# OWASP Top 10 with Code Solutions (ReactJS / Node.js)

### 1. Broken Access Control

Node.js: Check roles before granting access.

Example:

```
app.get('/admin', isAuthenticated, (req, res) => {
  if (req.user.role !== 'admin') return res.sendStatus(403);
  res.send('Welcome Admin');
});
```

### 2. Cryptographic Failures

Use bcrypt for hashing passwords:

```
const hashed = await bcrypt.hash(password, 10);
```

### 3. Injection (SQL, NoSQL)

SQL: db.query("SELECT * FROM users WHERE id = ?", [userId]);

MongoDB: Users.findOne({ username: req.body.username });

### 4. Insecure Design

Implement RBAC, threat modeling, least privilege rules.

### 5. Security Misconfiguration

Use Helmet and disable headers:

```
app.use(helmet());
app.disable('x-powered-by');
```

### 6. Vulnerable and Outdated Components

Run `npm audit fix`, regularly update packages.

### 7. Identification and Authentication Failures

Store JWT in httpOnly, secure cookies.

```
res.cookie('token', token, { httpOnly: true });
```

### 8. Software and Data Integrity Failures

Use SRI for CDN, validate token/file signatures.

### 9. Security Logging and Monitoring Failures

Use logging middleware and tools like Winston.

Example:

```
app.use((req, res, next) => {
  console.log(`[${new Date()}] ${req.method} ${req.url}`);
  next();
});
```

### 10. Server-Side Request Forgery (SSRF)

Validate and whitelist URLs:

```
if (!url.startsWith('https://trusted.com')) return res.status(400).send('Invalid URL');
```