PowerPoint Presentation: Hands-On Backend Development with MERN Stack

Slide 1: Welcome to the backend development Workshop

- **Objective**: Learn to build a backend using the MERN stack
- **Agenda**:
  - MERN Stack Overview
  - Node.js, Express, MongoDB, Mongoose
  - Tools: npm, Postman, Dev Tools
  - Security: Middleware, Bcrypt, CORS
  - Data Handling: Async/Await, Axios, Zod
  - Storage: Local Storage, Cookies, Headers

Slide 2: What is the MERN Stack?

MERN

- **M**ongoDB: NoSQL database for storing JSON-like data
- **E**xpress: Node.js framework for building RESTful APIs
- **R**eact: Frontend library (not covered in depth today)
- **N**ode.js: JavaScript runtime for server-side development

Why MERN?

- Full JavaScript ecosystem
- Scalable and flexible for modern apps

Slide 3: Setting Up the Environment

Node.js & npm

- npm: Node Package Manager, installs dependencies
- Initialize project: npm init -y

package.json

- Stores project metadata and dependencies

```
{ "name": "mern-workshop", "version": "1.0.0",
"dependencies": { "express": "^4.18.2" } }
```

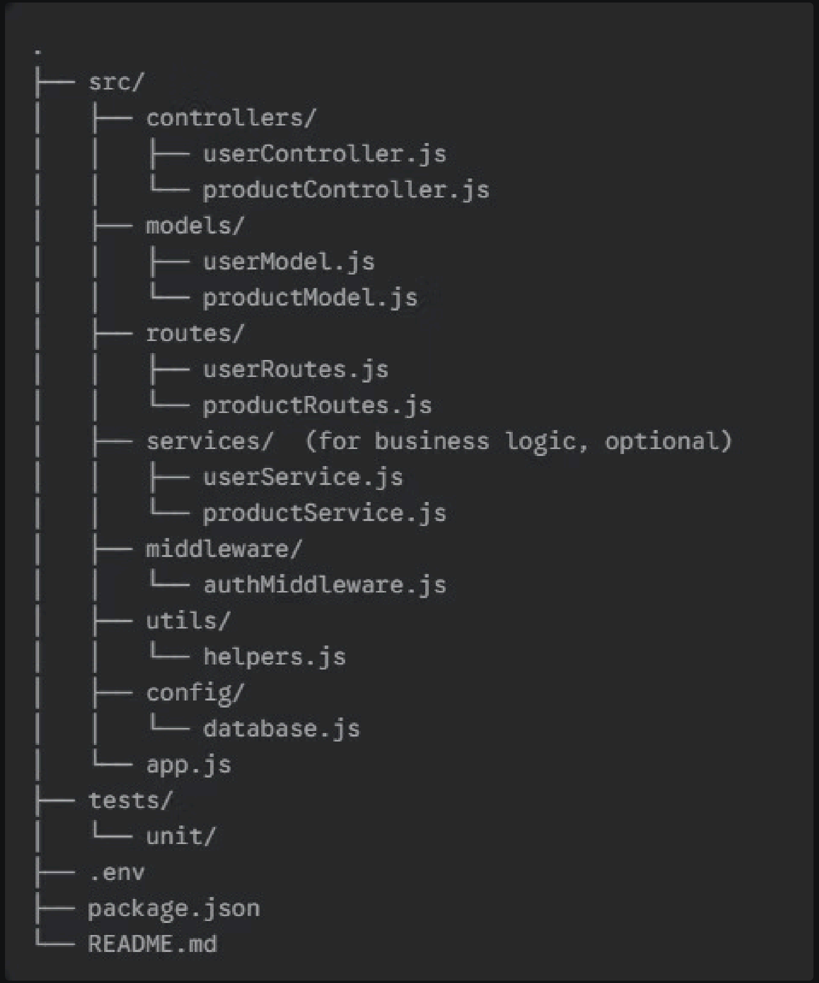
Slide 4: Project Folder Structure

Recommended Structure

```
/mern-workshop
 ├──── /node_modules
 ├──── /controllers
 ├──── /models
 ├──── /routes
 ├──── /middleware
 ├──── .env
 ├──── .gitignore
 ├──── index.js
 ├──── package.json
```

Purpose

- models: Database schemas
- controllers: Business logic
- routes: API endpoints
- middleware: Request processing



```
.
├── src/
│   ├── controllers/
│   │   ├── userController.js
│   │   └── productController.js
│   ├── models/
│   │   ├── userModel.js
│   │   └── productModel.js
│   ├── routes/
│   │   ├── userRoutes.js
│   │   └── productRoutes.js
│   ├── services/  (for business logic, optional)
│   │   ├── userService.js
│   │   └── productService.js
│   ├── middleware/
│   │   └── authMiddleware.js
│   ├── utils/
│   │   └── helpers.js
│   ├── config/
│   │   └── database.js
│   └── app.js
├── tests/
│   └── unit/
├── .env
├── package.json
└── README.md
```

Slide 5: Express.js - Building the Backend

What is Express?

- Lightweight framework for Node.js
- Simplifies routing and middleware

Basic Boilerplate

```
const express = require('express');
const app = express();
app.use(express.json());
app.get('/', (req, res) => {
 res.send('Hello, MERN Workshop!');
});
app.listen(3000, () => {
 console.log('Server running on http://localhost:3000');
});
```

Slide 6: Async/Await in Node.js

1

Why Async/Await?

- Simplifies asynchronous code (e.g., database queries)
- Replaces callbacks and promises

2

Example

```
async function fetchData() {
 try {
 const data = await someAsyncOperation();
 return data;
 } catch (error) {
 console.error('Error:', error);
 }
}
```

Slide 7: MongoDB and Mongoose

MongoDB: NoSQL database, stores data in JSON-like documents

Mongoose: ODM (Object Data Modeling) for MongoDB

Example Schema

```
const mongoose = require('mongoose');
const userSchema = new mongoose.Schema({
  name: String,
  email: { type: String, unique: true }
});
const User = mongoose.model('User', userSchema);
```

Connection

```
mongoose.connect('mongodb://localhost:27017/mern-workshop');
```

Slide 8: Environment Variables with .env

Purpose

- Store sensitive data (e.g., DB URI, API keys)

Setup

- Install: npm install dotenv
- Create .env:

```
MONGODB_URI=mongodb://localhost:27017/mern-workshop
PORT=3000
```

Use in code

```
require('dotenv').config();
console.log(process.env.MONGODB_URI);
```

.gitignore

```
node_modules/
.env
```

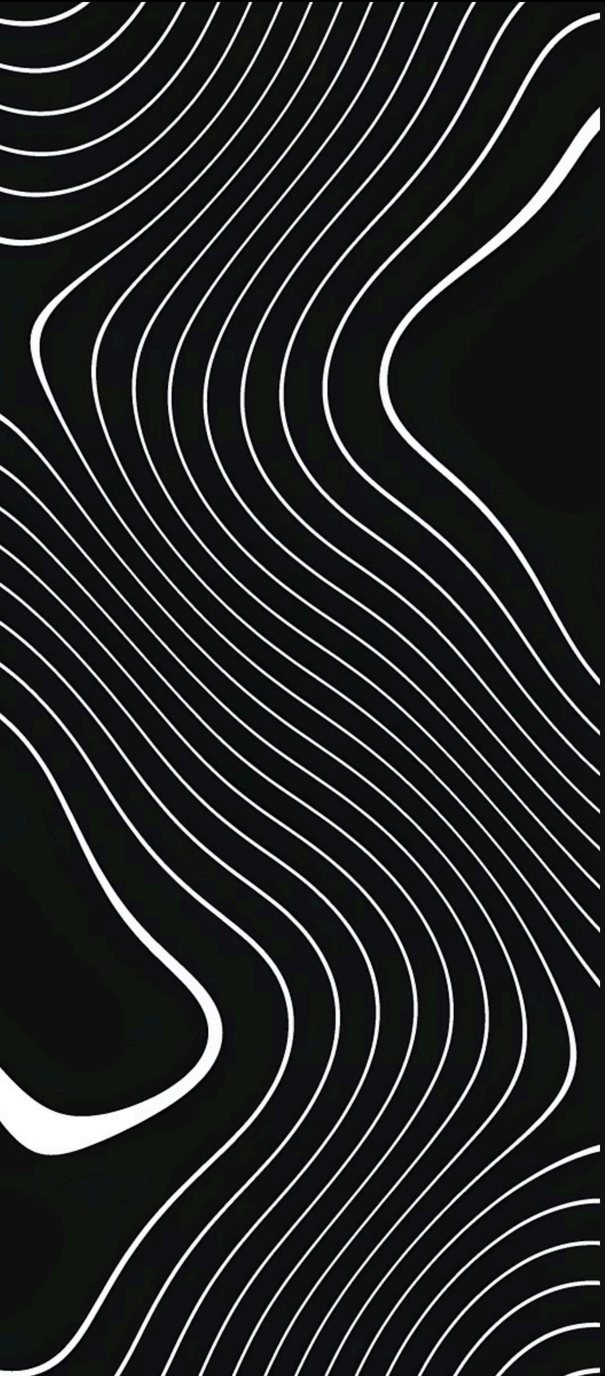
Slide 9: Middleware in Express

What is Middleware?

- Functions that process requests before routes
- Examples: Logging, authentication

Example

```
const logger = (req, res, next) => {
 console.log(`${req.method} ${req.url}`);
 next();
};
app.use(logger);
```



Slide 10: Zod Validation

Purpose

- Validate incoming data

Setup

- npm install zod

Example

```
const { z } = require('zod');
const userSchema = z.object({
 name: z.string().min(3),
 email: z.string().email()
});
app.post('/users', (req, res) => {
 try {
 userSchema.parse(req.body);
 res.status(201).send('Valid data');
 } catch (error) {
 res.status(400).send(error.errors);
 }
});
```

11.Bcrypt for Password Hashing

Securely hash passwords

Setup: npm install bcrypt

Example: const bcrypt = require('bcrypt'); async function hashPassword(password) { const salt = await bcrypt.genSalt(10); return await bcrypt.hash(password, salt); } async function verifyPassword(password, hash) { return await bcrypt.compare(password, hash); }

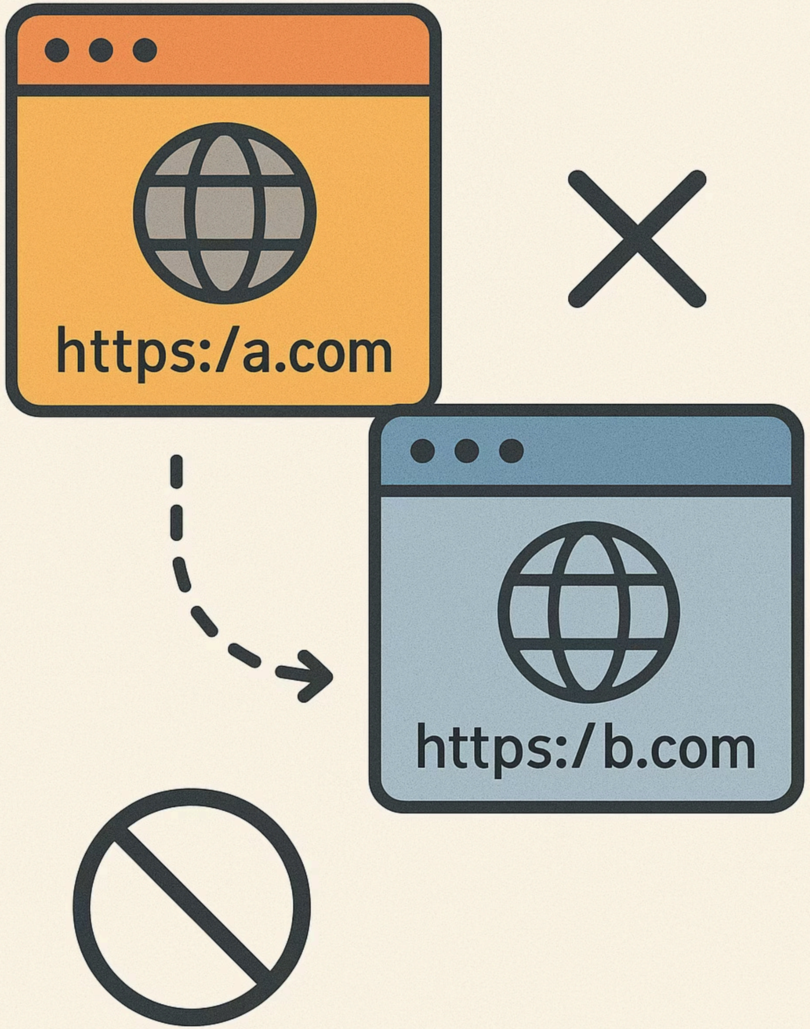
Slide 12: Axios for API Requests

- **Purpose**: Make HTTP requests from Node.js or frontend
- **Setup**: npm install axios
- **Example**:

```
const axios = require('axios');

async function fetchUsers() {

try {

const response = await axios.get('http://localhost:3000/users'); console.log(response.data);

}

catch (error) {

console.error('Error:', error);

} }
```

Slide 13: CORS for Cross-Origin Requests

- **Purpose**: Allow frontend to access backend
- **Setup**: npm install cors
- **Example**: const cors = require('cors'); app.use(cors({ origin: '**http://localhost:3000**' }));



Slide 14: Postman for API Testing

What is Postman?

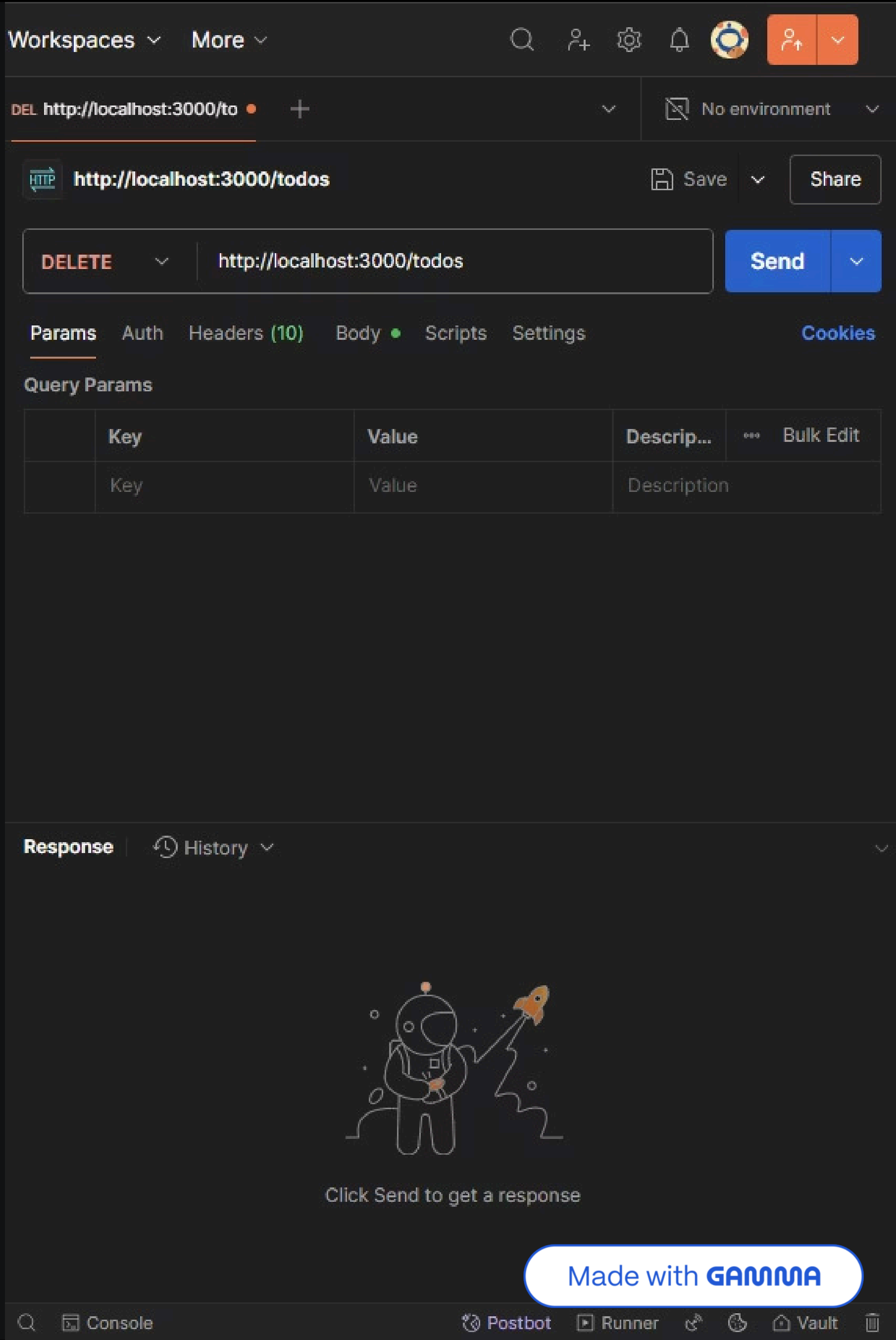
- Tool to test API endpoints (GET, POST, etc.)

Key Features

- Send requests and view responses
- Save collections for repeated testing

Example

- Create a POST request to **http://localhost:3000/users**
- Body: { "name": "John", "email": "**john@example.com**" }



Slide 15: Local Storage, Cookies, Headers

Local Storage

- Browser storage for key-value pairs
- Example: localStorage.setItem('token', 'abc123');

Cookies

- Store data sent with requests
- Setup: npm install cookie-parser

```
const cookieParser = require('cookie-parser');

app.use(cookieParser());

app.get('/set-cookie', (req, res) => {
res.cookie('session', '12345', { maxAge: 900000 });

res.send('Cookie set'); });
```

Headers

- Send metadata (e.g., Authorization)

```
app.get('/protected', (req, res) => {

const token = req.headers['authorization'];
res.send(Token: ${token});

});
```

Slide 16: Dev Tools / Inspect

Purpose

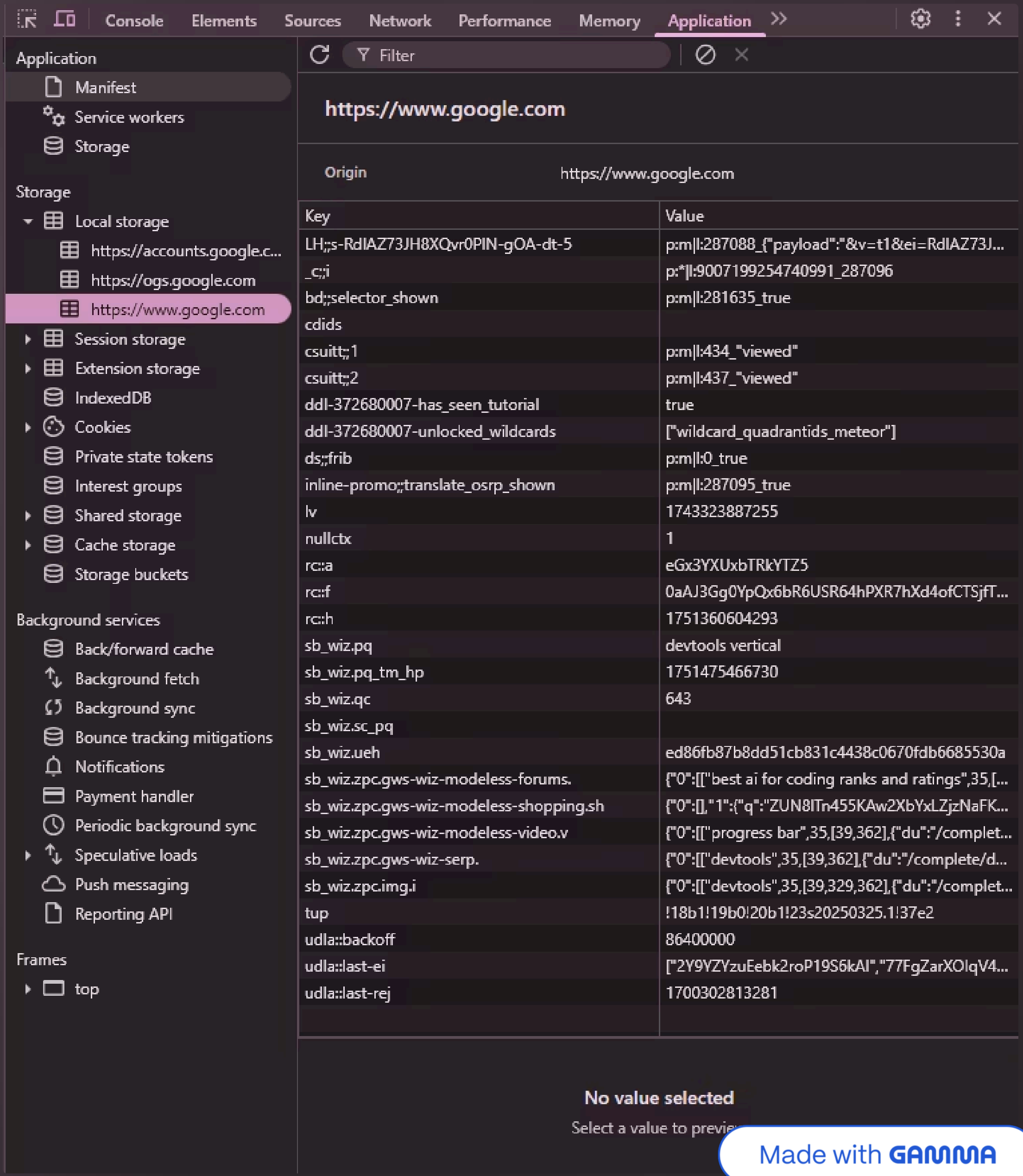
Debug backend and frontend

Key Features

- Network tab: Monitor API requests
- Console: View logs and errors

Example

- Open Chrome DevTools (F12) > Network > Test API call



https://www.google.com

Origin https://www.google.com

Key	Value
LH;;s-RdIAZ73JH8XQvr0PIN-gOA-dt-5	p:m|l:287088_{"payload":"&v=t1&ei=RdIAZ73J...
_c;;i	p:*|l:9007199254740991_287096
bd;;selector_shown	p:m|l:281635_true
cdids	
csuitt;;1	p:m|l:434_"viewed"
csuitt;;2	p:m|l:437_"viewed"
ddl-372680007-has_seen_tutorial	true
ddl-372680007-unlocked_wildcards	["wildcard_quadrantids_meteor"]
ds;;frib	p:m|l:0_true
inline-promo;;translate_osrp_shown	p:m|l:287095_true
lv	1743323887255
nullctx	1
rc::a	eGx3YXUxbTRkYTZ5
rc::f	0aAJ3Gg0YpQx6bR6USR64hPXR7hXd4ofCTSjfT...
rc::h	1751360604293
sb_wiz.pq	devtools vertical
sb_wiz.pq_tm_hp	1751475466730
sb_wiz.qc	643
sb_wiz.sc_pq	
sb_wiz.ueh	ed86fb87b8dd51cb831c4438c0670fdb6685530a
sb_wiz.zpc.gws-wiz-modeless-forums.	{"0":[["best ai for coding ranks and ratings",35,[...
sb_wiz.zpc.gws-wiz-modeless-shopping.sh	{"0":[],"1":{"q":"ZUN8ITn455KAw2XbYxLZjzNaFK...
sb_wiz.zpc.gws-wiz-modeless-video.v	{"0":[["progress bar",35,[39,362],{"du":"/complet...
sb_wiz.zpc.gws-wiz-serp.	{"0":[["devtools",35,[39,362],{"du":"/complete/d...
sb_wiz.zpc.img.i	{"0":[["devtools",35,[39,329,362],{"du":"/complet...
tup	!18b1!19b0!20b1!23s20250325.1!37e2
udla::backoff	86400000
udla::last-ei	["2Y9YZYzuEebk2roP19S6kAI","77FgZarXOlqV4...
udla::last-rej	1700302813281

No value selected

Select a value to previe

Slide 17: Hands-On Project

Task

Build a simple User API

- Endpoints:
  - POST /users: Create user (validate with Zod, hash password with Bcrypt)
  - GET /users: List all users
- Use MongoDB/Mongoose for storage
- Test with Postman

Steps

1. Set up Express server
2. Create User model with Mongoose
3. Add routes and middleware
4. Test endpoints

Deliverable

Working API with 2 endpoints

Slide 18: Wrap-Up and Q&A

Key Takeaways

- MERN stack for full JavaScript development
- Tools like npm, Postman, and DevTools streamline development
- Security with Bcrypt, CORS, and middleware

Next Steps

- Deploy backend to platforms like Render or Vercel

Q&A

Ask away!