Estado actual del proyecto de curso

UNIVERSIDAD DE ANTIOQUIA
1 8 0 3

Equipo de Innovación “InnovaTeam”

Calidad del Software

Universidad de Antioquia
Ingeniería de Sistemas
Medellín, Colombia
2019

Contexto

En el marco del desarrollo para el proyecto final para el curso *Calidad del Software* se ha realizado la creación de diferentes grupos de trabajo con un enfoque diferente, similar al flujo de trabajo en una empresa convencional de desarrollo de software. El presente trabajo es realizado entonces por el área de innovación, como una respuesta a las nuevas tecnologías que se presentan hoy en día, sin dejar de lado las necesidades del cliente como consumidor del producto. Por esto se considera pertinente llevar a cabo un estudio sobre las diferentes plataformas o frameworks, arquitecturas e infraestructura relacionadas con el desarrollo de software, para otorgar, como área de innovación, retroalimentación, guía y sugerencias sobre el camino que debe tomar el proyecto presente.

Estado actual del proyecto

Para tener una idea general del estado en el que se encuentra el desarrollo del proyecto, se define una serie de objetivos que debe ser evaluado por cada área. Esto permite tener una visión más amplia y atacar los posibles inconvenientes de forma focalizada:

- Organización y cronograma de las tareas asignadas
- Ejecución/documentación de las tareas asignadas
- Resultados

A continuación se hará reporte de los hallazgos y se realizará una serie de propuestas según lo convenido por el grupo, en caso de considerar pertinente la realización de cambios que se ajusten a los estándares actuales de desarrollo e innovación.

Área de Arquitectura y Diseño

Insumos encontrados:

- **Patrones de diseño:**
    - **Para Frontend:** Se define que se va a utilizar el patrón de diseño “Mostly Fluid”, este patrón es implementado en HTML y CSS utilizando FlexBox. En la parte lógica del front-end será manejada en Angular 7, ya definido el método de trabajo de este, manejaremos lo que es el Patrón MVC, que este ya lo trae “implementado” que trabaja con un método muy similar, manejando la Vista con los componentes de Angular, el Modelo, y la conexión con Angular Services, lo que se traspola a un un MVC.

- **Requisitos no funcionales:**
    - **Interfaz del sistema:** El sistema debe tener una interfaz de uso intuitiva, sencilla de manejar y estética, que se vea agradable a la vista.
    - **Ayuda en el uso del sistema:** La interfaz debe estar complementada con un buen sistema de ayuda, bien sea a través de tooltips o íconos fácilmente interpretables (la administración puede recaer en personal con poca experiencia en el uso de aplicaciones informáticas).

- **Mantenimiento:** El sistema debe disponer de una documentación fácilmente actualizable que permita realizar operaciones de mantenimiento con el menor esfuerzo posible.
- **Diseño de la interfaz amigable:** La interfaz de usuario debe ajustarse a las características de la web actual, utilizando características CSS como bootstrap o similares.
- **Desempeño:** Garantizar el desempeño del sistema informático a los diferentes usuarios. En este sentido la información almacenada o registros realizados podrán ser consultados y actualizados permanente y simultáneamente, sin que se afecte el tiempo de respuesta
- **Nivel de Usuario:** Facilidades y controles para permitir el acceso a la información al personal autorizado a través de Internet, con la intención de consultar y subir información pertinente para cada una de ellas.
- **Confiabilidad contínua del sistema:** La disponibilidad del sistema debe ser continua con un nivel de servicio para los usuarios de 7 días por 24 horas, garantizando un esquema adecuado que permita la posible falla en cualquiera de sus componentes.
- **Seguridad en información:** Garantizar la seguridad del sistema con respecto a la información y datos que se manejan tales sean documentos, archivos y contraseñas.

- **Evaluación de plataformas y tecnologías:**
  A pesar de ser un documento de evaluación, no se presentan los criterios bajo los cuales se van a evaluar las diferentes tecnologías de forma clara con las que se va a llevar a cabo el proyecto, y el por qué estos fueron seleccionados. Adicionalmente, no se realiza una propuesta de diferentes herramientas, la evaluación es cerrada a las tecnologías *Java* y *Angular*, por lo que no queda muy claro lo que se quiere lograr, tomando en cuenta la existencia de herramientas al mismo nivel que las propuestas.

- **Arquitectura y diagramas:**
  Se logra una gran identificación de los casos de uso y los objetivos del proyecto, así como el flujo de trabajo del mismo, Con base en esto, se encuentra una arquitectura MVVC claramente diferenciada y bien estructurada, separada entre Backend y Frontend, por lo que se va a realizar un proyecto web con comunicaciones basado en arquitectura REST. No se encuentra información sobre la infraestructura ni el despliegue del proyecto.

- **Estado del área de trabajo:**
  El estado de trabajo del área en general es muy bueno, se encontró un equipo organizado y bien liderado, haciendo uso de Git Flow para la asignación de tareas y realización de las mismas, cumpliendo el cronograma y comunicándose con las áreas que dependen de los resultados obtenidos. La documentación obtenida como resultado de su trabajo es muy completo que permite a las área de Desarrollo y Testing realizar su trabajo de forma adecuada, con documentos que respalden sus decisiones.

Área de Desarrollo

Insumos encontrados:

Dentro de los insumos encontrados se encuentra el repositorio del equipo, en la gestión de este repositorio se observan algunas falencias :

- **Control de versiones:**
    - **Convenciones:** No se tiene una convención establecida para nombre ya sea de variables, archivos, ramas y en general de cualquier cosa que pueda ser nombrada en el proyecto, se encuentran algunos nombres en inglés y otros en español.
    - **Flujo de trabajo:** No se tiene una metodología de manejo de control de versión establecida, todas las ramas salen de master, algunos miembros hacen fork del repositorio y otros no, aún hay algunos issues sin resolver del release 1, se intentan usar pull request pero se encuentra que no son debidamente usado ya que no se hace revisión y no todos los usuarios hacen uso de este mecanismo, por último los commits que se hacen en general no son descriptivos.

- **Buenas prácticas en código:**
  No se observan buenas prácticas en el código de CSS y tampoco en el código de TypeScript, no se hace uso de la notación BEM y tampoco se hace uso de las capacidades de ECMAScript6, a pesar de que se usa TSLint no se ve reflejado, además los datos que se usan para mostrar algo en en los componentes se encuentran quemados.

- **Ciclo de DevOps:**
  No se encuentran pruebas implementadas, por lo que aún no se puede llevar a cabo la integración continua, tampoco se ve evidencia el uso de herramientas para realizar el despliegue continuo y no se evidencia que se haya contemplado el uso de Docker para estos procesos. A pesar de que se tiene planeado tener varios ambientes de desarrollo los cuales serían develop, test y producción no es algo que se vea reflejado en el manejo de ramas y tampoco en la configuración de ambientes ya que no se tiene ningún ambiente implementado..

Área de Testing

Insumos encontrados:

- **Estado actual del área:**
  Inicialmente se encontraron muchas dificultades por parte del equipo por la falta de experiencia que se tenía en el área por lo cual casi todo el primer sprint fue dedicado a investigar acerca del trabajo que se requería hacer y cuál era la mejor forma de hacerlo.

  Los temas más importantes de su investigación se centraron en los tipos de pruebas que se deben hacer, dentro de las cuales se encontró muy teóricamente cuales existen y para que se hacen (pruebas de humo, pruebas de regresión, pruebas de caja negra, pruebas de caja blanca, etc), después de finalizada y asimilada toda esa información se continuó con la investigación pero se centraron en las herramientas que existen para la creación de este tipo de pruebas, en dicha investigación se dieron cuenta que existían numerosas herramientas para estos fines con lo cual optaron por reducir su búsqueda a los frameworks de pruebas que solucionan dicho problema en los lenguajes de programación seleccionados para el desarrollo de la plataforma.

Finalmente y cuando se tenia mucho mas conocimiento en el tema de testing, se realizó una investigación en la cual se encargaron de aprender acerca de automatización de pruebas, lo cual es un tema super importante a la hora de realizar software hoy en día, en dicha investigación encontraron que existían diversas librerías que se podían utilizar para la automatización de pruebas, por tanto optaron por el mismo criterio de hacer su búsqueda más específica a los lenguajes de programación que se utilizaran para el desarrollo, por tal motivo se escogió Protractor como librería para automatización del frontend y selenium para Backend.

- **Organización de equipo:**
  Como nota adicional se encontró un poco de desorden en la organización del equipo ya que es un tema nueva para ellos y fue difícil empezar a atacar la problemática sin conocimiento previos en el área de testing de un proyecto de software, sin embargo se encontró una mejoría en el transcurso se las semanas y se espera que siga así por el resto de los Sprint, hasta finalizar el proyecto.

Área de Calidad

Insumos encontrados:

- **Estado actual del área:**
  Inicialmente se encontraron muchas dificultades por parte del equipo por la falta de experiencia que se tenía en el área por lo cual las primeras semanas del sprint no se logró tener un avance significativo, sin embargo lograron resolver algunas dudas en las secciones de clase con ayuda del profesor y fue a partir de allí que se empezaron a realizar avances en los temas relacionados con el área.

  Después de esas primeras semanas el área de calidad empezó a realizar investigaciones de las métricas existentes para evaluar cada área del proyecto (desarrollo, diseño, innovación, etc) y de esta manera dar información a cada área de si se estaba haciendo un trabajando de calidad o no y en qué aspectos se debería mejorar.

  Una vez se estipularon las métricas de medición de calidad para cada uno de las áreas se generó un retraso ya que muchas de las áreas no tenían avances significativos y no tenían posibilidad de realizar su trabajo de medición de sí se estaban haciendo las cosas bien o no, por tal motivo el equipo de calidad se dio a la tarea de revisar los issues de cada una de las áreas del proyecto, en donde se indicaba cuales estaban bien redactados, cuales no y a cuáles áreas le faltaba crear sus historias de usuario, todo lo anterior lo hacían dejando comentarios claros de cuáles características debia tener una buena historia de usuario y gracias a esto muchas de las demás áreas del proyecto empezaron a tener una mejor visión del proyecto.

- **Organización de equipo:**
  Como nota adicional se observó que el área trabaja ordenadamente, ya que tiene una organización donde postean cada uno de sus avances y cada una de las métricas que debe ser utilizadas para medir la calidad en cada área, sin embargo es bueno que todo el proyecto

pueda tener acceso a esos documentos, por lo cual se sugiere adjuntarlos en las HU que son relacionadas a estos temas.

Propuestas

Con lo consignado anteriormente, se realiza una serie de propuestas enfocadas a cada área, con el objetivo de implementar las mejoras necesarias para lograr que la comunicación entre las áreas y el desarrollo del proyecto se lleven a cabo de la manera adecuada, y buscando la forma de implementar una serie de mejoras con tecnologías de vanguardia, dando como resultado un producto final robusto.

Propuestas transversales a todo el proyecto

- Hacer uso de Slack (herramienta de comunicación con enfoque corporativo) para mantener una comunicación constante y efectiva entre todas las áreas, hasta ahora cada de las áreas viene trabajando como una isla independiente y a lo máximo en comunicación con alguna otra pero no hay una comunicación eficiente donde cada área tenga conocimiento de que temas se están tratando en las demas y cual es el estado actual de estas.
- Asegurarse de que el objetivo trazado en cada una de las áreas esté alineado con el objetivo general del proyecto.

Área de Arquitectura y Diseño

- Se observa un concepto erróneo dentro de la definición de patrones de diseño en el front, si bien Angular parece que trabaja con un patrón MVC, en realidad se implementa es un patrón Model-View-View-Model (MVVM),este patrón de diseño se separan los datos de la aplicación, la interfaz de usuario pero en vez de controlar manualmente los cambios en la vista o en los datos, estos se actualizan directamente cuando sucede un cambio en ellos, por ejemplo si la vista actualiza un dato que está presentando se actualiza el modelo automáticamente y viceversa. En la definición de patrones de diseño del equipo de Arquitectura y Diseño, se recomienda cambiar el patrón MVC por este, si se pretende hacer uso de las habilidades de Angular.
- Se recomienda hacer uso de SASS en lugar de CSS, ya que permite una mejor estandarización de estilos y reutilización de los mismos.
- Para manejar el estado de la aplicación y permitir una mejor trazabilidad entre la data y los componentes se recomienda utilizar NGRX (Redux Angular).
- Se recomienda utilizar conceptos de accesibilidad web y las bondades de HTML5 semántico en el requisito no funcional de "**Ayuda en el uso del sistema**"
- Si bien la base de datos se plantea con un despliegue en la nube, se observa un despliegue en un servidor convencional, dejando de lado herramientas que permitan escalar y manejar el proyecto más fácilmente. Si se desea realizar en un servidor físico, se plantea el uso de docker para automatizar los procesos de despliegue y monitoreo del proyecto.

Área de Desarrollo

- Se recomienda implementar un ciclo de DevOPs debido a que trae consigo ventajas como la automatización a la hora de lanzar aplicaciones, herramientas de integración continua y despliegue continuo.
- Uso de la notación BEM para los selectores de las etiquetas en los archivos html.
- Usar el estándar de ECMAScript, es de vital importancia declarar bien las variables utilizando let y const para evitar errores en el código al dejar variables con el scope de una variable var.
- Aprovechar el potencial de Typescript. El tipado es uno de los fuertes de este lenguaje en donde podremos saber con facilidad el tipo de la variable o el retorno de los métodos, también permite la creación de interfaces, clases y modelos.
- Implementar Git WorkFlow para el trabajo en los repositorios, realizando commits más descriptivos, creando ramas bajo una convención de nombres, utilizar los Pull Request vinculando a integrantes del equipo para la revisión del código y funcionalidad.
- Para el despliegue del FrontEnd de la aplicación utilizar una plataforma como Heroku o utilizar AWS combinando S3 junto con Cloudfront, además de, implementar varios ambientes para producción, desarrollo y pruebas.
- A falta de un Backend y para evitar los datos quemados en el código, utilizar un servicio para “mockear” los datos, emulando así una estructura de un API Rest. Para este item se recomienda usar Mocky.io o Postman.

Áreas de Calidad y Testing

- No solo definir una HU como buena o mala porque tenga descripcion o criterios de aceptación, este puntuada y esté asociada a un milestone, también se debe revisar que dicha historia de usuario tenga un propósito, y logre llevar al equipo un paso más cerca al objetivo final.
- A medida que pasen los sprint se pueden hacer comparaciones entre cada uno y determinar si el equipo está mejorando o empeorando y en base a eso tomar decisiones.
- El área de calidad debe velar para cada área trabaje adecuadamente y exista buena comunicación entre todas, ya que se evidenció en el primer sprint que las áreas estaban incurriendo en errores por falta de comunicación.
- Elaborar plan de pruebas por cada issue que así lo requiera, de manera que pueda ser más fácil determinar si una historia de usuario de debe certificar o no.
- El área de testing debería consultar con las áreas de desarrollo e infraestructura en qué tecnologías se va a desarrollar el proyecto antes de elegir las herramientas de testing.
- Las herramientas de testing no se definen por moda o por tendencia, se debe escoger la que mejor resuelva y se acople a las necesidades del proyecto.