# NodeJS – Exam – October 2014

## File upload system in NodeJS

You are assigned to design and implement a File Upload system where visitors should be able to download a file by given link, as well as to register and login in the system. Registered users should be able to upload files, see all their files in a list and set public/private permissions. Public files can be downloaded from everyone; private files can be downloaded only from their uploader.

The system should be implemented as a server-side web application in NodeJS using Jade view engine.

### Data Layer (20 points)

* Use **Mongoose** as ORM engine and **MongoDB** as database storage engine.

2 points

* **Data layer abstraction** – the data layer should be implemented as an abstract module.

7 points

Design a simple data layer to hold **users** and **files**.

* Each **user** has **username, password** and **points** (each uploaded file gives one point). The password should be stored in the DB encrypted (not as clear text)

2 point

* + Use the **Passport** system to keep the users and their encrypted passwords.

2 point

* Each file is saved on the file system, but a reference in the database is also preserved. Keep in the database the **URL** of the file, **date of uploading, filename** and **whether it is private or not**. Each url of the file should be uniquely generated string for example “fdsfdsgfdasdasdw5435fdasda23132dasda” ☺

4 points

* Fill some **sample data** in the DB to simplify any further testing.

2 points

* Your **project should run after "copy/paste" deployment**, without changing connection strings or other settings

1 point

### NodeJS Application – Common Features (20 points)

* Use proper application architecture – separation of concerns and high quality code should be used

5 point

* **Layout** – design a layout page to reuse the common page elements like headers and footers and navigation in all other pages in the project.

1 point

* In the **navigation** when user is not authorized add link only to the home page, register and login pages.

For registered users add link for uploading files functionality and current user’s files.

3 points

* **Configure the Passport module** to enable user management functionality (login / logout).

4 points

* + The username should be between 6 and 20 characters long.

2 points

* **Error handling** – in case of errors (e.g. DB connection lost, incorrect request, etc.), an appropriate error message should be displayed. You are free to decide how exactly.

3 points

* **User interface (UI)** – the user interface should be usable enough. It is not needed to be beautiful. Use Bootstrap if you want.

2 points

### NodeJS Application – Public Area (20 points)

* **Home page** – at the application start page display statistics – uploaded files and registered users.

5 points

* + **Cache** the statistics for 10 minutes

5 points

* **Register** – registering new user

5 points

* **Login** – login a registered user

5 points

### User Area (40 points)

* **Registered users** should be able to **upload files**. Write a form for uploading files.

There should be an option for dynamically adding new inputs of files in the form

5 points

* + After uploading the file, also **add 1 point** to the user’s points.

2 point

* + Each file should receive a **unique string id**.

5 points

* + Do not upload each file in same directory. The files should not be overwritten.

5 points

* + After upload the user should get generated links for every file he/she uploaded.

5 points

* **Download** - Each file then can be downloaded by the given URL, if public, and only from its uploaded, if private

3 points

* **Profile page** – registered users (after login) should be able to see a list page with all their uploaded files sorted by date of uploading.

5 points

* + The user should be able to change the privacy of each file and delete it (through AJAX)

7 points

* + Use server side paging (page size 10).

3 points

## Evaluation Criteria

The evaluation criteria include: correct and complete fulfillment of the requirements; good technical design and appropriate use of technologies; high-quality code (correctness, readability, maintainability).

## Other Terms

During the exam you are allowed to use any teaching materials, lectures, books, existing source code, and other paper or Internet resources. Direct or indirect communication with anybody in class or outside is forbidden. This includes but does not limit to technical conversations with other students, using mobile phones, chat software (Skype, ICQ, etc.), email, forum posts, USB flash drives etc.

## Exam Duration

Students are allowed to work up to **8 hours**.