MTech Full Stack Development & Security Cheat Sheet

React App with Routing

React Router is used to handle routing in a React application. The <BrowserRouter> enables dynamic routing.

Routes are defined using <Route> where each path renders a specific component.

```js
import { BrowserRouter as Router, Routes, Route } from 'react-router-dom';
<Router>
  <Routes>
    <Route path="/add" element={<AddStudent />} />
    <Route path="/view" element={<ViewStudents />} />
  </Routes>
</Router>
```

Add Student Component

This component takes form input and sends a POST request to the backend API to add a student.

```js
function AddStudent() {
  const [student, setStudent] = useState({ name: '', email: '' });

  const handleChange = (e) => setStudent({ ...student, [e.target.name]: e.target.value });
```

```
  const handleSubmit = async (e) => {
    e.preventDefault();
    await fetch('http://localhost:8080/api/students', {
      method: 'POST',
      headers: { 'Content-Type': 'application/json' },
      body: JSON.stringify(student),
    });
  };
  return <form onSubmit={handleSubmit}>...</form>;
}
```

Spring Boot REST API

Using Spring Boot's @RestController, we define endpoints. JPA handles persistence.

```java
@RestController
@RequestMapping("/api/students")
public class StudentController {
  @Autowired private StudentRepository repo;

  @PostMapping public Student save(@RequestBody Student student) { return repo.save(student); }
  @GetMapping public List<Student> findAll() { return repo.findAll(); }
}
```

JWT Authentication

JWT tokens are generated after authentication and sent to the client. Backend verifies them.

```java
String token = Jwts.builder().setSubject(username).signWith(...).compact();
```

OAuth2 Login

Spring Security with YAML config allows Google login.

```yaml
spring:
  security:
    oauth2:
      client:
        registration:
          google:
            client-id: your-id
```

WebSockets

Used for real-time communication like chat apps.

```java
```

@EnableWebSocketMessageBroker

public class WebSocketConfig implements WebSocketMessageBrokerConfigurer { ... }

```

```java

@MessageMapping("/sendMessage")

@SendTo("/topic/messages")

```

GraphQL Integration

GraphQL queries allow selective data fetching.

```graphql

type Query { studentById(id: ID!): Student }

```

Testing with Jest

React components can be tested using @testing-library/react.

```js

test('fee button click shows alert', () => { ... });

```
```