AACE Court Watchdog technical guide

Overview

The court watchdog app is an HTTP web server that serves a REST API and a frontend application. The server is powered by the Express NodeJS framework and runs on all platforms where NodeJS is supported. The server connects to an external MongoDB database to persist its data.

The server hosts a number of web scrapers that can search Australian court databases for legal cases involving certain parties. The core feature of the app is the automatic searches that run in the background. These searches run twice daily and return cases up to 30 days in the past (where allowed by databases – see appendix A). Results from these searches are aggregated and stored in the MongoDB database, and an email is sent to users when new results are found. Some court databases only display cases for the current day, so it is important the server runs a search daily to find these cases.

Through the web app, users can also run interactive manual searches. The results from these searches are delivered straight to the web app in real time. They are not stored in the database and thus will not trigger email notifications nor appear in the application history. Unlike the automatic searches, manual searches run across the maximum date range of each court database.

Accompanying this guide is the source code of the application (available in the “source” folder), a pre-built distribution ready for deployment (available in the “dist” folder), and a series of short tutorial videos aimed at users (available in the “tutorials” folder, each 1-2min). There is also a file “auditors.json”, which contains an export of a MongoDB database that has been preloaded with the list of active auditors in Queensland at the time of writing. Instructions on importing this file are in the backup and restore section.

Deployment & management

Installation

Server

To install and launch the server, the NodeJS runtime is required (available from https://nodejs.org/en/, LTS edition recommended).

Once installed, the node package manager (npm) can be used to download all the required dependencies for the server. You can interact with npm from the command line.

Before continuing, ensure the following environment variable is set to indicate to NodeJS that you are only interested in the production dependencies. This will skip downloading certain dependencies that are only required during the development process. This environment variable should also be set whenever you launch the server, so that it runs in production mode.

```
NODE_ENV=production
```

In the distribution folder (called “dist” in the accompanying files), run

```
npm install
```

This will install all required dependencies for the server and save them in the “node_modules” folder.

Database

The server requires a MongoDB instance to persist its data. To self-host an instance, use the MongoDB community server (free, licensed under the SSPL) or the Enterprise server (paid). Alternatively, MongoDB Atlas can be used to easily create a remote MongoDB instance.

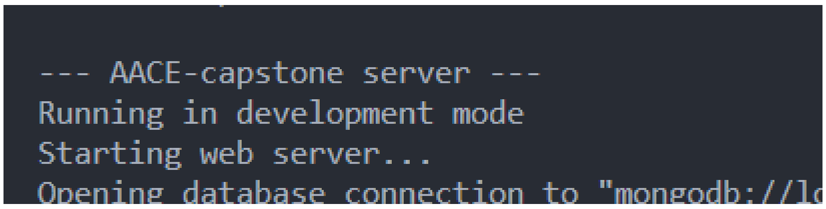
Instructions for installing a local MongoDB Community instance are available at https://www.mongodb.com/docs/manual/administration/install-community/. To enable authentication, follow https://www.mongodb.com/docs/manual/tutorial/configure-scram-client-authentication/.

Whatever the case, provide the server with the address and login details of the MongoDB server via the .env file (described in the Configuration section).

For maintenance of the database, including making backups, it may be useful to install the MongoDB Database Tools https://www.mongodb.com/docs/database-tools/.

Starting the server

Before launching the server, ensure the NODE_ENV environment variable is set to “production”. You can verify this has taken effect by reading the server’s output during launch. If the server is in development mode, you will see “Running in development mode” printed to the console when the server is launched:



```
--- AACE-capstone server ---
Running in development mode
Starting web server...
Opening database connection to "mongodb://lo
```

While in development mode, certain security features are disabled, and the server interacts with a development database rather than the real one. The server should only be exposed to users while in production mode.

To launch the server, run the “index.js” file with Node. From the root folder of the distribution, this command would be

```
node server/index.js
```

The server should then launch and connect to the database. If this is a fresh installation, the server will likely require configuration before it can connect (see Configuration section). In particular, no users will be allowed to log in until a password has been set.

Additionally, the server **MUST** be hosted behind an appropriate HTTPS reverse proxy. The Express server itself is only configured for HTTP and will not attempt to encrypt its messages over the network.

Configuration

The primary means of configuration of the server is via a .env file placed in the root of the distribution directory. This file allows you to specify a series of options, one per line, in the following format

```
OPTION=VALUE
```

Quotation marks around values are not required.

The following options are available

NAME	DESCRIPTION
MONGO_URL	Connection address to the MongoDB database. If authentication is required, it should be included in this string (default "mongodb://localhost:27017"). The format is expected to be a MongoDB connection URI, as defined by https://www.mongodb.com/docs/manual/reference/connection-string/. e.g. "mongodb://username:paSSw0rd@localhost:27017"
PORT	Port the webserver will listen on (default 80). This value can also be specified via the command line option "--port".
SMTP_HOST	Network address of an SMTP server to use to send email notifications to users.
SMTP_PORT	Port to connect to the SMTP server on.
SMTP_USER	Username to log into the SMTP server with (default no username).
SMTP_PASS	Password to log into the SMTP server with (default no password).
SMTP_FROM_EMAIL	Email address to send mail from when logged in to the SMTP server.
SMTP_ALLOW_SELF_SIGNED	Whether to allow connections to SMTP servers secured with self-signed SSL certificates. Set to true to allow these connections (default false).
CORS_ORIGIN	Restrict access to the HTTP server via the CORS Access-Control-Allow-Origin header (defaults to *, meaning all origins are allowed).

Setting a password

Users log in to the application using a shared group password. This password must be set before anyone can log in to the server. This procedure is also used to reset the password.

Ensure the MongoDB database is running and access is configured via the .env file.

Then, launch the server with the following command

```
node server/index.js resetPassword myNewPassword2022
```

The command line output will indicate the success of this reset.

Backup and restore

The server persists all its data in the MongoDB server under the "aace-capstone" database. To backup or restore the database, use the commands included in the MongoDB Database Tools.

For backing up, use the mongodump tool to create a BSON backup file (https://www.mongodb.com/docs/database-tools/mongodump/). Ensure the application server is offline while this operation is in progress.

For restoring, use the mongorestore tool (https://www.mongodb.com/docs/database-tools/mongorestore/).

To import JSON data (for example, the pre-filled list of auditors delivered with this artifact in “auditors.json”), use the mongoimport tool (https://www.mongodb.com/docs/database-tools/mongoimport). The collection these auditors should be imported into is called “auditors”. For a database running locally, the command would look like

```
mongoimport --db=aace-capstone --collection=auditors  --
file=auditors.json
```

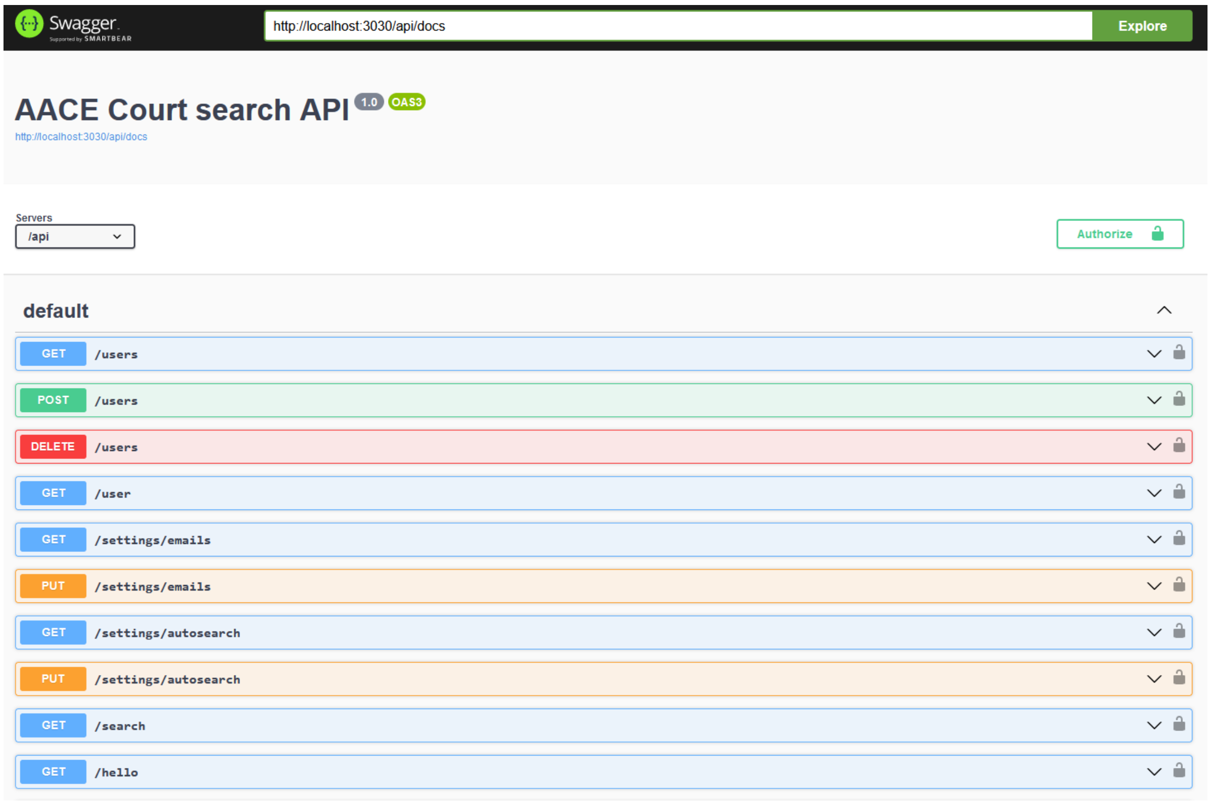
Development

When interacting with the server and installing dependencies during development, ensure the NODE_ENV flag is not set to “production”.

API documentation

The API for this server is defined according to OpenAPI version 3.0. The specification files can be found in “common” package, under “api-docs”. When the server is running in development mode, the OpenAPI specification is served at the ‘/api/docs’ endpoint.

The easiest way to view the specification is using the Swagger web UI. In development mode, a local copy of the Swagger UI app is served at the ‘/swagger’ endpoint. By entering the path to ‘/api/docs’ in the Swagger UI, you can view the documentation for each endpoint.



http://localhost:3030/api/docs

AACE Court search API 1.0 OAS3

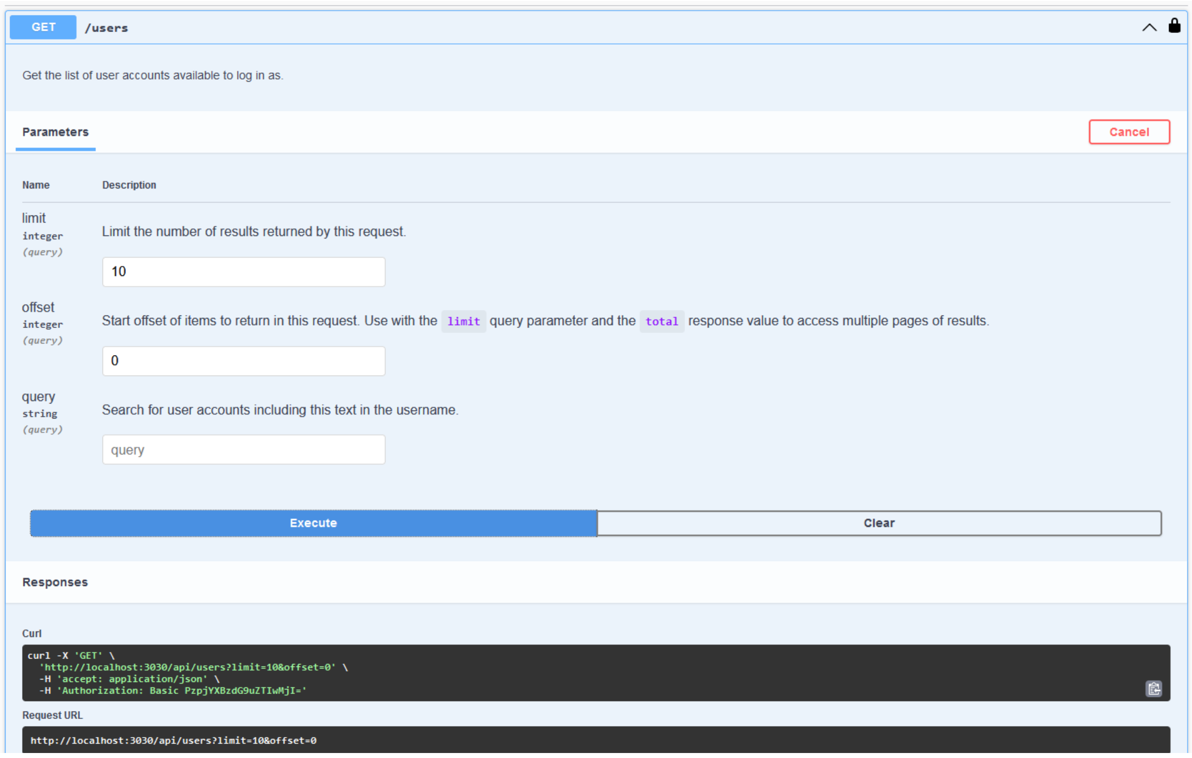
http://localhost:3030/api/docs

/api

default

GET /users
POST /users
DELETE /users
GET /user
GET /settings/emails
PUT /settings/emails
GET /settings/autosearch
PUT /settings/autosearch
GET /search
GET /hello

The web UI also allows you to try out the API by sending requests to the server. First click the “Authorize” button to enter a username and password. Then, expand any of the endpoints and click “Try it out” to send a request.



Get the list of user accounts available to log in as.

Parameters

Name	Description
limit integer (query)	Limit the number of results returned by this request. 10
offset integer (query)	Start offset of items to return in this request. Use with the `limit` query parameter and the `total` response value to access multiple pages of results. 0
query string (query)	Search for user accounts including this text in the username. query

Responses

Curl

```
curl -X 'GET' \
  'http://localhost:3030/api/users?limit=10&offset=0' \
  -H 'accept: application/json' \
  -H 'Authorization: Basic PzpjYXBzdG9uZTIwMjI='
```

Request URL

```
http://localhost:3030/api/users?limit=10&offset=0
```

Building

The project is structured as a monorepo using NPM workspaces. That is, the repository contained in the “source” folder includes three projects that can be built independently. They are “client” which contains the React web client, “common” which contains the API specification and auto-generated client library, and “server” which contains the Express server.

Before building or modifying the codebase, run “npm install” in the root directory of the repository as per the server installation guidelines.

You can build a full distribution by running

```
npm run build:full
```

The output will be saved in the “dist” folder.

To build or run individual projects, specify which workspace you would like to target using the “-w” flag. For example

```
npm run build -w common
```

Will build the common package. You should run this build command first, as both server and client depend on it.

```
npm run start -w client
```

Will start the React client in development mode (with hot reloading, debugging, etc.).

Appendix A – Available court data

The data that is published by each court database varies from state to state. The table below describes which information is accessible via searches in the application.

Location	Date range	Information available	Notes
Queensland	From 2002-2007 to today (varies by locality)	District and supreme court. All parties listed, official case title.	Detailed breakdown available. Nature of matter is available on case webpage.
New South Wales	1 week in the past, 3 weeks in the future	Supreme, District, Land and Environment; Local and Coroner's Courts; Industrial Relations Commission courts. Not all parties listed. Names are extracted from official case title, e.g., "Someone v Someone Else".	Known bug – manual searches can occasionally return duplicate results. This is an issue with the NSW server.
Australian Capital Territory	Future hearings only. 60 days ahead for Magistrates, 365 for Supreme.	Supreme and magistrates court. Names extracted from title. Maximum of two parties shown, cases with more are suffixed with "& Anor/Ors".	
Victoria	Future hearings only. Exact range unspecified.	Magistrates court. Maximum of two parties shown, split into "Plaintiff" and "Defendant" fields. No official case title.	Nature of matter is available on case webpage.
Tasmania	Daily appearance lists	None.	Not searched due to unstructured data (daily lists appear to be hand-typed).
South Australia	Future hearings only	Supreme and magistrates court. Names extracted from title. Can include more than two parties.	
Western Australia	Future hearings only	None.	Search requires a CAPTCHA.
Northern Territory	Daily appearance lists	Jurisdictions listed as "criminal" and "work health". No official title. 1-2 parties listed only.	