Jisc Web Development Guidelines

Revision History

Reference: IM-DOC-003

Version	Authors	Date	Summary of Changes
4.0	Simon Potthast < Simon.Potthast@jisc.ac.uk [1]>	20/11/2015	Updates to reflect change to company name and job titles
3.0	Robert Collyer <robert.collyer@ja.net [2]>	20/02/2014	Updates to supported web browser versions.
2.0	Adam Bishop <adam.bishop@ja.net [3] > Robert Collyer <robert.collyer@ja.net [2]>	11/06/2012	Clarification to wording based on feedback. Split minimal data requirement. Relaxed normalization and environmental requirements.
1.0	Adam Bishop <adam.bishop@ja.net [3] > Robert Collyer <robert.collyer@ja.net [2]>	28/05/2012	First Edition

The latest revision can be found at https://community.jisc.ac.uk/library/janet-policies/janet-web-development-guidelines [4] with the new version.

Introduction and Rationale

To ensure high-quality, portable and stable web applications, a set of guidelines for development have been created by Jisc. The **BOLD** words are to be interpreted according to IETF RFC 2119. If there is any confusion regarding the meaning of any requirement, feedback, or suggestions for improvement you should contact the Web Development Manager, Simon Potthast.

Due to the fast-moving nature of web technologies, this set of guidelines will be an evolving standard. Revisions and changes will always be noted in the revision history above.

Definitions

- The Application: The web service to be developed
- User: The persons using the application
- Customer: A user of the application that is not an employee or contractor of Janet

General Development Practices

It is **RECOMMENDED** and encouraged that open source libraries be used as appropriate when developing the application.

If the application?s development involves working with non-web technologies governed by a standards body (such as the IETF, ISO, IEEE etc.) the application **SHOULD** conform to the relevant standards.

Except for providing failback support for older browsers and other exceptional circumstances, the application **SHOULD NOT** use proprietary plugins such as Flash, Java, or SilverLight unless no alternative is available.

The application **MUST** be architected such that there is a clear separation between presentation and logic, (such as a Model-View-Controller/Model-View-Presenter patterns, layered architecture etc.). This **SHOULD** be achieved using a templating engine so changes to style can be made without source code changes to the core application (such as Smarty).

If the deliverable makes use of a relational datastore, it **SHOULD** be normalised to the third normal form. Minor denormalisations for performance are permitted.

The **RECOMMENDED** environment for a production service is a Linux (preferably RHEL/CentOS 6 with EPEL repository) OS, Apache, Lighttpd, or NGINX web server, MySQL or PostgreSQL database. For Java developments, the **RECOMMENDED** servlet container is Jetty.

Any dynamic or user-generated content **MUST** be modifiable without changes to the application?s source code (i.e. using content management techniques).

The application **MUST** use localisation and internationalisation mechanisms, so that new languages can be added trivially, and that changes to text can be performed without source code changes.

IPv6 **MUST** be supported within the application in all aspects, including but not limited to connections to the user, connections to external data sources, and internal data structures.

An application **SHOULD** be architected such that it can scale beyond a single server deployment if required (such as MemCache integration, distributed components, message queuing etc.).

Developers **MUST** respect the licensing terms of any third party or open source code used.

Documentation and Lifecycle

A full lifecycle plan **MUST** be produced as a deliverable, covering the applications complete lifetime, including but not limited to deployment, migration to the new application, maintenance, disaster recovery, troubleshooting, and eventual migration away and decommissioning.

Before development commences, the developer **MUST** provide a plan for user acceptance testing and user story scenarios.

An automated testing mechanism (using a testing framework such as Selenium, jUnit, PHPUnit etc.) **SHOULD** be provided as part of the final delivery.

The entire toolchain to build the application and steps for deploying the application from scratch **MUST** be documented.

Any API **MUST** be fully and clearly documented using an automated documentation tool (such as DocBook, PHPDoc, JavaDoc).

Use of any third-party code and the licenses covering them **MUST** be fully documented.

If the application generates any executable code, or makes use of any generated code, the code generator **MUST** be supplied as a deliverable, and **MUST** be fully documented.

User documentation in the form of guides and manuals **MUST** be produced as a deliverable.

Front End Code and UI Requirements

Front end code for display by a web browser **MUST** conform to the latest stable version of the HTML standard at the time development commences (currently version 5), and MUST conform to the latest stable version of the W3C CSS standard (currently level 3).

If the application uses any JavaScript, it **SHOULD** make use of a JavaScript toolkit, such as JQuery/JQueryUI.

The application **MUST** conform to the W3C WCAG 1.0/AAA accessibility guidelines.

The application **MUST** be useable on desktop, mobile and tablet form factors, unless the application has specific platform constraints) Specifically, a responsive design **SHOULD** be used.

The front end **MUST** render correctly on the lowest currently supported version of the following browsers (version numbers provide for convenience): Firefox 24+, Firefox 20esr+, Internet Explorer 8+, Safari 5+., Chrome 28+, Opera 11+.

If the tablet and phone form factors are supported, the front end **MUST** render correctly on the lowest currently supported version of the following browsers (version numbers provide for convenience): Mobile Safari on iOS 6+, Android Web Browser on Android 4.1+, Internet Explorer Mobile on Windows Phone 7+.

UI designs **MUST** be signed off by the Web Development Manager prior to development commencing.

The application **MUST** follow the Jisc UX & D design guidelines.

Security and Legal Requirements

If the application consumes any user-generated data, it **MUST** implement protection against Cross-Site Request Forgery (CSRF) attacks.

If the application places any cookies onto the users computer as part of its operation, the application **MUST** implement a mechanism to obtain consent from the user as required by the

EU e-Privacy Directive.

If the application presents any user generated or otherwise sensitive content, **it MUST** deliver this content exclusively over a TLS secured connection.

All user generated data **MUST** be validated both client (i.e. using JavaScript) and server side before being used or stored.

The application **MUST** undergo vulnerability testing to a standard agreed with Jisc('OWASP Top 10' should be considered as a minimum; higher standards of testing may be required based on the results of an information security assessment). Any vulnerabilities discovered **MUST** be fixed or otherwise mitigated before release.

To satisfy data protection requirements an application **SHOULD NOT** store any information except for the minimum amount required to perform its task. Applications requiring the storage of additional data **MUST NOT** do so without an information security assessment.

Automation and Integration

If the application includes service delivery data storage and manipulation, or stores customer information, the application **MUST** implement an API capable of fully automating all operations.

An API **MUST** be RESTful, or implement a standardised stateful RPC protocol (such as SOAP, MS-XMLRPC etc.).

An authentication layer suitable for machine-to-machine authentication **MUST** be implemented in the API, such as OAUTH 1.0/2.0.

If the application requires users to log in, a federated source of identity **MUST** be used (such as UK Access Management Federation for web logins, or Moonshot Technology for non-web).

Should the application subscribe or publish data to the Enterprise Service Bus it **MUST** implement this as a message queue (i.e. ActiveMQ). If the application subscribes or publishes data to or from an external data source, it is **RECOMMENDED** that this is also implemented as a message queue.

To avoid issues with data consistency, applications utilising the Enterprise Service Bus **MUST NOT** store data acquired from other ESB Services, except for short-term caching.

If the application consumes or publishes customer information it **MUST** integrate with the Jisc CRM and **MUST NOT** store volatile customer information (such as name, address, phone), except for performance purposes (i.e. caching).

Source URL: https://community.jisc.ac.uk/library/janet-policies/janet-web-development-guidelines

Links
[1] mailto:Simon.Potthast@jisc.ac.uk
[2] mailto:robert.collyer@ja.net
[3] mailto:adam.bishop@ja.net
[4] https://community.jisc.ac.uk/library/janet-policies/janet-web-development-guidelines