Universidad de San Carlos de Guatemala
Facultad de Ingeniería
Escuela de Ciencias y Sistemas
Laboratorio de Análisis y Diseño de Sistemas 1

Proyecto 2

Integrantes:	
Andrés Esteban Carvajal Morales	201612272
Kenneth Alexis Escobar Echeverría	201212563
Erick Fernando Elias Diaz	201325533
Ozmar René Escobar Avila	201602988
Lester Fernando Mazariegos Navarro	201403610

Propuesta del Proyecto

El producto deseado consiste en un portal para realizar compras en línea, en donde un usuario anónimo podrá revisar los distintos productos disponibles y agregarlos a su carrito de compras, para concretar la compra deberá de registrarse y completar la información de facturación, y se le será enviado un correo electrónico, confirmando su compra.

Metodología

La metodología de trabajo a utilizar será Scrum.

Justificación

El motivo de esta decisión radica en que se ajusta a la metodología ágil por los siguientes motivos:

- El tipo de proyecto a realizar se puede ejecutar de manera iterativa e incremental haciendo que cada una de las funcionalidades sea óptima y rápida.

- El proyecto se puede realizar en el transcurso de 6 meses por lo tanto puede ajustarse a la metodología Scrum para dar un resultado de 6 a 8 Sprints recurriendo a las necesidades y funcionalidades necesarias realizando entregables cada 2 o 4 semanas organizando al equipo de desarrollo de forma diaria en pequeñas reuniones de daily scrum.

- En la empresa que solicita el software sabe lo que necesita de manera exacta haciendo que la metodología sea adecuada para que un integrante del equipo de la empresa pueda ser Product Owner.

- La naturaleza del marco de trabajo es que tienen funcionalidades potencialmente desplegables al finalizar los sprints, por lo que se puede avanzar de una manera más organizada y definir equipos de testing específicos para cada entrega tomando en cuenta que hay que producir un software con requerimientos funcionales y no funcionales.

- El software resultante debe ser sometido a pruebas constantes ya que el sistema al ser un manejador de servicios de ventas debe ser estrictamente honesto, transparente y objetivo.

Equipo de trabajo

Estudiante	Rol
Ozmar René Escobar Avila	Product Owner
Andrés Esteban Carvajal Morales	Scrum Master
Kenneth Alexis Escobar Echeverría	Dev team
Erick Fernando Elias Diaz	Dev team
Lester Fernando Mazariegos Navarro	Tester

CI/CD

Plan

Se puede adaptar el planeamiento de lo que se va a codificar por medio de las actividades planeadas en el scrum backlog al inicio del sprint:

Code

En el transcurso del sprint se realizará todo el proceso de codificación y realización de las funcionalidades que están planificadas.

Build

Cuando se tenga la parte del código funcional potencialmente desplegable se realizará la construcción de la herramienta de entrega continua, como por ejemplo Jenkins, para que realice el despliegue de la aplicación funcional en un ambiente de depuración.

Test

Justo cuando se haga el despliegue se realizarán las pruebas correspondientes para cada parte de la aplicación para corroborar que funcione correctamente cada funcionalidad independientemente.

Deploy

Ya habiendo realizado las pruebas correspondientes se llevará a producción para que los usuarios empiecen a utilizar la aplicación de manera concreta.

Equipo de trabajo de pruebas

Andrés Carvajal	Prueba de esfuerzo Prueba de carga
Erick Elias	Pruebas unitarias Pruebas de validación

Pruebas a realizar

Pruebas unitarias

Se estará realizando pruebas unitarias para comprobar el funcionamiento de cada uno de los requerimientos a medida que se esté programando para corroborar que se están realizando correctamente y las respuestas de los componentes sea la correcta.

Pruebas de validación

Se estará realizando pruebas de validación para rectificar que las respuestas obtenidas por el software es como se necesita en el modelo de negocio y satisfaga las necesidades de los clientes.

Pruebas de esfuerzo

Se realizarán pruebas de esfuerzo con el objetivo de corroborar que el sistema será lo suficientemente robusto como para resistir la carga de varios clientes ingresando a la página todos los días, a cualquier hora y comprobar la cantidad de usuarios máximo que el sistema podría resistir.

Pruebas de carga

Se realizarán las pruebas de carga en el sistema para poder simular el software y su comportamiento con el uso en el mundo real, pudiendo controlar de manera efectiva, eficiente y veloz un gran volumen de tráfico en el cual esté estará sometido.

Pruebas de seguridad

Se realizará pruebas de seguridad para que los clientes tengan la suficiente confianza y seguridad que sus datos serán privados y que ninguna persona más que ellos mismos sean los que puedan acceder a los datos.

Documentación de pruebas

Pruebas unitarias

```
it('Obteniendo categorias',(done) => {
        chai.request(app).get("/categoria").end((err, res) => {
            if(err) done(err)
            chai.expect(res.body).to.be.an('array')
            done()
        })
    })

    it('Login correcto',(done) => {
        chai.request(app).post("/login").send({
            usuario: "admin",
            password: "admin"
        }).end((err, res) => {
            if(err) done(err)
            chai.expect(res.body).to.be.an('object')
            done()
        })
    })

    it('Obteniendo usuarios',(done) => {
        chai.request(app).get("/usuario").end((err, res) => {
            if(err) done(err)
            chai.expect(res.body).to.be.an('array')
            done()
        })
    })

    it('Obteniendo productos',(done) => {
        chai.request(app).get("/producto").end((err, res) => {
            if(err) done(err)
            chai.expect(res.body).to.be.an('array')
            done()
        })
```

```
    })

    it('carro de compras agregado',(done) => {
        chai.request(app).get("/carroProducto").end((err, res) => {
            if(err) done(err)
            chai.expect(res.body).to.be.an('array')
            done()
        })
    })

    it('Update de producto inexistente',(done) => {
        chai.request(app).post("/updateProducto/100000").end((err, res)
=> {
            if(err) done(err)
            chai.expect(res.body).to.have.length.above(0);
            done()
        })
    })

    it('Eliminar productos',(done) => {
        chai.request(app).delete("/producto").end((err, res) => {
            if(err) done(err)
            chai.expect(res.body).to.be.an('array')
            done()
        })
    })
```

Pruebas de validación

Se realizaron pruebas de validación a una versión alfa del proyecto, con la cual se determinó que se cumplían 55% de los requerimientos totales del proyecto, no se poseía con todas las funcionalidades necesarias y se decidió priorizar en esto.

Con una siguiente versión se determinó que se aumentó dicho porcentaje a un 85% de los requerimientos, por lo que se consideró un poco más completo el proyecto, pero de igual manera se recomendó seguir con los objetivos para lograr de mejor manera la realización del proyecto final.

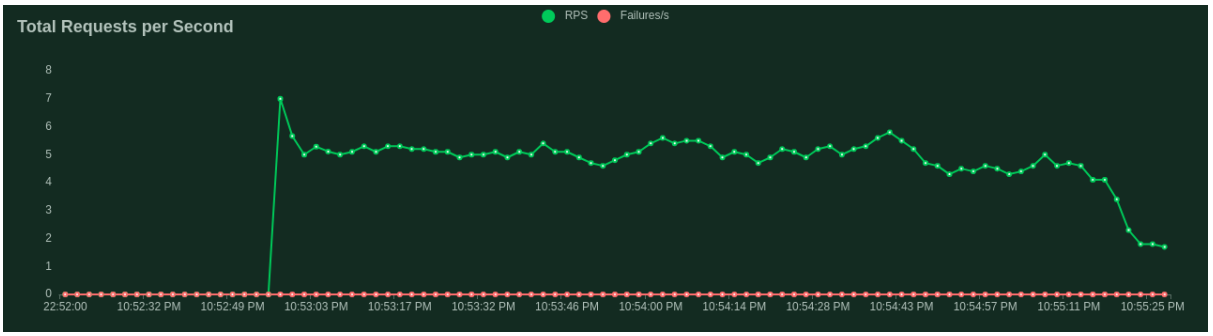
Pruebas de esfuerzo

Type	Name	# Requests	# Fails	Median (ms)	90%ile (ms)	Average (ms)	Min (ms)	Max (ms)	Average size (bytes)	Current RPS	Current Failures/s
GET	//	7404	390	1100	12000	4765	152	131211	2169	46.3	2.8
GET	//login	7246	348	1100	10000	4375	150	131200	2180	46.2	2.4
GET	//register	7263	340	1100	10000	4354	158	131168	2183	49.6	3
	Aggregated	**21913**	**1078**	**1100**	**11000**	**4500**	**150**	**131211**	**2177**	**142.1**	**8.2**

El esfuerzo realizado por 1000 usuarios registró una serie de fallas dentro de las respuestas de las peticiones en el transcurso de 5 minutos haciendo la prueba de esfuerzo la labor de proporcionar la cantidad máxima de usuarios que el sistema puede empezar a presentar fallos.

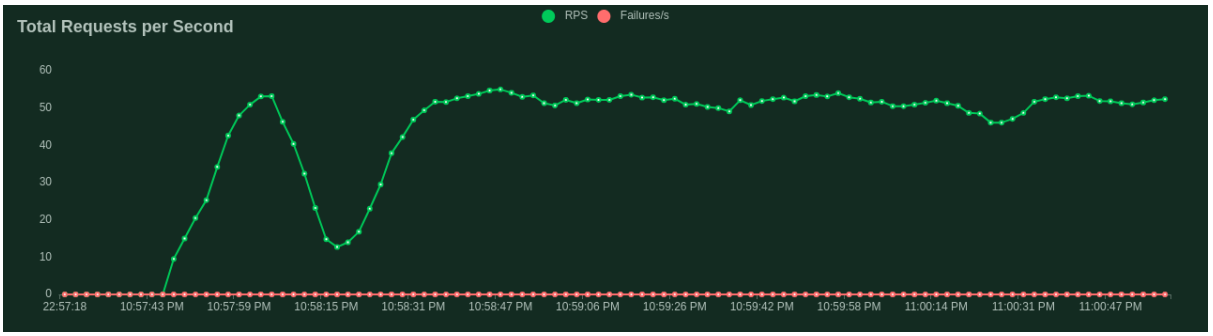
Pruebas de carga

10 usuarios



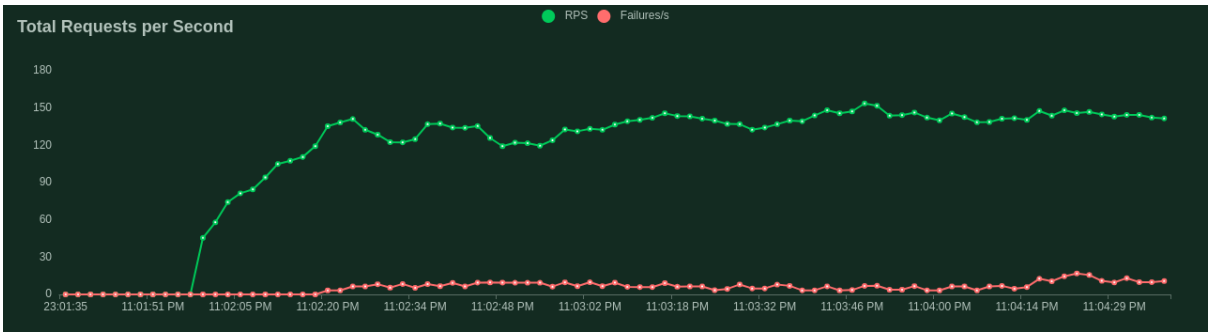
Total de peticiones a la página por segundo con 10 usuarios al mismo tiempo son aproximadamente de 5 peticiones por segundo, sin ninguna falla en el transcurso de la prueba

100 usuarios



El total de peticiones a la pagina por segundo con 100 usuarios al mismo tiempo son de aproximadamente 50 peticiones por segundo, sin ninguna falla. Se observa un valle de lentitud en las peticiones en el inicio de la prueba, sin embargo, se nivela nuevamente.

1000 usuarios



Total de peticiones a la página por segundo con 1000 usuarios al mismo tiempo son aproximadamente de 150 peticiones por segundo, empezando a tener fallas luego de que empiezan a ingresar peticiones de 800 usuarios al mismo tiempo.

Herramienta CI/CD

Jenkins

Jenkins es una herramienta open-source de automatización hecho en java vía plugins hecho para realizar propósitos de integración continua. Esta herramienta está hecha para probar y desplegar proyectos de software para que se convierta en una tarea fácil para los desarrolladores el acomodar cambios dentro del proyecto y hacer más simple el proceso de colección, construcción y despliegue.

Se utilizó un Pipeline como medio para obtener datos de un repositorio de Github, y darle instrucciones de los pasos que tiene que seguir cuando se dispare un cambio en el repositorio. Se utilizó para el frontend la librería de ReactJS y NodeJS, los comandos fueron de instalación de librerías, construcción y copia de los archivos a un servidor de Nginx para poder publicar la página web.