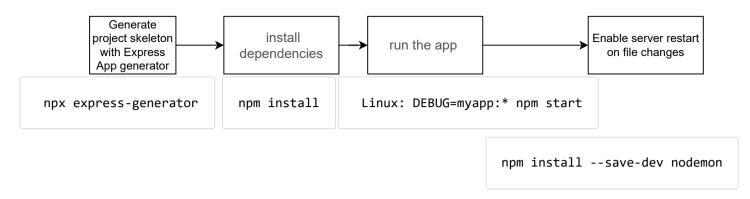
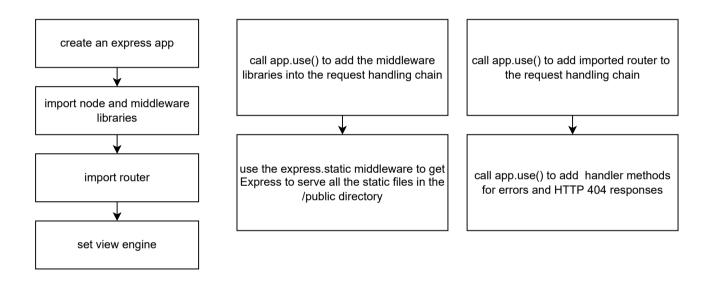
### 1. Setup app



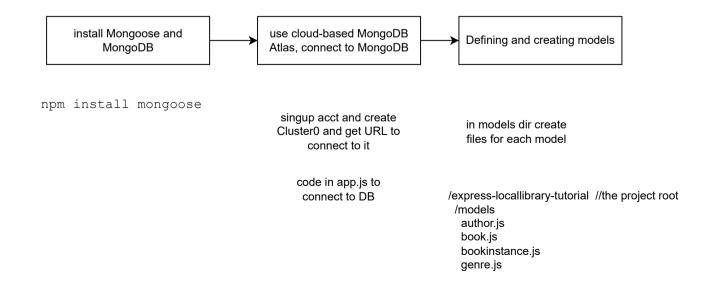
"devstart": "nodemon ./bin/www",

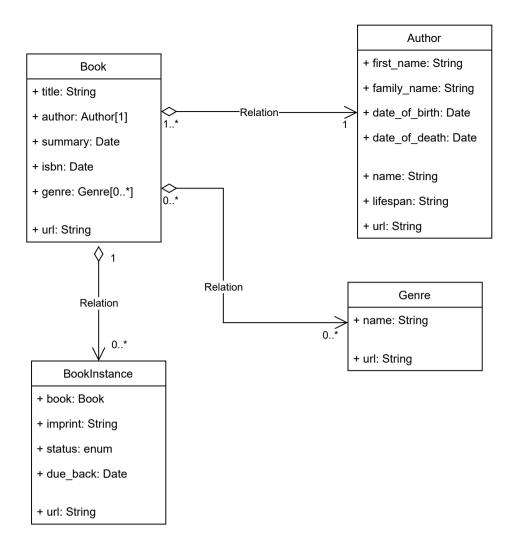
# 2. app.js file

"serverstart": "DEBUG=express-locallibrary-tutorial:\* npm run devstart"



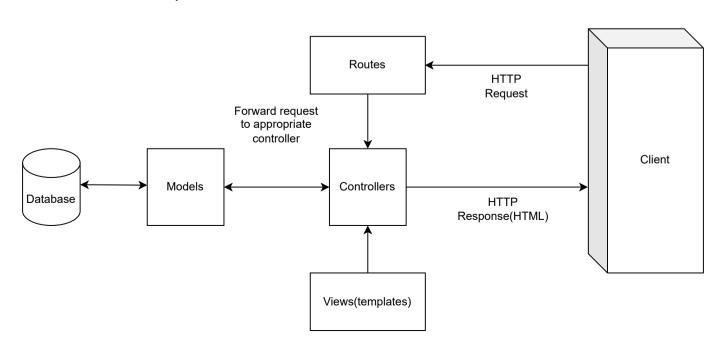
# 3. Setup DB Mongoose





Testing: using a script to create instances data

### Main flow of data and HTTP req/res handlers



HTTP verb

GET
POST

catalog.js

catalog/book

catalog/author

catalog/genre

# book\_controller + book\_create\_get() + book\_create\_post() + book\_delete\_get() + book\_delete\_post() + book\_update\_get() + book\_update\_post() + book\_detail() + book\_list()

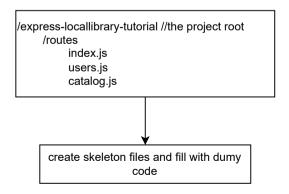
+ book\_index()

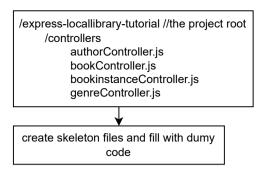
# author\_controller + author\_create\_get() + author\_create\_post() + author\_delete\_get() + author\_delete\_post() + author\_update\_get() + author\_update\_post() + author\_detail() + author\_list()

```
genre_controller

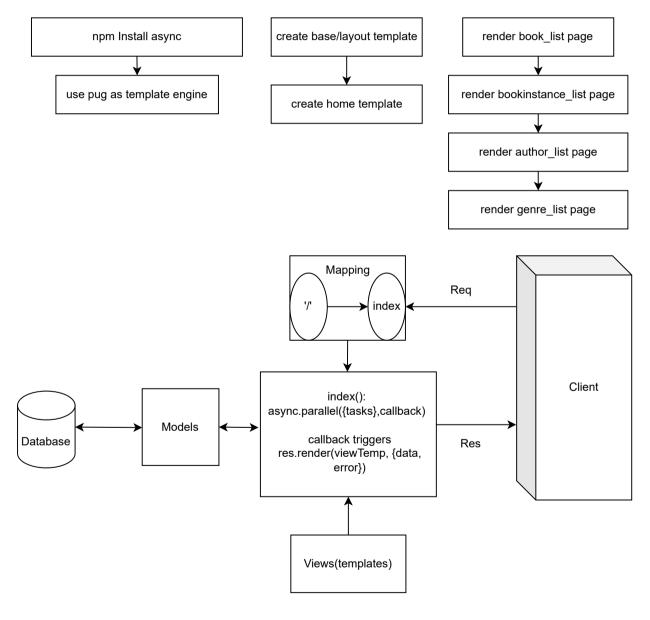
+ genre_create_get()
+ genre_create_post()
+ genre_delete_get()
+ genre_delete_post()
+ genre_update_get()
+ genre_update_post()
+ genre_detail()
+ genre_list()
```

### 5. Create routes





# 7. Config async and create templates



data obj is passed through res.render() method to create html and send back to the client

npm install luxon

const { DateTime } = require("luxon");

use:DateTime.fromJSDate(this.due\_back).toLocaleString(DateTime.DATE\_MED);

7. Work with forms 8. Deploy

create/update/delete document in DB

install middlewares needed to perform form data validation and sanitization

specify the function we want to import in each each controller file

npm install express-validator

const { body,validationResult } = require('express-validator');

### Create form

Method

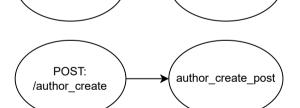
GET: /author\_create







author\_create\_get



# author\_create\_get

res.render("author\_form", {data})

# author\_create\_post

validate and sanitize data with middlewares and passing them as an array to the controller