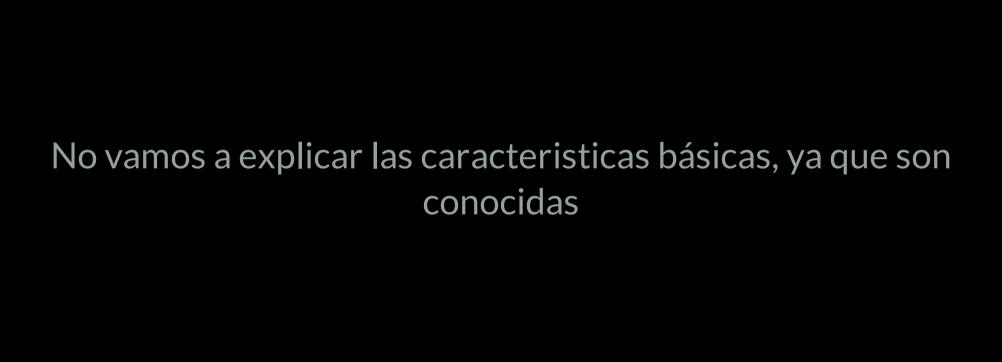
## MENSAJERO

EN

15 MINUTOS

Una pequeña introducción al proyecto detras del sistema de mensajería



#### SERVIDOR - PROBLEMAS

Se buscó resolver, siempre mendiante el uso de estandares abiertos

- Internet es un medio inseguro
- Overhead y lentitud de request pull para actualizaciones

#### SERVIDOR - PROBLEMAS

Se buscó resolver, siempre mendiante el uso de estandares abiertos

- Internet es un medio inseguro -> Encriptacion SSL
- Overhead y lentitud de request pull para actualizaciones -> Notificaciones PUSH

#### SERVIDOR - ENCRIPTACION SSL

Se permite la utilizacion del protocolo HTTPS y la utilización de certificados auto firmados

```
$ ./server --help
Usage: ./server [OPTION]
  -p, --port=puerto
                                Especificar el puerto a utilizar
  -l, --log-file=file
                                Especificar path de archivo de log
  -L, --log-level=level
                                Especificar nivel de log
  -t, --threads=num
                               Cantidad de threads
  -d, --db-path=level
                               Path de la base de datos
  -s, --ssl=file
                               Path del certificado ssl
$ ./server -s localhost server.pem
Version: 1.0
Configuracion:
    * port: 8000
    * threads: 1
    * logFile: -
    * dbPath: /tmp/test db
    * ssl: localhost server.pem
```

#### **NOTIFICACIONES PUSH**

Son mensajes que se envían de forma directa a dispositvios móviles, sirven para notificar de cambios al cliente de una manera rápida y eficiente.

Se utilizó el protocolo SSE (Server sent events) de HTML5.

```
$ curl -H "Accept: text/event-stream" http://localhost:8000/notification
id: 000519ca97cddd46
data: {"id":"000519ca97cddd46","time":"1435732804","type":"message","dat
id: 000519ca97cddd60
data: {"id":"000519ca97cddd60","time":"1435732816","type":"message","dat
```



#### COMO GARANTIZARLO?



- Planificación
- Control
- Trazabilidad de los cambios
- Proceso de desarrollo definido
- Conocimiento del sistema
- Comunicación

#### PROCESO DE DESARROLLO

Se deben utilizar herramientas que sean accesibles en cualquier entorno de trabajo.

- 1. Elección de una tarea a realizar del sistema de tareas
- 2. Resolución de la tarea
- 3. Desarrollo de tests
- 4. Actualizacion de la documentación
- 5. Generación de nuevos tickets

### **AUTOMATIZACIÓN**

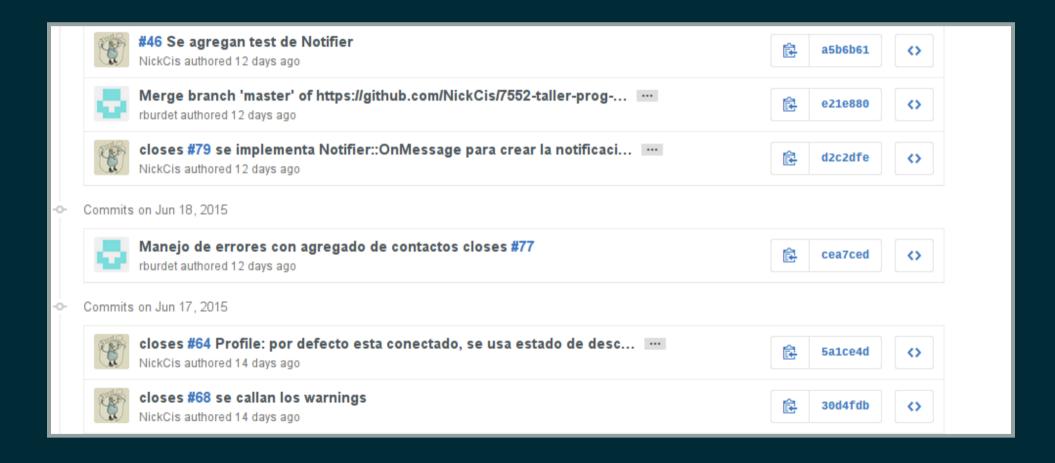
Un programador es vago por definición:

"Laziness: The quality that makes you go to great effort to reduce overall energy expenditure. It makes you write labor-saving programs that other people will find useful, and document what you wrote so you don't have to answer so many questions about it. Hence, the first great virtue of a programmer." La documentación, herramientas de testing y el proceso de desarrollo deben tener minimas dependecias para el desarrollador. Todo debe estar automatizado en la nube.

#### SISTEMA INTEGRADO DE TAREAS



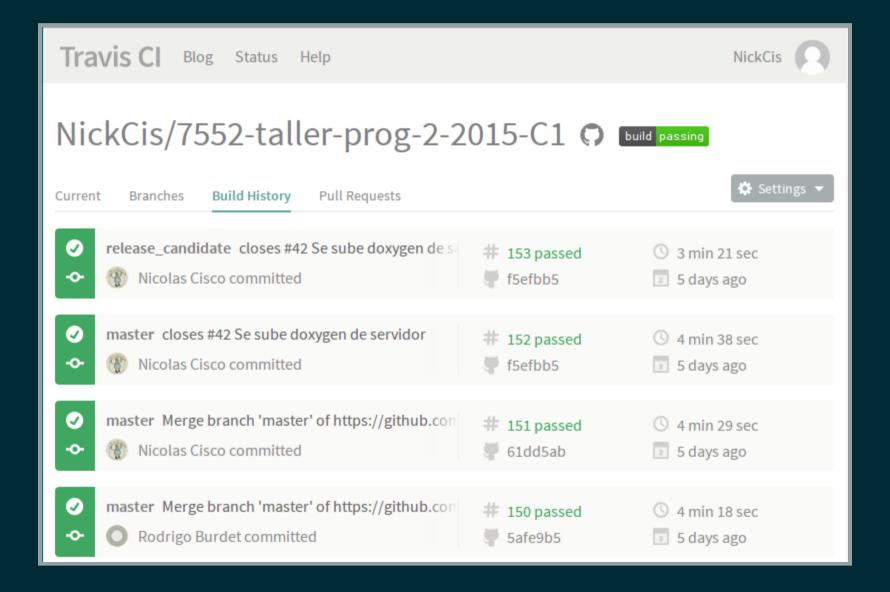
#### SEGUIMIENTO DE LOS CAMBIOS



## TEST UNITARIOS + TEST DE INTEGRACIÓN

- Test Unitarios: Catch (C++ test framework)
- Test Integración: Chakram + Mocha (Requests Node.js)

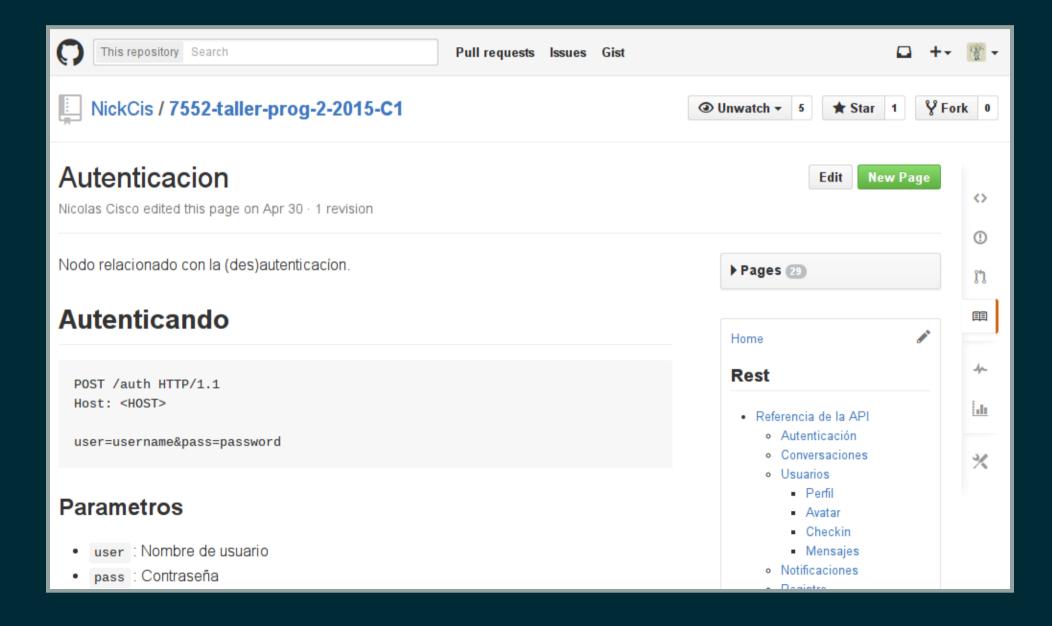
## SISTEMA DE INTEGRACIÓN CONTINUA



#### **CODE COVERAGE**



## DOCUMENTACIÓN



# THE END

- Alguna pregunta?