON (Content-Type: application/json).
* **Response Body Format:** JSON.
* **Date Format:** ISO 8601 format (e.g., 2025-05-21T14:30:00.000Z).
* **Pagination:** For list endpoints, pagination is supported via query parameters:
  + page (integer, default: 1): The page number to retrieve.
  + limit (integer, default: 20, max: 100): The number of items per page.

Paginated responses will include a pagination object:  
 JSON  
{

"data": [...],

"pagination": {

"currentPage": 1,

"totalPages": 5,

"totalItems": 98,

"perPage": 20

}

}

**Standardized Error Response:** Errors will return a JSON object with the following structure:  
 JSON  
{

"error": {

"code": "ERROR\_CODE\_STRING", // e.g., "VALIDATION\_ERROR", "UNAUTHENTICATED"

"message": "A descriptive error message.",

"details": [ // Optional: array of specific field errors for validation

{ "field": "fieldName", "message": "Specific issue" }

]

}

}

* **Common HTTP Status Codes:**
  + 200 OK: Request succeeded.
  + 201 Created: Resource successfully created.
  + 204 No Content: Request succeeded, but no data to return (e.g., after a DELETE).
  + 400 Bad Request: The request was malformed (e.g., validation error, missing parameters). Check error.details.
  + 401 Unauthorized: Authentication failed or JWT is missing/invalid.
  + 403 Forbidden: Authenticated user does not have permission to access the resource.
  + 404 Not Found: The requested resource does not exist.
  + 429 Too Many Requests: Rate limit exceeded.
  + 500 Internal Server Error: An unexpected error occurred on the server.

**🔐 Authentication (/auth)**

* **POST /auth/login**
  + Description: Handles email/password login for negotiators.

Request Body:  
 JSON  
{

"email": "negotiator@example.com",

"password": "securepassword123"

}

Response (200 OK):  
 JSON  
{

"user": {

"id": "uuid",

"email": "negotiator@example.com",

"role": "negotiator",

"name": "Negotiator Nick"

},

"token": "jwt\_token\_string"

}

* + Error Responses: 400 (Validation), 401 (Invalid credentials).
* **POST /auth/magic-link**
  + Description: Sends a one-time email or SMS login link for invited users (homeowner, listingAgent, buyersAgent, escrow, buyer). Role is assigned during transaction creation.

Request Body:  
 JSON  
{

"email": "user@example.com" // or "phone": "+15551234567"

}

Response (200 OK):  
 JSON  
{

"message": "Magic link sent successfully. Please check your email/SMS."

}

* + Error Responses: 400 (Validation, User not found or not invited), 429 (Rate limit).
* **POST /auth/magic-link/resend**
  + Description: Resends a magic link if the previous token expired or was invalid (max 3 per hour).

Request Body:  
 JSON  
{

"email": "user@example.com" // or "phone": "+15551234567"

}

Response (200 OK):  
 JSON  
{

"message": "Magic link resent successfully."

}

* + Error Responses: 400 (Validation), 429 (Rate limit exceeded ).
* **GET /auth/me**
  + Description: Returns current authenticated user’s ID, name, email, and role. Requires JWT.

Response (200 OK):  
 JSON  
{

"id": "uuid",

"name": "User Name",

"email": "user@example.com",

"role": "negotiator" // or other role

}

* + Error Responses: 401 (Unauthenticated).

**📁 Transactions (/transactions)**

* **POST /transactions**
  + Description: Creates a new transaction (negotiator only). Requires JWT.

Request Body:  
 JSON  
{

"title": "123 Main St Short Sale", // string, required

"property\_address": "123 Main St, Anytown, USA", // string, required

"parties": [ // array of party objects, required

{

"name": "Seller Sam", // string, required

"email": "seller@example.com", // string, required, unique within transaction parties

"phone": "+15551230001", // string, optional

"role": "seller" // string, required, enum: 'seller', 'buyer', 'listing\_agent', 'buyers\_agent', 'escrow'

}

// ...other parties

],

"initialPhase": "Transaction Initiated", // string, required, must match fixed list [cite: 238]

"initialMessage": "Welcome to the transaction for 123 Main St." // string, optional

}

* + Response (201 Created): Returns the full transaction details (similar to GET /transactions/:id).
  + Error Responses: 400 (Validation), 401 (Unauthenticated), 403 (Forbidden).
* **GET /transactions**
  + Description: Lists all transactions accessible to the authenticated user (negotiators see all their created transactions; other roles see transactions they are a party to). Requires JWT.
  + Query Parameters: page, limit.

Response (200 OK):  
 JSON  
{

"data": [

{

"id": "uuid\_transaction\_1",

"title": "123 Main St Short Sale",

"property\_address": "123 Main St, Anytown, USA",

"currentPhase": "Initial Document Collection",

"created\_by": "uuid\_negotiator", // User ID of the negotiator

"created\_at": "2025-05-20T10:00:00.000Z",

"lastActivityAt": "2025-05-21T14:00:00.000Z" // Calculated field

// Add completion % if readily available

}

// ...other transactions

],

"pagination": { ... }

}

* + Error Responses: 401 (Unauthenticated).
* **GET /transactions/:id**
  + Description: Fetches full transaction details including current phase, property address, associated users (parties), messages, document requests, and uploaded documents (filtered by user role). Requires JWT.

Response (200 OK):  
 JSON  
{

"id": "uuid\_transaction\_1",

"title": "123 Main St Short Sale",

"property\_address": "123 Main St, Anytown, USA",

"currentPhase": "Initial Document Collection",

"created\_by": { // Negotiator details

"id": "uuid\_negotiator",

"name": "Negotiator Nick"

},

"created\_at": "2025-05-20T10:00:00.000Z",

"parties": [ // From party\_status table, joined with users

{

"userId": "uuid\_user\_1",

"name": "Seller Sam",

"role": "seller",

"status": "pending", // 'pending', 'complete', 'overdue'

"lastAction": "Viewed update 2h ago" // text description

}

// ...other parties

],

"messages": [ /\* Paginated or limited list of recent messages - see GET /transactions/:id/messages \*/ ],

"documentRequests": [ /\* See document\_requests table structure \*/ ],

"uploads": [ /\* Filtered based on user role and upload visibility - see uploads table structure \*/ ]

}

* + Error Responses: 401 (Unauthenticated), 403 (Forbidden), 404 (Not Found).
* **PATCH /transactions/:id**
  + Description: Updates transaction metadata (title, current phase, property address) (negotiator only). Requires JWT.

Request Body (include only fields to update):  
 JSON  
{

"title": "New Title", // optional

"property\_address": "New Address", // optional

"currentPhase": "Lender Review" // optional, must match fixed list

}

* + Response (200 OK): Returns updated full transaction details.
  + Error Responses: 400 (Validation), 401, 403, 404.

**👥 Party Status (/transactions/:id/parties)**

* **GET /transactions/:id/parties**
  + Description: Lists all invited parties to a specific transaction, their roles, and current status. Requires JWT.

Response (200 OK):  
 JSON  
{

"data": [

{

"userId": "uuid\_user\_1",

"name": "Seller Sam",

"role": "seller",

"status": "pending", // 'pending', 'complete', 'overdue'

"lastAction": "Viewed update 2h ago", // text

"last\_updated": "2025-05-21T12:00:00.000Z"

}

// ...other parties

]

}

* + Error Responses: 401, 403, 404.
* **PATCH /transactions/:transactionId/parties/:userId**
  + Description: Updates a specific party’s status within a transaction (negotiator only). Requires JWT.

Request Body:  
 JSON  
{

"status": "complete", // required, enum: 'pending', 'complete', 'overdue'

"lastAction": "Documents reviewed by negotiator" // string, optional

}

* + Response (200 OK): Returns the updated party status object.
  + Error Responses: 400 (Validation), 401, 403, 404.

**💬 Messages (/transactions/:id/messages)**

* **GET /transactions/:id/messages**
  + Description: Retrieves the flat message feed for a transaction. Each message includes sender, text, timestamp, replyTo, isSeedMessage. Requires JWT.
  + Query Parameters: page, limit. (Sorted by created\_at descending by default).

Response (200 OK):  
 JSON  
{

"data": [

{

"id": "uuid\_message\_1",

"sender": { // sender details from users table

"id": "uuid\_user\_1",

"name": "Sender Name",

"role": "sender\_role"

},

"text": "Hello world!",

"replyTo": null, // or uuid\_parent\_message

"isSeedMessage": false, // boolean

"created\_at": "2025-05-21T10:00:00.000Z"

}

],

"pagination": { ... }

}

* + Error Responses: 401, 403, 404.
* **POST /transactions/:id/messages**
  + Description: Posts a new message to the transaction thread. Only negotiators may post new top-level messages; others can reply. Requires JWT.

Request Body:  
 JSON  
{

"text": "This is a new message.", // string, required

"replyTo": "uuid\_parent\_message\_id", // string, optional (null for new top-level message)

"isSeedMessage": false // boolean, optional, defaults to false (true only for initial transaction welcome message)

}

* + Response (201 Created): Returns the newly created message object.
  + Error Responses: 400 (Validation), 401, 403, 404.

**🗂️ Document Requests (/doc-requests, /transactions/:id/doc-requests)**

* **POST /transactions/:id/doc-requests**
  + Description: Creates a document request within a transaction (negotiator only). Requires JWT.
  + Note: docType must be from the hardcoded preset templates for MVP.

Request Body:  
 JSON  
{

"docType": "Paystubs (30 Days)", // string, required, from preset list

"assignedToUserId": "uuid\_user\_to\_assign", // string, required

"dueDate": "2025-06-01" // string YYYY-MM-DD, optional

}

* + Response (201 Created): Returns the newly created document request object.
  + Error Responses: 400 (Validation), 401, 403, 404.
* **GET /transactions/:id/doc-requests**
  + Description: Retrieves all document requests for a transaction. Requires JWT.
  + Query Parameters: page, limit.

Response (200 OK):  
 JSON  
{

"data": [

{

"id": "uuid\_doc\_request\_1",

"transactionId": "uuid\_transaction\_1",

"requestedBy": { "id": "uuid\_negotiator", "name": "Negotiator Nick"},

"assignedTo": { "id": "uuid\_user\_1", "name": "Seller Sam"},

"docType": "Paystubs (30 Days)",

"status": "requested", // 'requested', 'complete', 'pending', 'overdue'

"dueDate": "2025-06-01",

"revisionNote": "Please ensure all pages are included.", // string, optional

"created\_at": "2025-05-20T11:00:00.000Z",

"updated\_at": "2025-05-21T09:00:00.000Z"

}

],

"pagination": { ... }

}

* + Error Responses: 401, 403, 404.
* **PATCH /doc-requests/:requestId**
  + Description: Updates a document request (e.g., status, add revision note) (negotiator only). Requires JWT.

Request Body:  
 JSON  
{

"status": "pending", // optional, enum: 'requested', 'complete', 'pending', 'overdue'

"revisionNote": "Missing signature on page 2." // optional, string

// "dueDate": "2025-06-15" // optional

}

* + Response (200 OK): Returns the updated document request object.
  + Error Responses: 400 (Validation), 401, 403, 404.

**📤 Uploads (/uploads)**

* **POST /uploads/:transactionId**
  + Description: Upload endpoint for a file associated with a transaction. Requires JWT.
  + Request: multipart/form-data
    - file: The file to upload.
    - docType: string (e.g., "Paystubs", "Bank Statement", "Signed Offer").
    - visibility: string, enum: 'private', 'shared'.

Response (201 Created):  
 JSON  
{

"id": "uuid\_upload\_1",

"transactionId": "uuid\_transaction\_1",

"uploadedBy": { "id": "uuid\_uploader", "name": "Uploader Name"},

"fileUrl": "https://<supabase\_url>/storage/v1/object/public/uploads/file.pdf", // URL to access the file

"fileName": "file.pdf", // Original file name

"docType": "Paystubs",

"visibility": "private",

"contentType": "application/pdf", // MIME type

"size": 102400, // size in bytes

"created\_at": "2025-05-21T15:00:00.000Z"

}

* + Error Responses: 400 (Validation, File too large, Invalid file type), 401, 403, 404 (Transaction not found).
* **GET /uploads/:transactionId**
  + Description: Lists all file uploads for a transaction. Access is filtered by user role and upload visibility (negotiators see all; others see their own private uploads and all shared uploads). Requires JWT.
  + Query Parameters: page, limit.

Response (200 OK):  
 JSON  
{

"data": [

{

"id": "uuid\_upload\_1",

"uploadedBy": { "id": "uuid\_uploader", "name": "Uploader Name"},

"fileUrl": "...",

"fileName": "file.pdf",

"docType": "Paystubs",

"visibility": "private",

"contentType": "application/pdf",

"size": 102400,

"created\_at": "2025-05-21T15:00:00.000Z"

}

],

"pagination": { ... }

}

* + Error Responses: 401, 403, 404.