# Programming

Build a middleware solution that will serve as an interface for Video-on-Demand applications.

## *Solution Features*

- Must provide a well-documented RESTful interface that applications can access
- Must have the ability to manage and store user-related information
- Must be able to provide access to movie metadata
- Must be capable of handling individual client sessions
- Must implement a testing framework to ensure the solution is running as expected

*Additional Features (Highly Recommended)*

- Provide middleware logs
- Provide a secure endpoint for viewings logs
- API versioning

## *Prerequisites*

- Applicants are required to use JavaScript to complete the test.
- The solution should be written using the latest stable version of Node.js
- Data passed between the middleware and client should be in JSON format
- Detailed instructions on how to set up and run the service are required.
- A live endpoint hosted on a publicly accessible server will help to facilitate grading of the test. Examples are AWS, Heroku, and OpenShift.
- Credentials or sufficient access should be provided if you are using any private solution to host or develop your service.

## *Submission and Deliverables*

Your implementation of the middleware should be delivered in a compressed archive or private Git repository containing all necessary files and resources. In case the files are hosted on a private repository, make sure to provide Accedo access. Also, instructions on how to install and start the service are expected, preferably in the form of a **Markdown file**.

Please note that all works need to be original. By submitting this programming test, you are declaring that all code in the middleware service is your original work.

Some key items that are to be submitted:

1. High-level architecture in the form of class diagrams and/or auto-generated API documentation
2. Call flows / Sequence diagrams
3. Postman collection for testing the endpoints

## *Endpoints to be Created*

User Authentication

- Register
- Sign In / Sign Out
- (BONUS) Change Password

Movie Metadata (using https://demo2697834.mockable.io/movies)

- Get Movies
- (BONUS) Provide ability to filter the response

User Preferences

- Add to History
- Add to Favourites
- Remove from Favourites
- Fetch History/Favourites List

More applicable endpoints will be marked as **bonus**

## *Useful Resources*

Movie catalog API:

[https://demo2697834.mockable.io/movies](https://demo2697834.mockable.io/movies)

OpenShift:

[https://www.openshift.com/](https://www.openshift.com/)

Heroku:

[https://www.heroku.com/](https://www.heroku.com/)

---

## *Marking Scheme (Practical Test)*

Points out of 10

- Meet the requirements (2)
- Documentation (2)
- Coding Style / Structure (4)
- Deployment / Setup / Hosting (1)
- Bonus Features (If Implemented) (1)

---

## Theory Questions

Please provide written answers to the following questions, to demonstrate your knowledge of server-side services and technologies:

1. In your own words, describe what a middleware is and why it is important.
2. Please describe the architecture of your middleware solution and why it is implemented in such a way.
3. How would you handle cross-origin resource sharing in your solution?
4. Based on your opinion, what is the best approach in setting up the middleware environments?
5. How does one test a middleware to make sure it is running according to specifications?
6. What is a secure middleware and how do you ensure you have one?
7. What are some approaches to improving middleware performance?

---

### *Marking Scheme (Written Test)*

Points out of 10

- Answering the Question (2)
- Show in-depth knowledge (4)
- Show industry relevance (3)
- Show creativity (1)

---