1. Install npm init.
2. Check github and push it.
3. Cloudnary cloudinary.com
4. Project File Directory

Install express js , mongoose ,  
 npm i –save-dev prettier

Npm i –save-dev nodemon

After install these packages now setting our projects.

Project folder structure

Main folder > package.json extra

Main > public > temp > .gitkeep

Main > src > db > controller >routes > models > middlewares >utils >

Src > app.js ,index.js , constants.js

Now after these setting create two files more .prettierrc and .prettierignore

Install important packages cookie-parser , cors cors for proxy server for whitelist ip addrss and everyone can access.

After creating a file in utils folder.

Now goto create models in models folder first start in user.model.js

Video.model.js

Install package mongoose aggregate pagination v2

If you are looking for basic query pagination library without aggregate, use this one [**mongoose-paginate-v2**](https://github.com/aravindnc/mongoose-paginate-v2)

**Return first 10 documents from 100**

**Return first 10 documents from 100**

const options = {

page: 1,

limit: 10,

};

// Define your aggregate.

var aggregate = Model.aggregate();

Model.aggregatePaginate(aggregate, options)

.then(function (result) {

// result.docs

// result.totalDocs = 100

// result.limit = 10

// result.page = 1

// result.totalPages = 10

// result.hasNextPage = true

// result.nextPage = 2

// result.hasPrevPage = false

// result.prevPage = null

})

.catch(function (err) {

console.log(err);

});

**bcrypt.js its help hash password**

Optimized bcrypt in JavaScript with zero dependencies, with TypeScript support. Compatible to the C++ [**bcrypt**](https://npmjs.org/package/bcrypt) binding on Node.js and also working in the browser.

To hash a password:

const salt = bcrypt.genSaltSync(10);

const hash = bcrypt.hashSync("B4c0/\/", salt);

// Store hash in your password DB

**Jsonwebtoken**

**jwt.sign(payload, secretOrPrivateKey, [options, callback])**

**if u wana read goto website**

**mongoose provide** [**Middleware**](https://mongoosejs.com/docs/middleware.html#middleware)

* [Types of Middleware](https://mongoosejs.com/docs/middleware.html#types-of-middleware)
* [Pre](https://mongoosejs.com/docs/middleware.html#pre)
* [Errors in Pre Hooks](https://mongoosejs.com/docs/middleware.html#error-handling)
* [Post](https://mongoosejs.com/docs/middleware.html#post)
* [Asynchronous Post Hooks](https://mongoosejs.com/docs/middleware.html#post-async)
* [Define Middleware Before Compiling Models](https://mongoosejs.com/docs/middleware.html#defining)
* [Save/Validate Hooks](https://mongoosejs.com/docs/middleware.html#order)
* [Accessing Parameters in Middleware](https://mongoosejs.com/docs/middleware.html#accessing-parameters-in-middleware)
* [Naming Conflicts](https://mongoosejs.com/docs/middleware.html#naming)
* [Notes on findAndUpdate() and Query Middleware](https://mongoosejs.com/docs/middleware.html#notes)
* [Error Handling Middleware](https://mongoosejs.com/docs/middleware.html#error-handling-middleware)
* [Aggregation Hooks](https://mongoosejs.com/docs/middleware.html#aggregate)
* [Synchronous Hooks](https://mongoosejs.com/docs/middleware.html#synchronous)

Also modelSchema.methods.functionName = function(){

Write custom code there.

}

Mode.Schema.pre(“save”,function(next){

userSchema.pre("save", function (next) { //this is middleware for before save password encrypt

  if (!this.isModified("password")) return next(); //only new password only work this below code

  this.password = bcrypt.hash(this.password, 10); //10 round hash but something wrong bcz everytime encrypt everytime so use condition when new password

  next();

});

//next mongoose methods. use isPasswordCorrect

userSchema.methods.isPasswordCorrect = async function (password) {

  return await bcrypt.compare(password, this.password); //its return only true or false

}

///Generate Access Token

userSchema.methods.generateAccessToken = function () {

  return jwt.sign(

    {

      \_id: this.\_id,

      email: this.email,

      username: this.username,

      fullName: this.fullName

    },

    // eslint-disable-next-line no-undef

    process.env.ACCESS\_TOKEN\_SECRET,

    {

      expiresIn: process.env.ACCESS\_TOKEN\_EXPIRY

    }

  );

}

///Generate Refresh Token

userSchema.methods.generateRefreshToken = function () {

  return jwt.sign(

    {

      \_id: this.\_id,

    },

    // eslint-disable-next-line no-undef

    process.env.REFRESH\_TOKEN\_SECRET,

    {

      expiresIn: process.env.REFRESH\_TOKEN\_EXPIRY    } );}

Now push code git hub