Software System Development

Semester ProjectReport File

TeamNumber:7

Submitted by:

Mohit Sharma 2022201060

Bhagyashree Barhate 2022201033

Chegu Sai Poorna Chandu 2022201062

Aryan Gupta 2022202028

Submitted to:

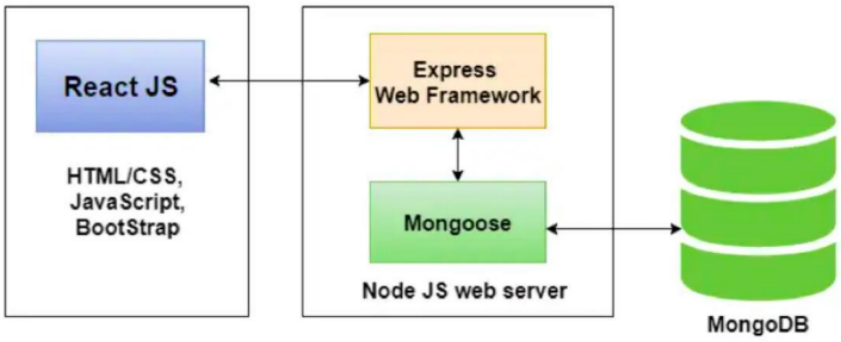
Prof. Sai Anirudh

Introduction

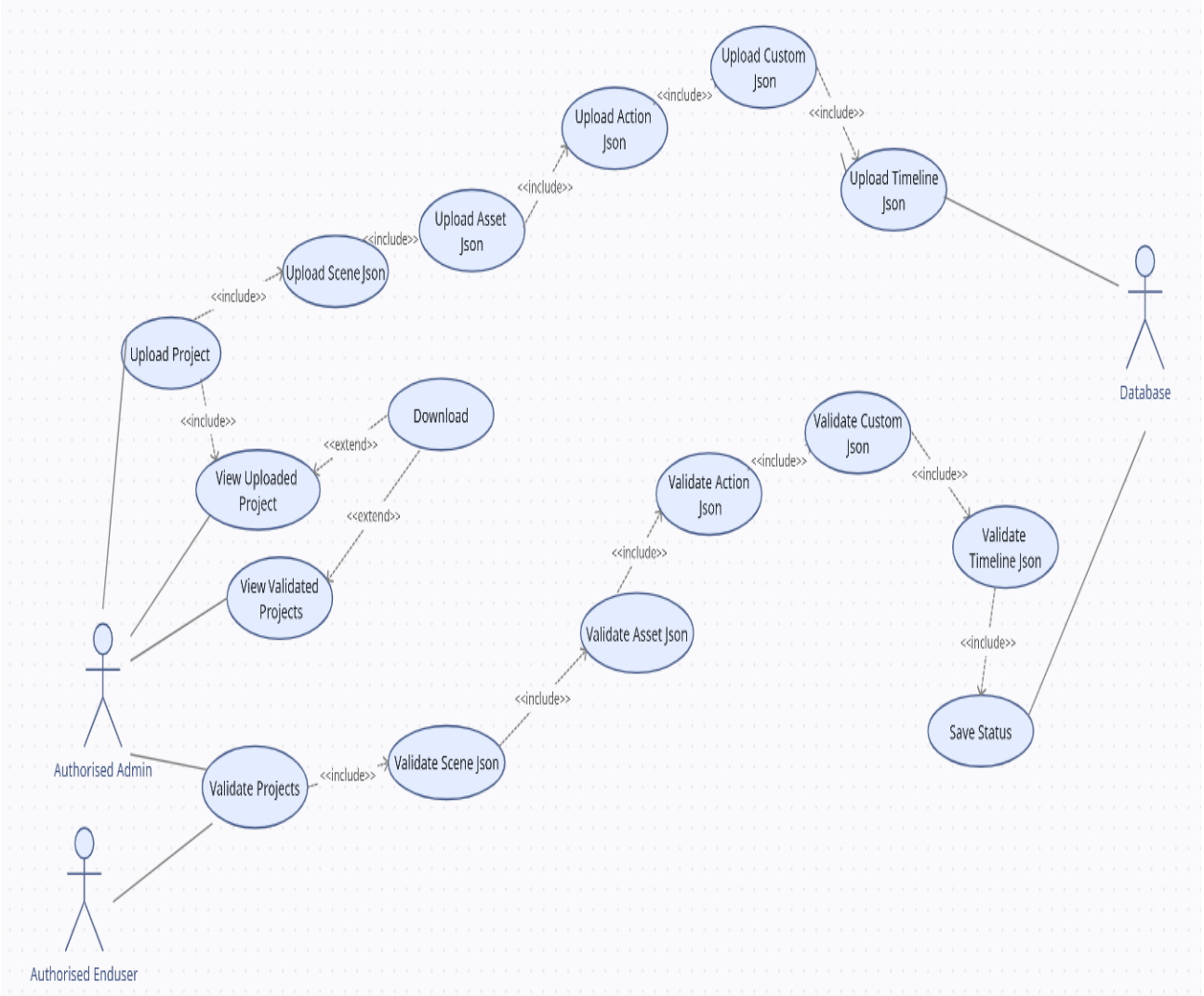
JSON VALIDATOR is a website built completely on the MERN stack. The main purpose of this project is to validate a json file against a predefined json schema.

It gives you an interface to create an account, track your work, validate and upload json schema if you are an admin.

Technology Stack- This tool is a Web based tool with a MERN technology stack. ReactJS as fronted JS framework for building UI. Express Web Framework running on NodeJS web server with MongoDB as database backend.



UseCase Diagram:



Functionality

There are two users Enduser and Admin:

Functionality for Admin:

1. Users will have to sign up as an Admin and then login.
2. Admin will then land on the Project List page where he can see all the projects validated as well as projects uploaded by him so far.
3. Admin can also download or view the json that he has validated/uploaded.
4. He can either upload schema or validate json.
5. For the above step he will have to upload/validate scene json, asset json, action json, custom json, timelineJson.
6. Admin can save the validated project only when he has validated each json with respect to their schemas.

Functionality for EndUser:

7. Users will have to sign up as an EndUser and then login.
8. EndUser will then land on the ProjectList page where he can see all the projects validated by him so far.
9. EndUser can also download or view the json that he has validated.
10. He can validate json only.
11. For the above step he will have to validate scene json, asset json, action json, custom json, timelineJson.
12. EndUser can save the validated project only when he has validated each json with respect to their schemas.

I**MPLEMENTATION LOGIC OF ADMIN**

1. Admin will upload 5 different schemas along with the templates
2. Each schema is verified whether it is syntactically valid or not
3. If all the 5 schemas are written and syntactically valid then Admin can upload

IMPLEMENTATION LOGIC OF END-USER

1. End-user first need to select the project that he needs to validate against
2. Templates of the project that he selected are shown in the text-area respectively.
3. If the User check validate button then first we will check whether it is syntactically correct or not.
4. If it is correct then User-json is validated against the schema.
5. Iterate over the User-json keys

    - If it is not present in schema keys

    - throw an error
- If the User-json key contains an object type as a value then recursively do step 5 by changing the parameters accordingly.
- Otherwise the type and required attributes of user-json are checked.
    - If the required attribute is optional then it is not compulsory for user to write the data.
    - Even if it is optional If the user writes some data then it should match with the type that is present in schema

6. Even an error occurs the logic will check for all the User-json keys
7. Repeat the same process for SceneJson,ActionJson,Assets Json,TimelineJson.
8. If all the jsons are validated properly then the user can save it.

Assumptions:

- In schema only data types of number,string,object,boolean should be given.
- Typeof and req attributes should be written for every key in the schema.
- If the type is an object then all the nested data should contain the same data type.
- The value of every key is given in a string format. It is converted to it's respective data-type.

HOW TO RUN?

- Download the zip folder from github.
- Extract this folder in your system.
- Open extracted folder in visual code studio.
- Open two terminals at this folder's directory.
- In first terminal type following commands :

    cd MERN-server
    npm instal
    node app.js
- In second terminal type following commands :

    cd MERN-client/mern-client
    npm instal
    npm start