**Manual de Coordinación**

**Sistema de Diagnóstico y Agendamiento de Citas (SDAC)**

**Versión 1**

**Jairo Daniel Bautista Castro**

**Miguel de Oliveira Dias Gonçalves**

**Índice**

[**Ciclo de vida** 2](#_Toc529505274)

[**Metodología de desarrollo** 2](#_Toc529505275)

[**Recursos Software Desarrollo** 2](#_Toc529505276)

[**Organización del equipo de trabajo (estructura, normas)** 2](#_Toc529505277)

[**Herramientas para comunicaciones en el equipo de trabajo** 2](#_Toc529505278)

[**Relaciones con el cliente (entrevistas, reuniones, revisiones, …)** 2](#_Toc529505279)

[**Estándares de documentación** 3](#_Toc529505280)

[**Estándares de código** 3](#_Toc529505281)

[**Control de versiones (método y herramientas)** 3](#_Toc529505282)

[**Gestión de calidad (proceso y herramientas)** 3](#_Toc529505283)

# **Ciclo de vida**

Seguiremos un ciclo de vida ágil, basado en los marcos de desarrollos Scrum y Kanban.

# **Metodología de desarrollo**

Seguiremos la metodología de desarrollo Extreme Programming. De esa forma, haremos integración y despliegue continuos; programación en pareja; desarrollo basado en pruebas, entre otros.

# **Recursos Software Desarrollo**

Usaremos el IDE Visual Studio Code tanto para desarrollar el back-end en Node.js; como también el front-end en Angular 2 y NativeScript.

Para las bases de datos, usaremos MongoDB y el IDE Robo 3T.

La herramienta de despliegue será Jenkins.

Usaremos Chef para orquestar las máquinas virtuales necesarias.

# **Organización del equipo de trabajo (estructura, normas)**

Una vez que se seguirá el marco de desarrollo Scrum, habrán 3 células de trabajo: una para las bases de datos; otra para el back-end; y una tercera para el front-end, con tamaño entre 3 a 9 personas. Cada célula tendrá un Scrum Master y un Product Owner, todos los restantes miembros de las células serán Developers.

Usaremos Trello para crear los tableros Kanban y gestionar las historias de usuario y el Product Backlog de Scrum.

# **Herramientas para comunicaciones en el equipo de trabajo**

Usaremos grupos en Slack para comunicación continua e intercambio de ficheros dentro del equipo; correo electrónico para comunicación escrita e intercambio de ficheros entre equipos; y Skype para videollamadas entre equipos.

# **Relaciones con el cliente (entrevistas, reuniones, revisiones, …)**

En el fin de cada sprint, el cliente estará presente en el “Sprint Review” de modo a ser informado de los avances en el proyecto y dar su retroalimentación del proyecto a los equipos de desarrollo. El cliente tendrá también acceso sólo de lectura a los tableros Kanban, historias de usuario y el Product Backlog creados el Trello por todos los equipos de trabajo.

# **Estándares de documentación**

Cada acta producida después de un “Sprint Review” del proyecto deberá tener la firma de, al menos, un representativo del cliente e uno de cada equipo de trabajo.

# **Estándares de código**

Todo el código será escrito en inglés, se utilizará las recomendaciones sobre código limpio, la cual busca que el código sea lo más expresivo posible, es decir que las funciones y las variables digan para que existen y porque existen, y solamente escribir comentarios cuando sea necesario.

# **Control de versiones (método y herramientas)**

Usaremos GitHub como herramienta principal de control de versiones. La rama “master” estará reservada al código final, esto es, más estable y probado; se subirá código a esa rama mediante la técnica de integración continua mediante “pull requests” que serán aceptados después de que se ejecuten los test automatizados y después que dos miembros del equipo lo aprueben.

Se realizara un “feature branch” es decir por cada “Feature” o bug del software se creara una rama.

# **Gestión de calidad (proceso y herramientas)**

El cliente tendrá acceso sólo de lectura a la rama “master” del repositorio Git y podrá a cualquier momento hacer sugestiones y proponer cambios en el código y funcionalidades, que serán analizadas por el Scrum Master y por el Product Owner.