Day 21 – [21st July 2025]

TOPICS COVERED

CRUD Operations (Create, Read, Update, Delete):

CRUD is the basic concept of backend development where we handle data operations on a database.

Create:

```
app.post("/users", async (req, res) => {
  const user = new User(req.body);
  await user.save();
  res.status(201).send(user);
});
```

Read:

```
app.get("/users", async (req, res) => {
  const users = await User.find();
  res.send(users);
});
```

Update:

```
app.put("/users/:id", async (req, res) => {
  const updatedUser = await User.findByIdAndUpdate(req.params.id, req.body, { new: true });
  res.send(updatedUser);
});
```

Delete:

```
app.delete("/users/:id", async (req, res) => {
  await User.findByIdAndDelete(req.params.id);
  res.send({ message: "User deleted" });
});
```

JWT Authentication (Login System with Token):

JWT (JSON Web Token) allows us to create secure login systems without sessions.

How It Works:

- ❖ User logs in
- ❖ Server returns a JWT token
- ❖ Client stores the token and sends it in every request
- ❖ Server verifies token to authorize user

Token-Based Authentication (Header-Based):

The token is sent in the request headers:

```
headers: {
  Authorization: "Bearer <your_token>"
}
```

Protecting a Route using Token:

```
const jwt = require("jsonwebtoken");
const secret = "mySecret";
function verifyToken(req, res, next) {
  const token = req.headers.authorization?.split(" ")[1];
  if (!token) return res.status(401).send("Access Denied");
  try {
    const verified = jwt.verify(token, secret);
    req.user = verified;
    next();
  } catch (err) {
    res.status(400).send("Invalid Token");
  }
}
```

Login Route that returns a JWT token:

```
app.post("/login", async (req, res) => {
  const user = await User.findOne({ email: req.body.email });
  const valid = await bcrypt.compare(req.body.password, user.password);
  if (!valid) return res.status(401).send("Invalid Credentials");
  const token = jwt.sign({ id: user._id }, secret, { expiresIn: "1h" });
  res.send({ token });
});
```

TOOLS USED:

Express.js

MongoDB Atlas

Mongoose

JWT (jsonwebtoken): Token creation and verification

bcrypt: Hashing passwords

Hoppscotch

Nodemon