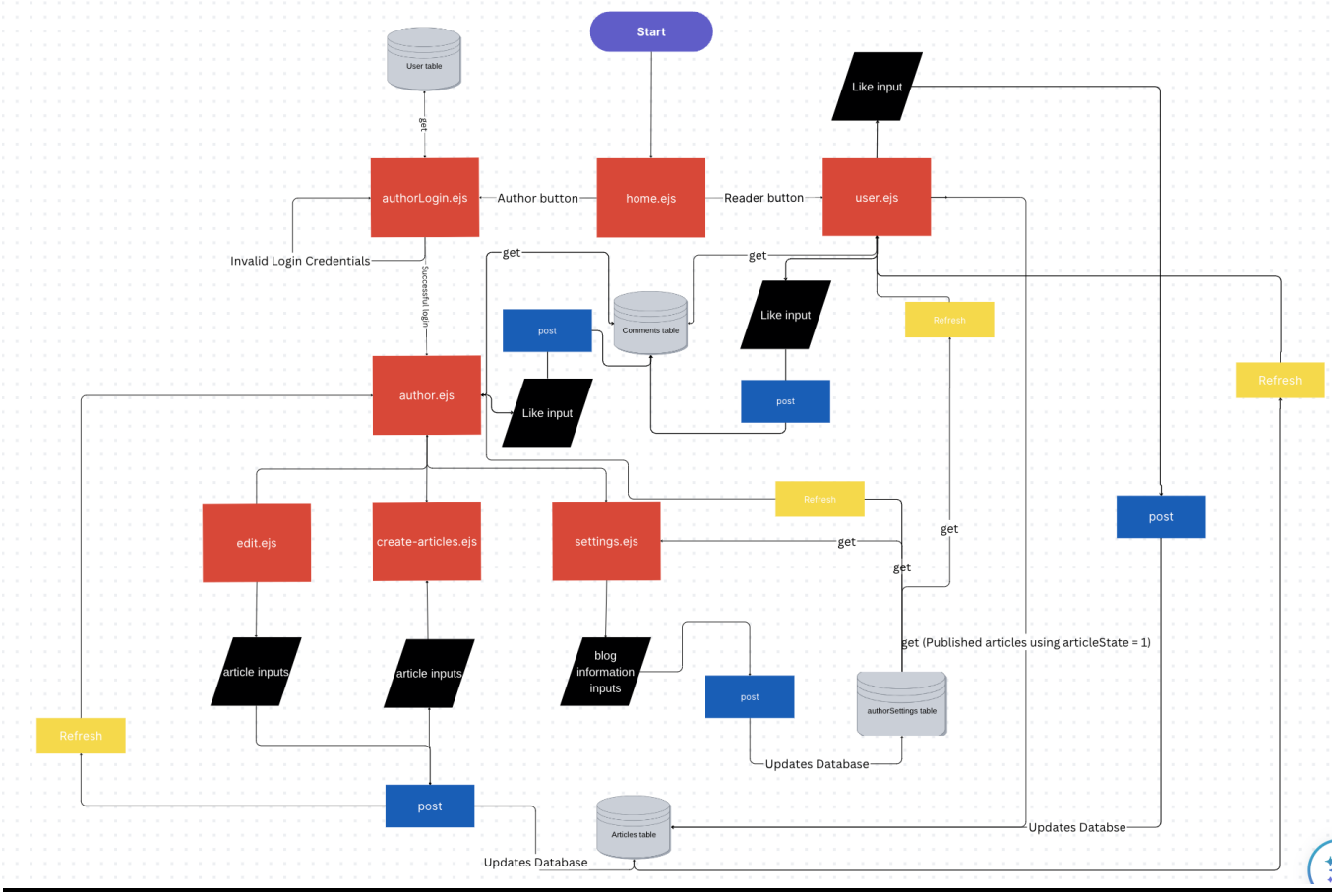
End Points



3-Tiers of architecture

Database	Application	Presentation
User: -> user_id -> user_name -> pw authorSettings -> blogTitle -> blogSubtitle -> authors_name -> authors_user_id Articles -> articleID -> articleSubtitle -> articleText -> articleLikes -> articleLastModDate -> articlePublishedDate -> articleState Comments -> commentID -> commentCreationDate -> articleID	Index.js user.js	user.ejs authorLogin.ejs authors.ejs create-article.ejs edit.ejs settings.ejs

Extension Description

For the Database, Networks and the Web Mid-term I've decided to implement a login system as part of the extension requirement.

What I've implement is a login system that checks what the user's input is and will check to see if it matches what is given in the database.

```
CREATE TABLE IF NOT EXISTS User (
    user_id INTEGER PRIMARY KEY AUTOINCREMENT,
    user_name TEXT NOT NULL,
    pw TEXT NOT NULL
);

--insert default data (if necessary here)

INSERT INTO User ("user_id","user_name", "pw") VALUES(1,"MarcusMui", "password123");
```

I utilized express session middleware for this login system. By using express session I will access to the req.session to store session data requests and in this case it would be req.session.isAuthenticated which tells the middleware that login is successful.

```
//Starts a session
const session1 = "My_1st_session";

//To config express-session middleware
app.use(
  session({
    secret: session1,
    resave: false,
    saveUninitialized: false,
  })
);
```

The requireLogin function checks to see if the user has logged into the session when accessing certain pages, author related pages, by using the if statement. When the values of req.session and req.session.isAuthenticated equals to true, it will allow access to the pages. Otherwise it would redirect them to the authorLogin page.

```
//To check if the user is authenticated
function requireLogin(req, res, next){
  if(req.session && req.session.isAuthenticated){
    return next();
  }
  else{
    res.redirect("authorLogin");
  }
}
```

The router.get(“/authorLogin”) function basically retrieves the data for the author login page.

```
//Renders the login page for author & checks if the user_id equals to what is given in the database
router.get("/authorLogin", (req, res, next)=> {
  const sqlUser = "SELECT * FROM User WHERE user_id = ?";
  global.db.get(sqlUser, [1], (err, row) =>{
    if(err){
      next(err);
    }
    if(!row){
      res.status(404).send("User not found");
      return;
    }
    res.render("authorLogin", {User: row});
  })
})
```

The router.post(“/authorLogin”) will be where the system will check the input given by the user against the data inserted into the database, as shown above, if both information matches, then req.session.isAuthenticated will be set to true and will redirect the user to the authors page. If not it will show a 401 error and the user would have to enter the url again.

```
//Sends the author login form submission
router.post("/authorLogin", (req, res, next) => {
  let sqlLogin = "SELECT * FROM User WHERE user_id = ? AND user_name = ? AND pw = ?";
  const loginData = [req.body.user_id ,req.body.user_name, req.body.pw];
  console.log(loginData);
  global.db.get(sqlLogin, loginData, (err, rows) => {
    if(err){
      next(err);
    }
    else if (rows){
      //If login is successful, will redirect to the author page & sets isAuthenticated to a true value
      req.session.isAuthenticated = true;
      res.redirect("/user/authors");

    }
    else{
      //If not will show an error upon entering wrong credentials
      res.status(401).send("Invalid Credentials");
    }
  })
});
```

Below shows the authorLogin.ejs where it mainly styles the form submission page, but the important part would be where each input has a name attribute so that the codes are able to recognize which input will be sent to the database according to their names. The name attribute will have to match the one given in the database so that it can fetch the input and place it into the database.

```
<html lang="en">
  <meta charset="UTF-8" />
  <meta http-equiv="X-UA-Compatible" content="IE=edge" />
  <link
    rel="stylesheet"
    href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/bootstrap.min.css"
    integrity="sha384-Gn5384xqQ1aoWXA+058RXPxPg6fy4IWvTNh0E263XmFcJlSAwiGgFAW/dAiS6JXm"
    crossorigin="anonymous"
  />
  <meta name="viewport" content="width=device-width, initial-scale=1.0" />
  <title>Author Login Page</title>
  <body>
    <h4>Login</h4>
    <form action="/user/authorLogin" method="POST">
      <div class="form-group">
        <label for="username">Username:</label>
        <input type="text" class="form-control" id="username" name = "user_name" aria-describedby="emailHelp" placeholder="Enter username"
        required>
        <small id="emailHelp" class="form-text text-muted">We'll never share your username with anyone.</small>
      </div>
      <div class="form-group">
        <label for="password">Password:</label>
        <input type="password" class="form-control" id="password" name = "pw" placeholder="Password" required>
      </div>
      <!-- A hidden input field for user_id -->
      <input type="hidden" name="user_id" value="<%= User.user_id %>">
      <button type="submit" class="btn btn-primary">Submit</button>
      <a href ="/user/home" class="btn btn-primary">Back</a>
    </form>
  </body>
</html>
```

```
////////////////////////////////////////// EXTENSION //////////////////////////////////////////
//To check if the user is authenticated
function requireLogin(req, res, next){
  if(req.session && req.session.isAuthenticated){
    return next();
  }
  else{
    res.redirect("authorLogin");
  }
}
//Renders the login page for author & checks if the user_id equals to what is given in the database
router.get("/authorLogin", (req, res, next)=> {
  const sqlUser = "SELECT * FROM User WHERE user_id = ?";
  global.db.get(sqlUser, [1], (err, row) =>{
    if(err){
      next(err);
    }
    if(!row){
      res.status(404).send("User not found");
      return;
    }
    res.render("authorLogin", {User: row});
  })
})

 //Sends the author login form submission
 router.post("/authorLogin", (req, res, next) => {
   let sqlLogin = "SELECT * FROM User WHERE user_id = ? AND user_name = ? AND pw = ?";
   const loginData = [req.body.user_id ,req.body.user_name, req.body.pw];
   console.log(loginData);
   global.db.get(sqlLogin, loginData, (err, rows) => {
     if(err){
       next(err);
     }
     else if (rows){
       //If login is successful, will redirect to the author page & sets isAuthenticated to a true value
       req.session.isAuthenticated = true;
       res.redirect("/user/authors");

     }
     else{
       //If not will show an error upon entering wrong credentials
       res.status(401).send("Invalid Credentials");
     }
   })
 });

 //To render the login page when someone clicks on the authors page
 router.get("/user/authors", requireLogin, (req, res) => {
   //fetches the data from the database
   res.render("authors");
 });
```