BookShop API Documentation

2025-01-30

Bookshop API

Overview

A comprehensive RESTful API for an academic book auction platform.
 This API enables students to buy and sell textbooks through an auction-based system, with features including real-time bidding, secure authentication, messaging between users, and moderation capabilities.

 Key Features:
 - Student and moderator authentication
 - Book auction management
 - Real-time bidding system
 - Public and private messaging
 - Advanced search and filtering
 - Auction monitoring and statistics

 For detailed authentication requirements and rate limits, please refer to the individual endpoint documentation.

Version

1.0.0

POST /v1/users

Create a new user.

Create a new user with the given information.

Request Body:

Content: application/json | UserCreation

All of:

User

```
{
    givenName: string;  // The given name of the user.
    familyName: string;  // The family name of the user.
    username: string;  // The username of the user.
    email: string;  // The email of the user.
}
```

and

```
{
    password: string;  // The password of the user.
}
```

Response 201:

Success.

```
Content: application/json | {
    user: Profile;
    token: AuthToken;
}
```

Response 400:

Error.

```
Content: application/json | {
    message: string;  // The error message.
    status: number;  // The error status.
    traceId: string;  // The trace ID.
}
```

Response 401:

Error.

Content: application/json | {
  message: string; // The error message.
  status: number; // The error status.
  traceId: string; // The trace ID.
}

Response 403:

Error.

Content: application/json | {
  message: string; // The error message.
  status: number; // The error status.
  traceId: string; // The trace ID.
}

Response 404:

Error.

Content: application/json | {
  message: string; // The error message.
  status: number; // The error status.
  traceId: string; // The trace ID.
}

Response 500:

Error.

Content: application/json | {
  message: string; // The error message.
  status: number; // The error status.
  traceId: string; // The trace ID.
}

GET /v1/users

List users.

List all users, must be an admin
Available filters: user.givenName, user.familyName, user.username, user.email.

Request Parameters:

page: integer;
pageSize: integer;
user.givenName?: LhsApiQueryOption;
user.familyName?: LhsApiQueryOption;
user.username?: LhsApiQueryOption;
user.email?: LhsApiQueryOption;

Response 200:

Success.

Content: application/json | {
  list: Array<Profile>; // The list of items.

```
    metadata: object;
}
```

Response 400:

Error.

```
Content: application/json | {
    message: string;  // The error message.
    status: number;  // The error status.
    traceId: string;  // The trace ID.
}
```

Response 401:

Error.

```
Content: application/json | {
    message: string;  // The error message.
    status: number;  // The error status.
    traceId: string;  // The trace ID.
}
```

Response 403:

Error.

```
Content: application/json | {
    message: string;  // The error message.
    status: number;  // The error status.
    traceId: string;  // The trace ID.
}
```

Response 404:

Error.

```
Content: application/json | {
    message: string;  // The error message.
    status: number;  // The error status.
    traceId: string;  // The trace ID.
}
```

Response 500:

Error.

```
Content: application/json | {
    message: string;  // The error message.
    status: number;  // The error status.
    traceId: string;  // The trace ID.
}
```

GET /v1/users/me

Get current user information.

Get the information of the current user.

Response 200:

Success.

Content: application/json | Profile

```
{
    publicId: PublicId;
    kind: string;  // The kind of the profile.
    user: User;
    properties: ProfileProperties;
    createdAt: string;  // The creation date.
    updatedAt: string;  // The last update date.
}
```

Response 400:

Error.

Content: application/json | {

```
    message: string;  // The error message.
    status: number;  // The error status.
    traceId: string;  // The trace ID.
}
```

Response 401:

Error.

Content: application/json | {

```
    message: string;  // The error message.
    status: number;  // The error status.
    traceId: string;  // The trace ID.
}
```

Response 403:

Error.

Content: application/json | {

```
    message: string;  // The error message.
    status: number;  // The error status.
    traceId: string;  // The trace ID.
}
```

Response 404:

Error.

Content: application/json | {

```
    message: string;  // The error message.
    status: number;  // The error status.
    traceId: string;  // The trace ID.
}
```

Response 500:

Error.

Content: application/json | {

```
    message: string;  // The error message.
    status: number;  // The error status.
    traceId: string;  // The trace ID.
}
```

PATCH /v1/users/me

Patch current user information.

Patch the information of the current user
Available patch fields: user/givenName, user/familyName, user/username, user/email, user/password.

Request Body:

Content: application/json | JsonPatchList

```
{
    list: Array<JsonPatch>;
}
```

Response 204:

Success.

Response 400:

Error.

```
Content: application/json | {
    message: string;  // The error message.
    status: number;  // The error status.
    traceId: string;  // The trace ID.
}
```

Response 401:

Error.

```
Content: application/json | {
    message: string;  // The error message.
    status: number;  // The error status.
    traceId: string;  // The trace ID.
}
```

Response 403:

Error.

```
Content: application/json | {
    message: string;  // The error message.
    status: number;  // The error status.
    traceId: string;  // The trace ID.
}
```

Response 404:

Error.

```
Content: application/json | {
    message: string;  // The error message.
    status: number;  // The error status.
    traceId: string;  // The trace ID.
}
```

Response 500:

Error.

```
Content: application/json | {
    message: string;  // The error message.
```

```
    status: number;  // The error status.
    traceId: string;  // The trace ID.
}
```

POST /v1/users/status

Ban users.

Ban a list of users, must be an admin.

Request Body:

```
Content: application/json | {
    action: string;
    publicIds: Array<PublicId>;
}
```

Response 204:

Success.

Response 400:

Error.

```
Content: application/json | {
    message: string;  // The error message.
    status: number;  // The error status.
    traceId: string;  // The trace ID.
}
```

Response 401:

Error.

```
Content: application/json | {
    message: string;  // The error message.
    status: number;  // The error status.
    traceId: string;  // The trace ID.
}
```

Response 403:

Error.

```
Content: application/json | {
    message: string;  // The error message.
    status: number;  // The error status.
    traceId: string;  // The trace ID.
}
```

Response 404:

Error.

```
Content: application/json | {
    message: string;  // The error message.
    status: number;  // The error status.
    traceId: string;  // The trace ID.
}
```

Response 500:

Error.

Content: application/json | {
  message: string; // The error message.
  status: number; // The error status.
  traceId: string; // The trace ID.
}

POST /v1/auth/token

Create a new token.

Create a new token for a user.

Request Body:

Content: application/json | LoginType

{
  username: string; // The username of the user.
  password: string; // The password of the user.
  scope: string; // The scope of the login.
}

Response 201:

Success.

Content: application/json | AuthToken

All of:

BaseToken
{
  token: string; // The token.
  expiresAt: number; // The expiration date of the token.
}

and

{
  type: string; // The type of the token.
}

Response 400:

Error.

Content: application/json | {
  message: string; // The error message.
  status: number; // The error status.
  traceId: string; // The trace ID.
}

Response 401:

Error.

Content: application/json | {
  message: string; // The error message.
  status: number; // The error status.
  traceId: string; // The trace ID.
}

Response 403:

Error.

Content: application/json | {
  message: string; // The error message.
  status: number; // The error status.
  traceId: string; // The trace ID.
}

Response 404:

Error.

Content: application/json | {
  message: string; // The error message.
  status: number; // The error status.
  traceId: string; // The trace ID.
}

Response 500:

Error.

Content: application/json | {
  message: string; // The error message.
  status: number; // The error status.
  traceId: string; // The trace ID.
}

DELETE /v1/auth/token

Delete a token.

Delete a token for a user.

Response 204:

Success.

Response 400:

Error.

Content: application/json | {
  message: string; // The error message.
  status: number; // The error status.
  traceId: string; // The trace ID.
}

Response 401:

Error.

Content: application/json | {
  message: string; // The error message.
  status: number; // The error status.
  traceId: string; // The trace ID.
}

Response 403:

Error.

Content: application/json | {
  message: string; // The error message.
  status: number; // The error status.
  traceId: string; // The trace ID.
}

Response 404:

Error.

Content: application/json | {
  message: string; // The error message.
  status: number; // The error status.
  traceId: string; // The trace ID.
}

Response 500:

Error.

Content: application/json | {
  message: string; // The error message.
  status: number; // The error status.
  traceId: string; // The trace ID.
}

POST /v1/auth/token/refresh

Refresh a token.

Refresh a token for a user.

Response 201:

Success.

Content: application/json | AuthToken

All of:

BaseToken
{
  token: string; // The token.
  expiresAt: number; // The expiration date of the token.
}

and

{
  type: string; // The type of the token.
}

Response 400:

Error.

Content: application/json | {
  message: string; // The error message.
  status: number; // The error status.
  traceId: string; // The trace ID.
}

Response 401:

Error.

Content: application/json | {
  message: string; // The error message.
  status: number; // The error status.
  traceId: string; // The trace ID.
}

Response 403:

Error.

Content: application/json | {
  message: string; // The error message.
  status: number; // The error status.
  traceId: string; // The trace ID.
}

Response 404:

Error.

Content: application/json | {
  message: string; // The error message.
  status: number; // The error status.
  traceId: string; // The trace ID.
}

Response 500:

Error.

Content: application/json | {
  message: string; // The error message.
  status: number; // The error status.
  traceId: string; // The trace ID.
}

POST /v1/listings

Create a listing.

Create a listing with main and gallery images.

Request Body:

Content: multipart/form-data | {
  body: ListingCreation;
  main: string; // The file.
  gallery: Array<string>; // The files.
}

Response 201:

Success.

Content: application/json | Listing

{
  publicId: PublicId;
  mainMedia: Media;
  gallery: Array<undefined>; // The gallery of the listing.
  book: Book;

```
    auction: Auction;
    createdAt: string;  // The creation date.
    updatedAt: string;  // The last update date.
}
```

Response 400:

Error.

```
Content: application/json | {
    message: string;  // The error message.
    status: number;  // The error status.
    traceId: string;  // The trace ID.
}
```

Response 401:

Error.

```
Content: application/json | {
    message: string;  // The error message.
    status: number;  // The error status.
    traceId: string;  // The trace ID.
}
```

Response 403:

Error.

```
Content: application/json | {
    message: string;  // The error message.
    status: number;  // The error status.
    traceId: string;  // The trace ID.
}
```

Response 404:

Error.

```
Content: application/json | {
    message: string;  // The error message.
    status: number;  // The error status.
    traceId: string;  // The trace ID.
}
```

Response 500:

Error.

```
Content: application/json | {
    message: string;  // The error message.
    status: number;  // The error status.
    traceId: string;  // The trace ID.
}
```

GET /v1/listings

Get all listings.

Get all listings with pagination
Available filters: book.university, book.course, auction.startingPrice, publicId.

Request Parameters:

page: integer;
pageSize: integer;
search?: string;
sort?: undefined;
book.university?: LhsApiQueryOption;
book.course?: LhsApiQueryOption;
auction.startingPrice?: LhsApiQueryOption;
publicId?: LhsApiQueryOption;

Response 200:

Success.

Content: application/json | {
  list: Array<Listing>; // The list of items.
  metadata: object;
}

Response 400:

Error.

Content: application/json | {
  message: string; // The error message.
  status: number; // The error status.
  traceId: string; // The trace ID.
}

Response 401:

Error.

Content: application/json | {
  message: string; // The error message.
  status: number; // The error status.
  traceId: string; // The trace ID.
}

Response 403:

Error.

Content: application/json | {
  message: string; // The error message.
  status: number; // The error status.
  traceId: string; // The trace ID.
}

Response 404:

Error.

Content: application/json | {
  message: string; // The error message.
  status: number; // The error status.
  traceId: string; // The trace ID.
}

Response 500:

Error.

Content: application/json | {
  message: string; // The error message.
  status: number; // The error status.
  traceId: string; // The trace ID.
}

GET /v1/listings/filter-metadata

Get listings filter metadata.

Get listings filter metadata.

Response 200:

Success.

Content: application/json | ListingFilterMetadata

{
  universities: Array<string>; // The universities of the listings.
  courses: Array<string>; // The courses of the listings.
  geographicalAreas: Array<string>; // The geographical areas of the listings.
  startingPrice: object; // The starting price range of the listings.
}

Response 400:

Error.

Content: application/json | {
  message: string; // The error message.
  status: number; // The error status.
  traceId: string; // The trace ID.
}

Response 401:

Error.

Content: application/json | {
  message: string; // The error message.
  status: number; // The error status.
  traceId: string; // The trace ID.
}

Response 403:

Error.

Content: application/json | {
  message: string; // The error message.
  status: number; // The error status.
  traceId: string; // The trace ID.
}

Response 404:

Error.

Content: application/json | {
  message: string; // The error message.
  status: number; // The error status.
  traceId: string; // The trace ID.

}

Response 500:

Error.

Content: application/json | {
  message: string; // The error message.
  status: number; // The error status.
  traceId: string; // The trace ID.
}

GET /v1/listings/keywords

Get listing keywords.

Get paginated listing keywords by search query.

Request Parameters:

search: string;

Response 200:

Success.

Content: application/json | Array<string>

Response 400:

Error.

Content: application/json | {
  message: string; // The error message.
  status: number; // The error status.
  traceId: string; // The trace ID.
}

Response 401:

Error.

Content: application/json | {
  message: string; // The error message.
  status: number; // The error status.
  traceId: string; // The trace ID.
}

Response 403:

Error.

Content: application/json | {
  message: string; // The error message.
  status: number; // The error status.
  traceId: string; // The trace ID.
}

Response 404:

Error.

Content: application/json | {

```
    message: string;  // The error message.
    status: number;  // The error status.
    traceId: string;  // The trace ID.
}
```

Response 500:

Error.

```
Content: application/json | {
    message: string;  // The error message.
    status: number;  // The error status.
    traceId: string;  // The trace ID.
}
```

GET /v1/listings/history/won

Get listing history of a user won.

Get listing history of a user won, must be authenticated as a user.

Request Parameters:

```
page: integer;
pageSize: integer;
```

Response 200:

Success.

```
Content: application/json | {
    list: Array<Listing>;  // The list of items.
    metadata: object;
}
```

Response 400:

Error.

```
Content: application/json | {
    message: string;  // The error message.
    status: number;  // The error status.
    traceId: string;  // The trace ID.
}
```

Response 401:

Error.

```
Content: application/json | {
    message: string;  // The error message.
    status: number;  // The error status.
    traceId: string;  // The trace ID.
}
```

Response 403:

Error.

```
Content: application/json | {
    message: string;  // The error message.
    status: number;  // The error status.
```

```
    traceId: string;  // The trace ID.
}
```

Response 404:

Error.

```
Content: application/json | {
    message: string;  // The error message.
    status: number;  // The error status.
    traceId: string;  // The trace ID.
}
```

Response 500:

Error.

```
Content: application/json | {
    message: string;  // The error message.
    status: number;  // The error status.
    traceId: string;  // The trace ID.
}
```

GET /v1/listings/history/participated

Get listing history of a user participated in.

Get listing history of a user participated in, must be authenticated as a user.

Request Parameters:

```
page: integer;
pageSize: integer;
```

Response 200:

Success.

```
Content: application/json | {
    list: Array<Listing>;  // The list of items.
    metadata: object;
}
```

Response 400:

Error.

```
Content: application/json | {
    message: string;  // The error message.
    status: number;  // The error status.
    traceId: string;  // The trace ID.
}
```

Response 401:

Error.

```
Content: application/json | {
    message: string;  // The error message.
    status: number;  // The error status.
    traceId: string;  // The trace ID.
}
```

Response 403:

Error.

Content: application/json | {
  message: string; // The error message.
  status: number; // The error status.
  traceId: string; // The trace ID.
}

Response 404:

Error.

Content: application/json | {
  message: string; // The error message.
  status: number; // The error status.
  traceId: string; // The trace ID.
}

Response 500:

Error.

Content: application/json | {
  message: string; // The error message.
  status: number; // The error status.
  traceId: string; // The trace ID.
}

GET /v1/listings/owned

Get listings owned by a user.

Get listings owned by a user, must be authenticated as a user.

Request Parameters:

page: integer;
pageSize: integer;

Response 200:

Success.

Content: application/json | {
  list: Array<Listing>; // The list of items.
  metadata: object;
}

Response 400:

Error.

Content: application/json | {
  message: string; // The error message.
  status: number; // The error status.
  traceId: string; // The trace ID.
}

Response 401:

Error.

Content: application/json | {
  message: string; // The error message.
  status: number; // The error status.
  traceId: string; // The trace ID.
}

Response 403:

Error.

Content: application/json | {
  message: string; // The error message.
  status: number; // The error status.
  traceId: string; // The trace ID.
}

Response 404:

Error.

Content: application/json | {
  message: string; // The error message.
  status: number; // The error status.
  traceId: string; // The trace ID.
}

Response 500:

Error.

Content: application/json | {
  message: string; // The error message.
  status: number; // The error status.
  traceId: string; // The trace ID.
}

GET /v1/listings/{publicId}

Get a listing.

Get a listing by its public ID.

Request Parameters:

publicId: string;

Response 200:

Success.

Content: application/json | Listing

{
  publicId: PublicId;
  mainMedia: Media;
  gallery: Array<undefined>; // The gallery of the listing.
  book: Book;
  auction: Auction;
  createdAt: string; // The creation date.
  updatedAt: string; // The last update date.
}

Response 400:

Error.

Content: application/json | {
  message: string; // The error message.
  status: number; // The error status.
  traceId: string; // The trace ID.
}

Response 401:

Error.

Content: application/json | {
  message: string; // The error message.
  status: number; // The error status.
  traceId: string; // The trace ID.
}

Response 403:

Error.

Content: application/json | {
  message: string; // The error message.
  status: number; // The error status.
  traceId: string; // The trace ID.
}

Response 404:

Error.

Content: application/json | {
  message: string; // The error message.
  status: number; // The error status.
  traceId: string; // The trace ID.
}

Response 500:

Error.

Content: application/json | {
  message: string; // The error message.
  status: number; // The error status.
  traceId: string; // The trace ID.
}

DELETE /v1/listings/{publicId}

Delete a listing.

Delete a listing by its public ID.

Request Parameters:

publicId: string;

Response 204:

Success.

Response 400:

Error.

Content: application/json | {
  message: string; // The error message.
  status: number; // The error status.
  traceId: string; // The trace ID.
}

Response 401:

Error.

Content: application/json | {
  message: string; // The error message.
  status: number; // The error status.
  traceId: string; // The trace ID.
}

Response 403:

Error.

Content: application/json | {
  message: string; // The error message.
  status: number; // The error status.
  traceId: string; // The trace ID.
}

Response 404:

Error.

Content: application/json | {
  message: string; // The error message.
  status: number; // The error status.
  traceId: string; // The trace ID.
}

Response 500:

Error.

Content: application/json | {
  message: string; // The error message.
  status: number; // The error status.
  traceId: string; // The trace ID.
}

PATCH /v1/listings/{publicId}

Patch a listing.

Patch a listing by its public ID
 Available patch fields: book/title, book/author, book/geographicalArea, book/isbn, book/university, book/course, auction/startingPrice, auction/endDate.

Request Parameters:

publicId: string;

Request Body:

Content: application/json | JsonPatchList

```
{
    list: Array<JsonPatch>;
}
```

Response 204:

Success.

Response 400:

Error.

```
Content: application/json | {
    message: string;  // The error message.
    status: number;  // The error status.
    traceId: string;  // The trace ID.
}
```

Response 401:

Error.

```
Content: application/json | {
    message: string;  // The error message.
    status: number;  // The error status.
    traceId: string;  // The trace ID.
}
```

Response 403:

Error.

```
Content: application/json | {
    message: string;  // The error message.
    status: number;  // The error status.
    traceId: string;  // The trace ID.
}
```

Response 404:

Error.

```
Content: application/json | {
    message: string;  // The error message.
    status: number;  // The error status.
    traceId: string;  // The trace ID.
}
```

Response 500:

Error.

```
Content: application/json | {
    message: string;  // The error message.
    status: number;  // The error status.
    traceId: string;  // The trace ID.
}
```

GET /v1/listings/{listingId}/chats

Get all chats for a listing.

Get all chats for a listing, the authentication is required for private chats.

Request Parameters:

listingId: string;
page: integer;
pageSize: integer;

Response 200:

Success.

Content: application/json | {
  list: Array<Chat>; // The list of items.
  metadata: object;
}

Response 400:

Error.

Content: application/json | {
  message: string; // The error message.
  status: number; // The error status.
  traceId: string; // The trace ID.
}

Response 401:

Error.

Content: application/json | {
  message: string; // The error message.
  status: number; // The error status.
  traceId: string; // The trace ID.
}

Response 403:

Error.

Content: application/json | {
  message: string; // The error message.
  status: number; // The error status.
  traceId: string; // The trace ID.
}

Response 404:

Error.

Content: application/json | {
  message: string; // The error message.
  status: number; // The error status.
  traceId: string; // The trace ID.
}

Response 500:

Error.

Content: application/json | {
  message: string; // The error message.
  status: number; // The error status.
  traceId: string; // The trace ID.
}

POST /v1/listings/{listingId}/chats

Create a chat.

Create a chat.

Request Parameters:

listingId: string;

Request Body:

Content: application/json | MessageCreation

{
  content: string; // The content of the message.
}

Response 201:

Success.

Content: application/json | Chat

{
  publicId: PublicId;
  isPublic: boolean; // Whether the chat is public.
  buyer?: object; // The buyer of the chat.
  lastMessage?: Message;
}

Response 400:

Error.

Content: application/json | {
  message: string; // The error message.
  status: number; // The error status.
  traceId: string; // The trace ID.
}

Response 401:

Error.

Content: application/json | {
  message: string; // The error message.
  status: number; // The error status.
  traceId: string; // The trace ID.
}

Response 403:

Error.

Content: application/json | {
  message: string; // The error message.

```
    status: number;  // The error status.
    traceId: string;  // The trace ID.
}
```

Response 404:

Error.

```
Content: application/json | {
    message: string;  // The error message.
    status: number;  // The error status.
    traceId: string;  // The trace ID.
}
```

Response 500:

Error.

```
Content: application/json | {
    message: string;  // The error message.
    status: number;  // The error status.
    traceId: string;  // The trace ID.
}
```

GET /v1/chats/{chatId}/messages

Get all messages for a chat.

Get all messages for a chat, the authentication is required for private chats.

Request Parameters:

```
chatId: string;
page: integer;
pageSize: integer;
```

Response 200:

Success.

```
Content: application/json | {
    list: Array<undefined>;  // The list of items.
    metadata: object;
}
```

Response 400:

Error.

```
Content: application/json | {
    message: string;  // The error message.
    status: number;  // The error status.
    traceId: string;  // The trace ID.
}
```

Response 401:

Error.

```
Content: application/json | {
    message: string;  // The error message.
    status: number;  // The error status.
```

```
    traceId: string;  // The trace ID.
}
```

Response 403:

Error.

```
Content: application/json | {
    message: string;  // The error message.
    status: number;  // The error status.
    traceId: string;  // The trace ID.
}
```

Response 404:

Error.

```
Content: application/json | {
    message: string;  // The error message.
    status: number;  // The error status.
    traceId: string;  // The trace ID.
}
```

Response 500:

Error.

```
Content: application/json | {
    message: string;  // The error message.
    status: number;  // The error status.
    traceId: string;  // The trace ID.
}
```

GET /v1/watchlist

Retrieve the watchlist for the current user.

Retrieve the watchlist for the current user.

Request Parameters:

```
page: integer;
pageSize: integer;
listingGuids?: undefined;
```

Response 200:

Success.

```
Content: application/json | {
    list: Array<Listing>;  // The list of items.
    metadata: object;
}
```

Response 400:

Error.

```
Content: application/json | {
    message: string;  // The error message.
    status: number;  // The error status.
    traceId: string;  // The trace ID.
```

```
}
```

Response 401:

Error.

```
Content: application/json | {
    message: string;  // The error message.
    status: number;  // The error status.
    traceId: string;  // The trace ID.
}
```

Response 403:

Error.

```
Content: application/json | {
    message: string;  // The error message.
    status: number;  // The error status.
    traceId: string;  // The trace ID.
}
```

Response 404:

Error.

```
Content: application/json | {
    message: string;  // The error message.
    status: number;  // The error status.
    traceId: string;  // The trace ID.
}
```

Response 500:

Error.

```
Content: application/json | {
    message: string;  // The error message.
    status: number;  // The error status.
    traceId: string;  // The trace ID.
}
```

POST /v1/watchlist

Add a listing to watchlist.

Add a listing to watchlist.

Request Body:

```
Content: application/json | {
    listings: Array<string>;
}
```

Response 201:

Success.

```
Content: application/json |
```

Response 400:

Error.

Content: application/json | {
  message: string; // The error message.
  status: number; // The error status.
  traceId: string; // The trace ID.
}

Response 401:

Error.

Content: application/json | {
  message: string; // The error message.
  status: number; // The error status.
  traceId: string; // The trace ID.
}

Response 403:

Error.

Content: application/json | {
  message: string; // The error message.
  status: number; // The error status.
  traceId: string; // The trace ID.
}

Response 404:

Error.

Content: application/json | {
  message: string; // The error message.
  status: number; // The error status.
  traceId: string; // The trace ID.
}

Response 500:

Error.

Content: application/json | {
  message: string; // The error message.
  status: number; // The error status.
  traceId: string; // The trace ID.
}

DELETE /v1/watchlist/{listingId}

Remove a listing from watchlist.

Remove a listing from watchlist.

Request Parameters:

listingId: string;

Response 204:

Success.

Response 400:

Error.

Content: application/json | {
  message: string; // The error message.
  status: number; // The error status.
  traceId: string; // The trace ID.
}

Response 401:

Error.

Content: application/json | {
  message: string; // The error message.
  status: number; // The error status.
  traceId: string; // The trace ID.
}

Response 403:

Error.

Content: application/json | {
  message: string; // The error message.
  status: number; // The error status.
  traceId: string; // The trace ID.
}

Response 404:

Error.

Content: application/json | {
  message: string; // The error message.
  status: number; // The error status.
  traceId: string; // The trace ID.
}

Response 500:

Error.

Content: application/json | {
  message: string; // The error message.
  status: number; // The error status.
  traceId: string; // The trace ID.
}

GET /v1/notifications

Get all notifications.

Get all notifications for the current user.

Request Parameters:

page: integer;
pageSize: integer;

Response 200:

Success.

Content: application/json | {
  list: Array<Notification>; // The list of items.
  metadata: object;
}

Response 400:

Error.

Content: application/json | {
  message: string; // The error message.
  status: number; // The error status.
  traceId: string; // The trace ID.
}

Response 401:

Error.

Content: application/json | {
  message: string; // The error message.
  status: number; // The error status.
  traceId: string; // The trace ID.
}

Response 403:

Error.

Content: application/json | {
  message: string; // The error message.
  status: number; // The error status.
  traceId: string; // The trace ID.
}

Response 404:

Error.

Content: application/json | {
  message: string; // The error message.
  status: number; // The error status.
  traceId: string; // The trace ID.
}

Response 500:

Error.

Content: application/json | {
  message: string; // The error message.
  status: number; // The error status.
  traceId: string; // The trace ID.
}

POST /v1/notifications/{publicId}/read

Mark a notification as read.

Mark a notification as read.

Request Parameters:

publicId: string;

Response 204:

Success.

Response 400:

Error.

Content: application/json | {
  message: string; // The error message.
  status: number; // The error status.
  traceId: string; // The trace ID.
}

Response 401:

Error.

Content: application/json | {
  message: string; // The error message.
  status: number; // The error status.
  traceId: string; // The trace ID.
}

Response 403:

Error.

Content: application/json | {
  message: string; // The error message.
  status: number; // The error status.
  traceId: string; // The trace ID.
}

Response 404:

Error.

Content: application/json | {
  message: string; // The error message.
  status: number; // The error status.
  traceId: string; // The trace ID.
}

Response 500:

Error.

Content: application/json | {
  message: string; // The error message.
  status: number; // The error status.
  traceId: string; // The trace ID.
}

POST /v1/listings/{listingId}/bids

Create a bid for a listing.

Create a bid for a listing.

Request Parameters:

listingId: string;

Request Body:

Content: application/json | BidCreation

```
{
    amount: number;  // The amount of the bid.
}
```

Response 201:

Success.

Content: application/json

All of:

Bid

```
{
    publicId: PublicId;
    bidder: object;  // The bidder of the bid.
    amount: number;  // The amount of the bid.
}
```

and

Response 400:

Error.

Content: application/json | {

```
    message: string;  // The error message.
    status: number;  // The error status.
    traceId: string;  // The trace ID.
}
```

Response 401:

Error.

Content: application/json | {

```
    message: string;  // The error message.
    status: number;  // The error status.
    traceId: string;  // The trace ID.
}
```

Response 403:

Error.

Content: application/json | {

```
    message: string;  // The error message.
    status: number;  // The error status.
    traceId: string;  // The trace ID.
}
```

Response 404:

Error.

Content: application/json | {

```
    message: string;  // The error message.
```

```
    status: number;  // The error status.
    traceId: string;  // The trace ID.
}
```

Response 500:

Error.

```
Content: application/json | {
    message: string;  // The error message.
    status: number;  // The error status.
    traceId: string;  // The trace ID.
}
```

GET /v1/listings/{listingId}/bids

Get bids for a listing.

Get bids for a listing.

Request Parameters:

```
listingId: string;
```

Response 200:

Success.

```
Content: application/json | Array<undefined>
```

Response 400:

Error.

```
Content: application/json | {
    message: string;  // The error message.
    status: number;  // The error status.
    traceId: string;  // The trace ID.
}
```

Response 401:

Error.

```
Content: application/json | {
    message: string;  // The error message.
    status: number;  // The error status.
    traceId: string;  // The trace ID.
}
```

Response 403:

Error.

```
Content: application/json | {
    message: string;  // The error message.
    status: number;  // The error status.
    traceId: string;  // The trace ID.
}
```

Response 404:

Error.

Content: application/json | {
  message: string; // The error message.
  status: number; // The error status.
  traceId: string; // The trace ID.
}

Response 500:

Error.

Content: application/json | {
  message: string; // The error message.
  status: number; // The error status.
  traceId: string; // The trace ID.
}

GET /v1/statistics/auctions/now

Get the current auction statistics.

Get the current auction statistics, user must be an admin.

Response 200:

Success.

Content: application/json | CurrentAuctionStatistics

{
  auctions: object; // The statistics of the auctions.
}

Response 400:

Error.

Content: application/json | {
  message: string; // The error message.
  status: number; // The error status.
  traceId: string; // The trace ID.
}

Response 401:

Error.

Content: application/json | {
  message: string; // The error message.
  status: number; // The error status.
  traceId: string; // The trace ID.
}

Response 403:

Error.

Content: application/json | {
  message: string; // The error message.
  status: number; // The error status.
  traceId: string; // The trace ID.
}

Response 404:

Error.

Content: application/json | {
  message: string; // The error message.
  status: number; // The error status.
  traceId: string; // The trace ID.
}

Response 500:

Error.

Content: application/json | {
  message: string; // The error message.
  status: number; // The error status.
  traceId: string; // The trace ID.
}

GET /v1/statistics/auctions/daily

Get the daily auction statistics.

Get the daily auction statistics, user must be an admin.

Request Parameters:

fromDate: string;
toDate: string;

Response 200:

Success.

Content: application/json | DailyAuctionStatistics

{
  daily: Array<object>;
}

Response 400:

Error.

Content: application/json | {
  message: string; // The error message.
  status: number; // The error status.
  traceId: string; // The trace ID.
}

Response 401:

Error.

Content: application/json | {
  message: string; // The error message.
  status: number; // The error status.
  traceId: string; // The trace ID.
}

Response 403:

Error.

Content: application/json | {
  message: string; // The error message.
  status: number; // The error status.
  traceId: string; // The trace ID.
}

Response 404:

Error.

Content: application/json | {
  message: string; // The error message.
  status: number; // The error status.
  traceId: string; // The trace ID.
}

Response 500:

Error.

Content: application/json | {
  message: string; // The error message.
  status: number; // The error status.
  traceId: string; // The trace ID.
}

GET /v1/statistics/active

Get the active statistics.

Get the active statistics, user must be an admin.

Response 200:

Success.

Content: application/json | {
  activeAuctions: number; // The number of active auctions.
  activeUsers: number; // The number of active users.
}

Response 400:

Error.

Content: application/json | {
  message: string; // The error message.
  status: number; // The error status.
  traceId: string; // The trace ID.
}

Response 401:

Error.

Content: application/json | {
  message: string; // The error message.
  status: number; // The error status.
  traceId: string; // The trace ID.
}

Response 403:

Error.

Content: application/json | {
  message: string; // The error message.
  status: number; // The error status.
  traceId: string; // The trace ID.
}

Response 404:

Error.

Content: application/json | {
  message: string; // The error message.
  status: number; // The error status.
  traceId: string; // The trace ID.
}

Response 500:

Error.

Content: application/json | {
  message: string; // The error message.
  status: number; // The error status.
  traceId: string; // The trace ID.
}

POST /v1/admins

Create a new admin user.

Create a new admin user with the given information, must be authenticated as an admin.

Request Body:

Content: application/json | UserCreation

All of:

User
{
  givenName: string; // The given name of the user.
  familyName: string; // The family name of the user.
  username: string; // The username of the user.
  email: string; // The email of the user.
}

and

{
  password: string; // The password of the user.
}

Response 201:

Success.

Content: application/json | {
  user: Profile;
}

Response 400:

Error.

Content: application/json | {
  message: string; // The error message.
  status: number; // The error status.
  traceId: string; // The trace ID.
}

Response 401:

Error.

Content: application/json | {
  message: string; // The error message.
  status: number; // The error status.
  traceId: string; // The trace ID.
}

Response 403:

Error.

Content: application/json | {
  message: string; // The error message.
  status: number; // The error status.
  traceId: string; // The trace ID.
}

Response 404:

Error.

Content: application/json | {
  message: string; // The error message.
  status: number; // The error status.
  traceId: string; // The trace ID.
}

Response 500:

Error.

Content: application/json | {
  message: string; // The error message.
  status: number; // The error status.
  traceId: string; // The trace ID.
}

Schemas

PublicId

string

User

```
{
    givenName: string;  // The given name of the user.
    familyName: string;  // The family name of the user.
    username: string;  // The username of the user.
    email: string;  // The email of the user.
}
```

ProfileProperties

```
{
    isFirstLogin: boolean;  // Whether the profile is the first login.
    isBanned: boolean;  // Whether the profile is banned.
}
```

Profile

```
{
    publicId: PublicId;
    kind: string;  // The kind of the profile.
    user: User;
    properties: ProfileProperties;
    createdAt: string;  // The creation date.
    updatedAt: string;  // The last update date.
}
```

BaseToken

```
{
    token: string;  // The token.
    expiresAt: number;  // The expiration date of the token.
}
```

AuthToken

All of:

BaseToken

```
{
    token: string;  // The token.
    expiresAt: number;  // The expiration date of the token.
}
```

and

```
{
    type: string;  // The type of the token.
}
```

UserCreation

All of:

```
User
{
    givenName: string;  // The given name of the user.
    familyName: string;  // The family name of the user.
    username: string;  // The username of the user.
    email: string;  // The email of the user.
}
```

and

```
{
    password: string;  // The password of the user.
}
```

JsonPatchOp

string
Values: add, remove, replace

JsonPatchPath

string

JsonPatchValue

Any of:

```
string
```

or

```
number
```

or

```
boolean
```

JsonPatch

```
{
    op: JsonPatchOp;
    path: JsonPatchPath;
    value: JsonPatchValue;
}
```

JsonPatchList

```
{
    list: Array<JsonPatch>;
}
```

LhsApiQueryOption

```
{
    eq: string;  // The field must be equal to the value.
    lt: string;  // The field must be less than the value.
    le: string;  // The field must be less than or equal to the value.
    gt: string;  // The field must be greater than the value.
    ge: string;  // The field must be greater than or equal to the value.
    ne: string;  // The field must not be equal to the value.
    in: string;  // The field must be in the list of values.
    nin: string;  // The field must not be in the list of values.
    m: string;  // The field must match the regex.
    mi: string;  // The field must match the regex, ignoring case.
}
```

LoginType

```
{
    username: string;  // The username of the user.
    password: string;  // The password of the user.
    scope: string;  // The scope of the login.
}
```

Media

```
{
    publicId: PublicId;
    fileName: string;  // The file name of the media.
    originalFileName: string;  // The original file name of the media.
    path: string;  // The path of the media.
    mimetype: string;  // The mimetype of the media.
    size: number;  // The size of the media.
    createdAt: string;  // The creation date.
    updatedAt: string;  // The last update date.
}
```

Book

```
{
    title: string;  // The title of the book.
    author: string;  // The author of the book.
    course: string;  // The course of the book.
    isbn: string;  // The ISBN of the book.
    university: string;  // The university of the book.
    geographicalArea: string;  // The geographical area of the book.
    condition?: string;  // The condition of the book.
    edition?: string;  // The edition of the book.
    language?: string;  // The language of the book.
    pages?: number;  // The number of pages of the book.
```

```
}
```

Bid

```
{
    publicId: PublicId;
    bidder: object;  // The bidder of the bid.
    amount: number;  // The amount of the bid.
}
```

Auction

```
{
    seller: object;  // The seller of the auction.
    endDate: string;  // The end date of the auction.
    startingPrice: number;  // The starting price of the auction.
    reservePrice: number;  // The reserve price of the auction.
    winningBid?: Bid;
}
```

Listing

```
{
    publicId: PublicId;
    mainMedia: Media;
    gallery: Array<undefined>;  // The gallery of the listing.
    book: Book;
    auction: Auction;
    createdAt: string;  // The creation date.
    updatedAt: string;  // The last update date.
}
```

AuctionCreation

```
{
    endDate: string;  // The end date of the auction.
    startingPrice: number;  // The starting price of the auction.
    reservePrice: number;  // The reserve price of the auction.
}
```

ListingCreation

```
{
    book: undefined;
    auction: AuctionCreation;
}
```

ListingFilterMetadata

```
{
```

```
    universities: Array<string>;  // The universities of the listings.
    courses: Array<string>;  // The courses of the listings.
    geographicalAreas: Array<string>;  // The geographical areas of the listings.
    startingPrice: object;  // The starting price range of the listings.
}
```

Message

```
{
    publicId: PublicId;
    sender: object;  // The sender of the message.
    content: string;  // The content of the message.
    chatId: string;  // The chat ID.
    createdAt: string;  // The creation date of the message.
}
```

Chat

```
{
    publicId: PublicId;
    isPublic: boolean;  // Whether the chat is public.
    buyer?: object;  // The buyer of the chat.
    lastMessage?: Message;
}
```

MessageCreation

```
{
    content: string;  // The content of the message.
}
```

Notification

```
{
    publicId: PublicId;
    type: string;  // The type of the notification.
    listingPublicId: undefined;
    readAt?: string;  // The date the notification was read.
    createdAt: string;  // The date the notification was created.
}
```

BidCreation

```
{
    amount: number;  // The amount of the bid.
}
```

CurrentAuctionStatistics

```
{
```

```
    auctions: object;  // The statistics of the auctions.
}
```

DailyAuctionStatistics

```
{
    daily: Array<object>;
}
```