OWASP Juice Shop QA Dashboard

A comprehensive testing dashboard for the OWASP Juice Shop application that allows you to run Cypress tests and view detailed reports.

Features

- **Test Execution**: Run all tests or specific test suites with a single click
- **Test Results Visualization**: View detailed test results with pass/fail status
- **History Tracking**: Keep a record of all test runs for comparison
- **Detailed Error Reporting**: Easily identify and debug test failures
- **Responsive Design**: Works on desktop and mobile devices

Screenshots

Show Image

Show Image

Getting Started

Prerequisites

- Node.js (v14 or newer)
- npm (v6 or newer)

Installation

1. Clone the repository

bash

```bash
git clone https://github.com/yourusername/juice-shop-qa.git
cd juice-shop-qa
```

2. Install dependencies

```bash
npm install
```

3. Install frontend dependencies

```bash
cd dashboard/frontend
npm install
cd ../..
```

Running the Dashboard

1. Start the backend API server

```bash
npm run start:api
```

2. Start the frontend dashboard (in a new terminal)

```bash
npm run start:dashboard
```

3. Access the dashboard at http://localhost:3000

Running Tests

Through the Dashboard

1. Open the dashboard in your browser
2. Click the "Run Tests" button
3. View results when tests complete

Command Line

Run all tests:

```bash
npm run test
```

Run specific test suite:

```
bash

npx cypress run --spec "cypress/integration/cypress-user-management-tests.js"
```

Generate reports:

```
bash

npm run report:merge
npm run report:generate
```

Project Structure

```
juice-shop-qa/
├── cypress/                # Cypress test files
│   ├── integration/        # Test suites
│   ├── fixtures/           # Test data
│   ├── plugins/            # Cypress plugins
│   ├── results/            # Test results
│   ├── reports/            # HTML reports
│   └── support/            # Helper functions
├── dashboard/
│   ├── frontend/           # React dashboard
│   └── backend/            # Express API server
├── cypress.json            # Cypress configuration
└── package.json            # Project dependencies
```

Test Categories

The dashboard organizes tests into the following categories:

- **Authentication**: Login functionality and session management
- **User Management**: User registration, profile updates, and password resets
- **Shopping Cart**: Adding, updating, and removing items from the cart
- **Checkout**: Payment processing and order confirmation
- **Products**: Product search, filtering, and display
- **API**: Backend API functionality
- **Security**: XSS, SQL injection, and other security tests

API Endpoints

The backend provides the following REST API endpoints:

- `GET /api/suites` - Get all available test suites
- `POST /api/tests/run` - Run tests
- `GET /api/tests/status/:runId` - Get status of a test run
- `GET /api/tests/results/:runId` - Get detailed results of a test run
- `GET /api/tests/recent` - Get recent test runs

Configuration

Cypress Configuration

Modify `cypress.json` to customize test behavior:

```json
{
  "baseUrl": "https://juice-shop.herokuapp.com",
  "viewportWidth": 1280,
  "viewportHeight": 800,
  "video": true,
  "screenshotOnRunFailure": true,
  "reporter": "cypress-multi-reporters"
}
```

Dashboard Configuration

Frontend environment variables can be set in `dashboard/frontend/.env`:

```
REACT_APP_API_URL=http://localhost:5000/api
REACT_APP_DEFAULT_BROWSER=chrome
```

Extending the Dashboard

Adding New Test Suites

1. Create a new test file in `cypress/integration/`
2. Follow the same describe/it pattern as existing tests
3. Restart the backend server to detect the new tests

Customizing the Dashboard UI

The frontend is built with React and uses Tailwind CSS for styling. To customize:

1. Navigate to the frontend directory: `cd dashboard/frontend`
2. Make changes to the components in `src/components/`
3. Build the frontend: `npm run build`

Continuous Integration

GitHub Actions

A workflow file is provided in `.github/workflows/cypress.yml` for running tests on GitHub Actions.

Jenkins

For Jenkins integration, use the Jenkinsfile in the root directory.

Contributing

Contributions are welcome! Please feel free to submit a Pull Request.

1. Fork the repository
2. Create your feature branch (`git checkout -b feature/amazing-feature`)
3. Commit your changes (`git commit -m 'Add some amazing feature'`)
4. Push to the branch (`git push origin feature/amazing-feature`)
5. Open a Pull Request

License

Acknowledgments

- OWASP Juice Shop - The vulnerable web application being tested
- Cypress - End-to-end testing framework
- React - Frontend library for building user interfaces
- Express - Backend web application framework