Cloud Recovery
Sprint 3

Carlton Duffett, Deema Kutbi, Emilio Teran
Konstantino Sparakis, Minhan Xiang

ReClo Overview

Our Purpose:

Provide an affordable, simple means for recovery of failed servers

Our Clients:

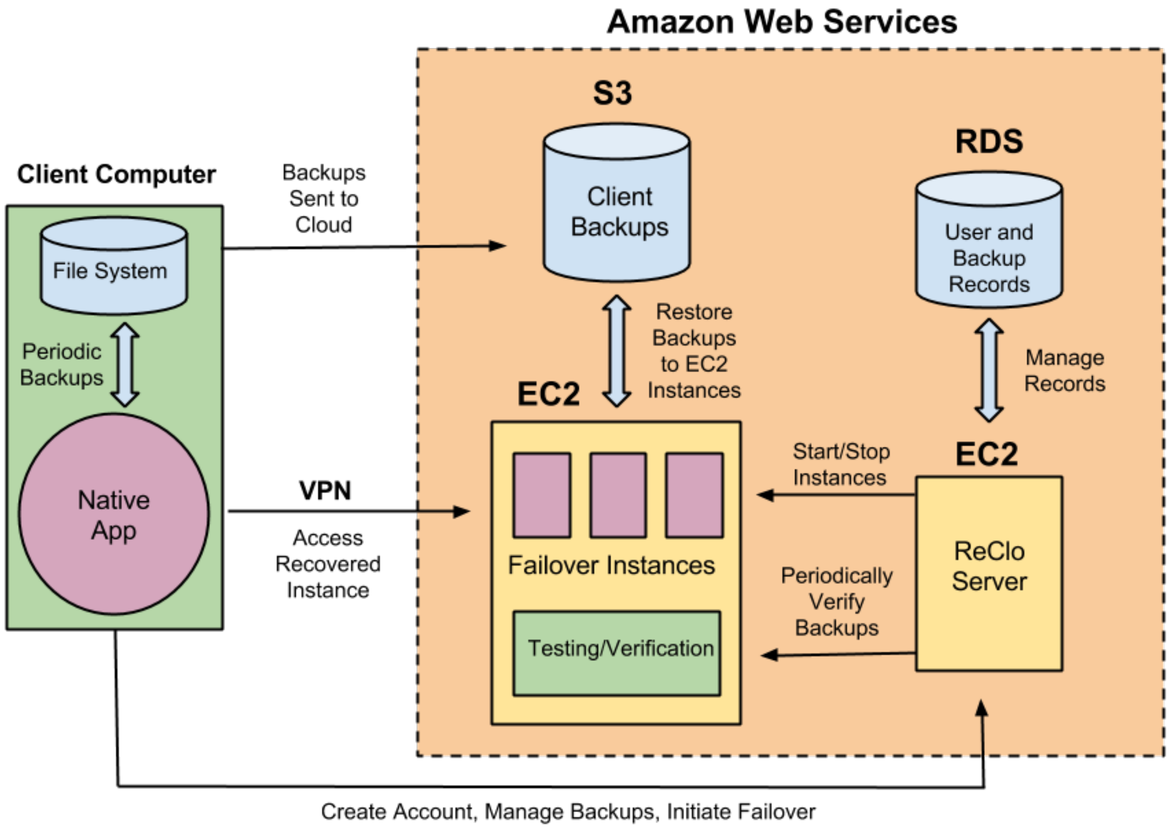
Small to medium-sized businesses, like franchise stores or doctors offices

Key Features:

Backup Manager performs regular backups and pushes them to the cloud

Recovery Manager starts a new EC2 instance from a backup file and establishes a VPN connection, allowing users to access their data

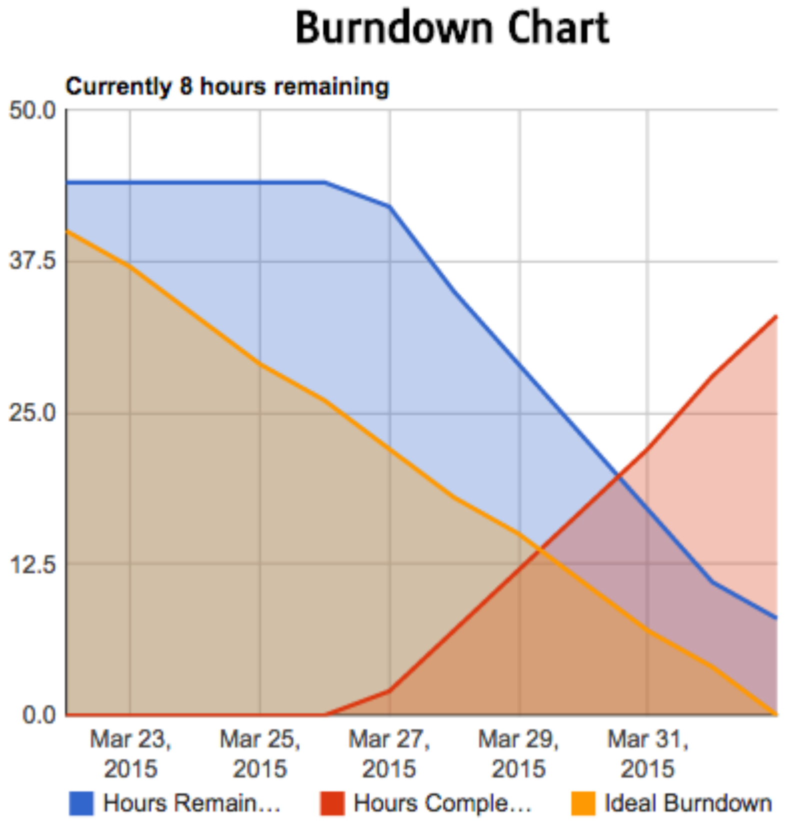
Architecture



This Sprint: Our Goals

- Functional Backup Manager and Recovery Manager UIs
- Incremental Backups
- Complete API (node.js) and client library (C#)
- Upload backups to S3
- Better estimation of tasks and better burndown pace

This Sprint: Burndown

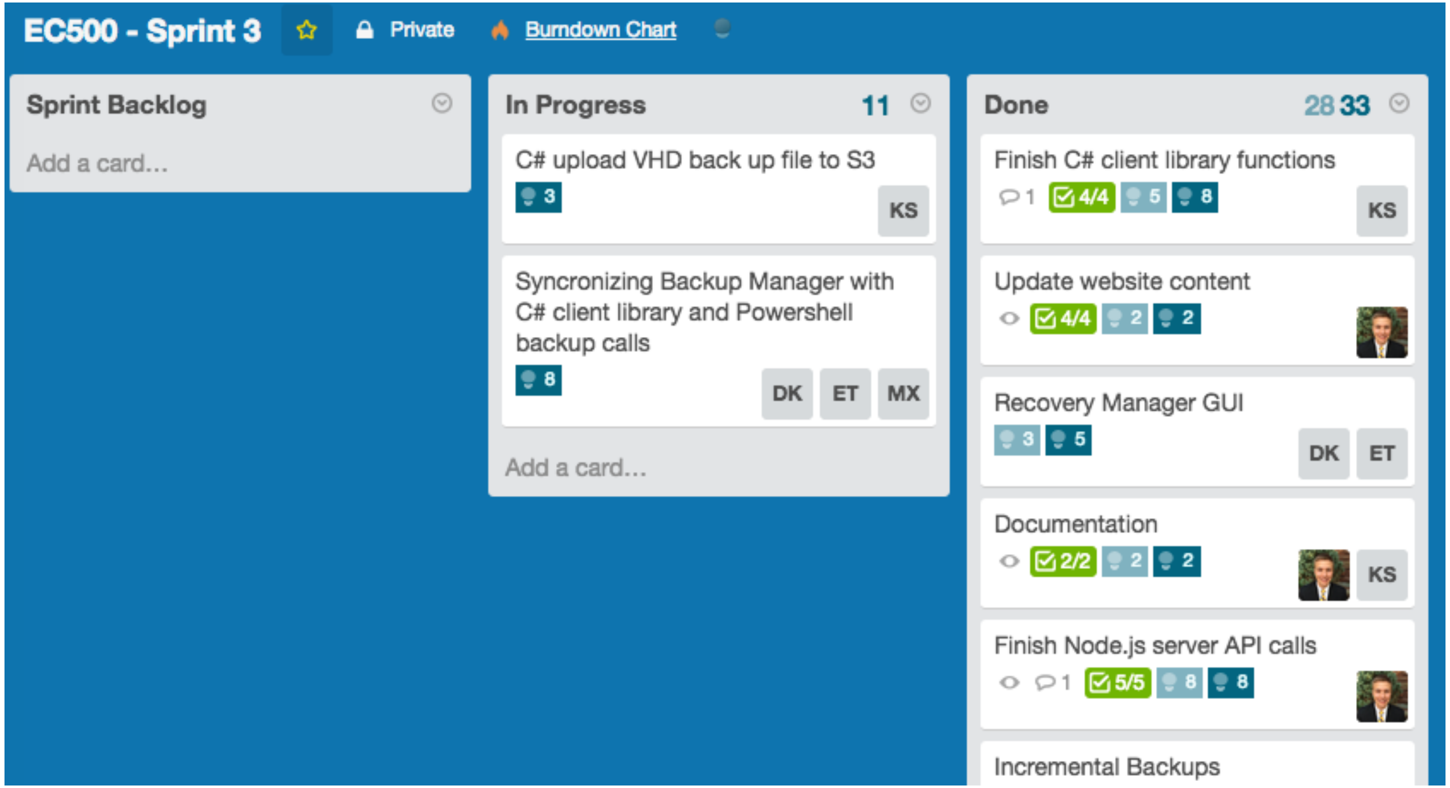


Our best burndown yet.

Took a while to do research and implement features before anything was completed.

Getting close to the finish!

This Sprint: Trello



EC500 - Sprint 3 ☆ Private Burndown Chart

Sprint Backlog

Add a card...

In Progress 11

C# upload VHD back up file to S3
3 KS

Syncronizing Backup Manager with C# client library and Powershell backup calls
8 DK ET MX

Add a card...

Done 28 33

Finish C# client library functions
1 4/4 5 8 KS

Update website content
4/4 2 2

Recovery Manager GUI
3 5 DK ET

Documentation
2/2 2 2 KS

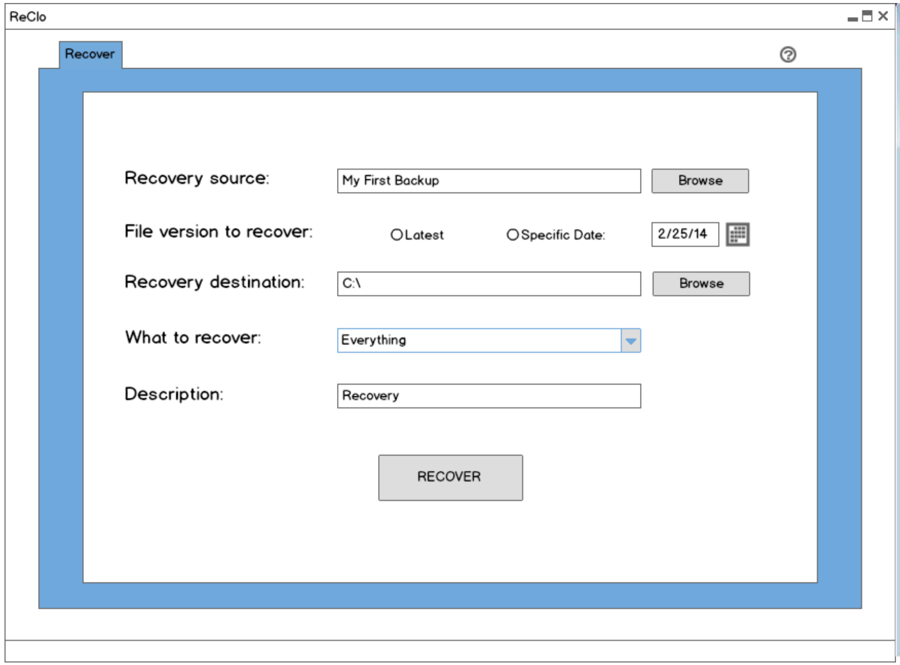
Finish Node.js server API calls
1 5/5 8 8

Incremental Backups

What We Accomplished:

- **Fully-Tested API** and client for user_auth, backups
- **Expanded API** for starting, stopping EC2 instances
  (still working on starting instances from backups)
- **Incremental Backups** from Backup Manager
  (still working on uploading these backups to S3)
- Recovery Manager UI
- Cron jobs to periodically maintain our resources

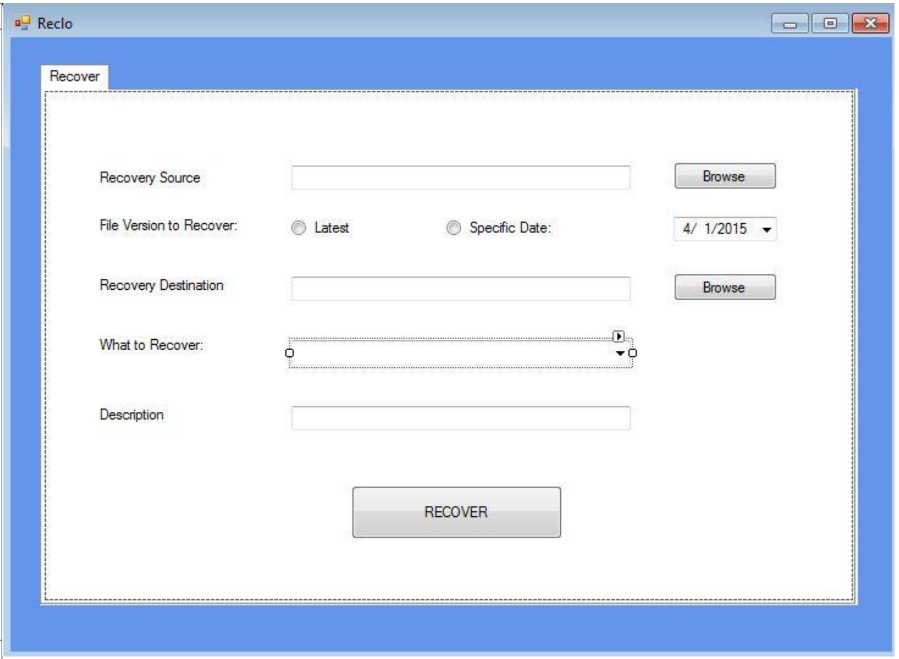
Recovery Manager Application


ReClo

Recover

Recovery source:	My First Backup	Browse
File version to recover:	○Latest ○Specific Date:	2/25/14
Recovery destination:	C:\	Browse
What to recover:	Everything	
Description:	Recovery	

RECOVER


Reclo

Recover

Recovery Source		Browse
File Version to Recover:	◌ Latest ◌ Specific Date:	4/ 1/2015
Recovery Destination		Browse
What to Recover:		
Description		

RECOVER

Backup Manager

Incremental Backups

Tool: Cobian Backup 8

- Supports incremental backups
- Supports command line operations
- Free
- Backed-up files are kept in original format

Incremental Backups

What we do:

- Initialize Cobian backup task
- Execute Cobian task from our backup manager
- Convert the backed-up file into .vhd format

RESTful API

Added API calls to start, stop, and get information on EC2 instances

Amazon EC2

Uses node.js (lightweight, asynchronous JavaScript) for the server.

C# client library for use with the applications.

Cron Jobs

ManageSessions()

Periodically scans our database and deactivates expired session tokens

FreeInstances()

Permanently terminates any stopped instances that have been idle for more than 2 weeks

Forever.js

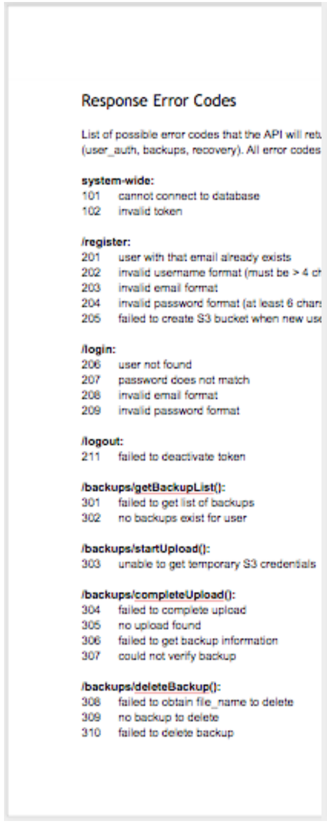
Easy to Use:

Perfect tool for running a stand alone web server. No need for Apache or nginx.

Fault tolerant:

Keeps our server running continuously and automatically restarts the server when it exits unexpectedly.

API Documentation



Response Error Codes

List of possible error codes that the API will ret
(user_auth, backups, recovery). All error codes

system-wide:

101	cannot connect to database
102	invalid token

/register:

201	user with that email already exists
202	invalid username format (must be > 4 ch
203	invalid email format
204	invalid password format (at least 6 chars
205	failed to create S3 bucket when new use

/login:

206	user not found
207	password does not match
208	invalid email format
209	invalid password format

/logout:

211	failed to deactivate token

/backups/getBackupList():

301	failed to get list of backups
302	no backups exist for user

/backups/startUpload():

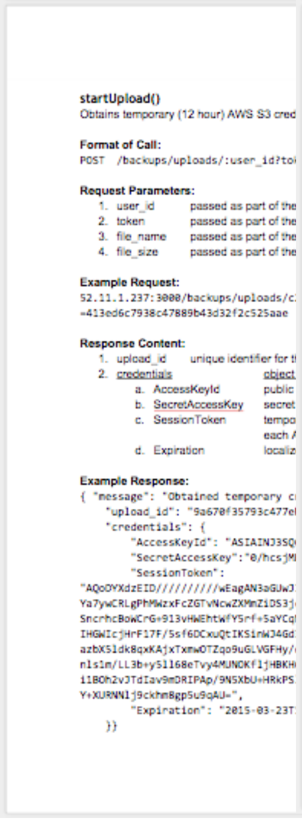
303	unable to get temporary S3 credentials

/backups/completeUpload():

304	failed to complete upload
305	no upload found
306	failed to get backup information
307	could not verify backup

/backups/deleteBackup():

308	failed to obtain file_name to delete
309	no backup to delete
310	failed to delete backup



startUpload()
Obtains temporary (12 hour) AWS S3 cred

Format of Call:
```
POST   /backups/uploads/:user_id?to
```

Request Parameters:

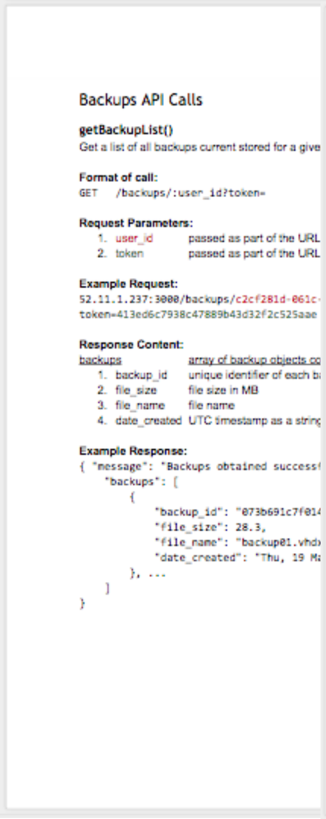
1. user_id   passed as part of the
2. token   passed as part of the
3. file_name   passed as part of the
4. file_size   passed as part of the

Example Request:
```
52.11.1.237:3000/backups/uploads/c
=413ed6c7938c47889b43d32f2c525aae
```

Response Content:

1. upload_id   unique identifier for t
2. credentials   object
    a. AccessKeyId   public
    b. SecretAccessKey   secret
    c. SessionToken   tempo
       each A
    d. Expiration   localiz

Example Response:
```
{ "message": "Obtained temporary c
    "upload_id": "9a670f35793c477e
    "credentials": {
        "AccessKeyId": "ASIAINJ3SQ
        "SecretAccessKey":"0/hcsjM
        "SessionToken":
"AQoDYXdzEID//////////wEagAN3aGUwJ
Ya7ywCRLgPhMWzxFcZGTvNcwZXMnZiDS3j
SncrhcBowCrG+913vHWEhtWfY5rf+5aYCq
IHGWIcjHrF17F/5sf6DCxuQtIKSinWJ4Gd
azbX5ldk8qxKAjxTxmwOTZqo9uGLVGFHy/
nls1m/LL3b+y5ll68eTvy4MUNOKfljHBKH
i1BOh2vJTdIav9mDRIPAp/9N5XbU+HRkPS
Y+XURNNlj9ckhm8gp5u9qAU=",
        "Expiration": "2015-03-23T
    }}
```



Backups API Calls

getBackupList()
Get a list of all backups current stored for a give

Format of call:
```
GET    /backups/:user_id?token=
```

Request Parameters:

1. user_id   passed as part of the URL
2. token   passed as part of the URL

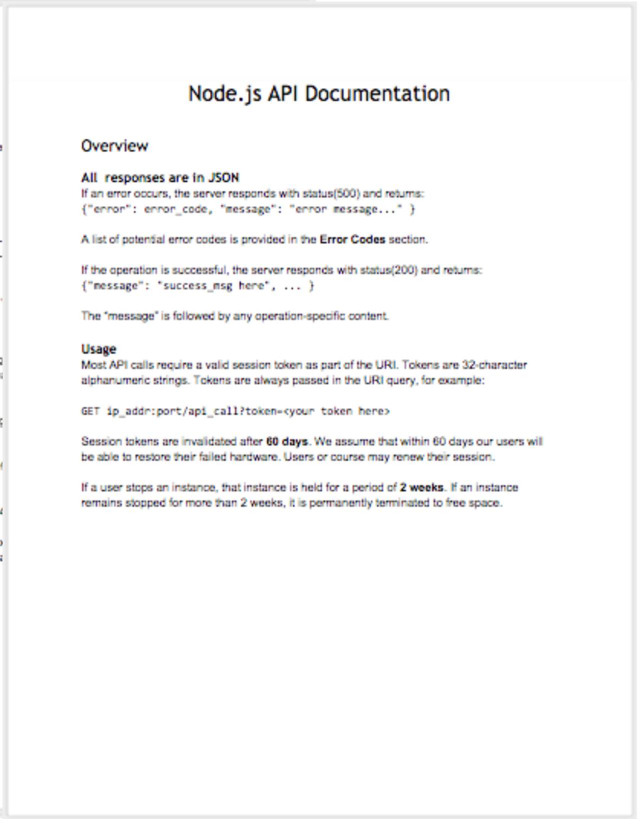
Example Request:
```
52.11.1.237:3000/backups/c2cf281d-061c-
token=413ed6c7938c47889b43d32f2c525aae
```

Response Content:

backups   array of backup objects co

1. backup_id   unique identifier of each b
2. file_size   file size in MB
3. file_name   file name
4. date_created   UTC timestamp as a string

Example Response:
```
{ "message": "Backups obtained success
    "backups": [
        {
            "backup_id": "073b691c7f014
            "file_size": 28.3,
            "file_name": "backup01.vhd
            "date_created": "Thu, 19 Ma
        }, ...
    ]
}
```



Node.js API Documentation

Overview

All responses are in JSON
If an error occurs, the server responds with status(500) and returns:
```
{"error": error_code, "message": "error message..." }
```

A list of potential error codes is provided in the **Error Codes** section.

If the operation is successful, the server responds with status(200) and returns:
```
{"message": "success_msg here", ... }
```

The "message" is followed by any operation-specific content.

Usage
Most API calls require a valid session token as part of the URI. Tokens are 32-character alphanumeric strings. Tokens are always passed in the URI query, for example:

```
GET ip_addr:port/api_call?token=<your token here>
```

Session tokens are invalidated after **60 days**. We assume that within 60 days our users will be able to restore their failed hardware. Users or course may renew their session.

If a user stops an instance, that instance is held for a period of **2 weeks**. If an instance remains stopped for more than 2 weeks, it is permanently terminated to free space.

API Client Library

- HttpMethods.cs and RecloApiCaller.cs
- All calls to API are asynchronous
- Finished functions for all backup API calls
- Tested all functions except for DELETE
- mc.sh
- Simplified and documented

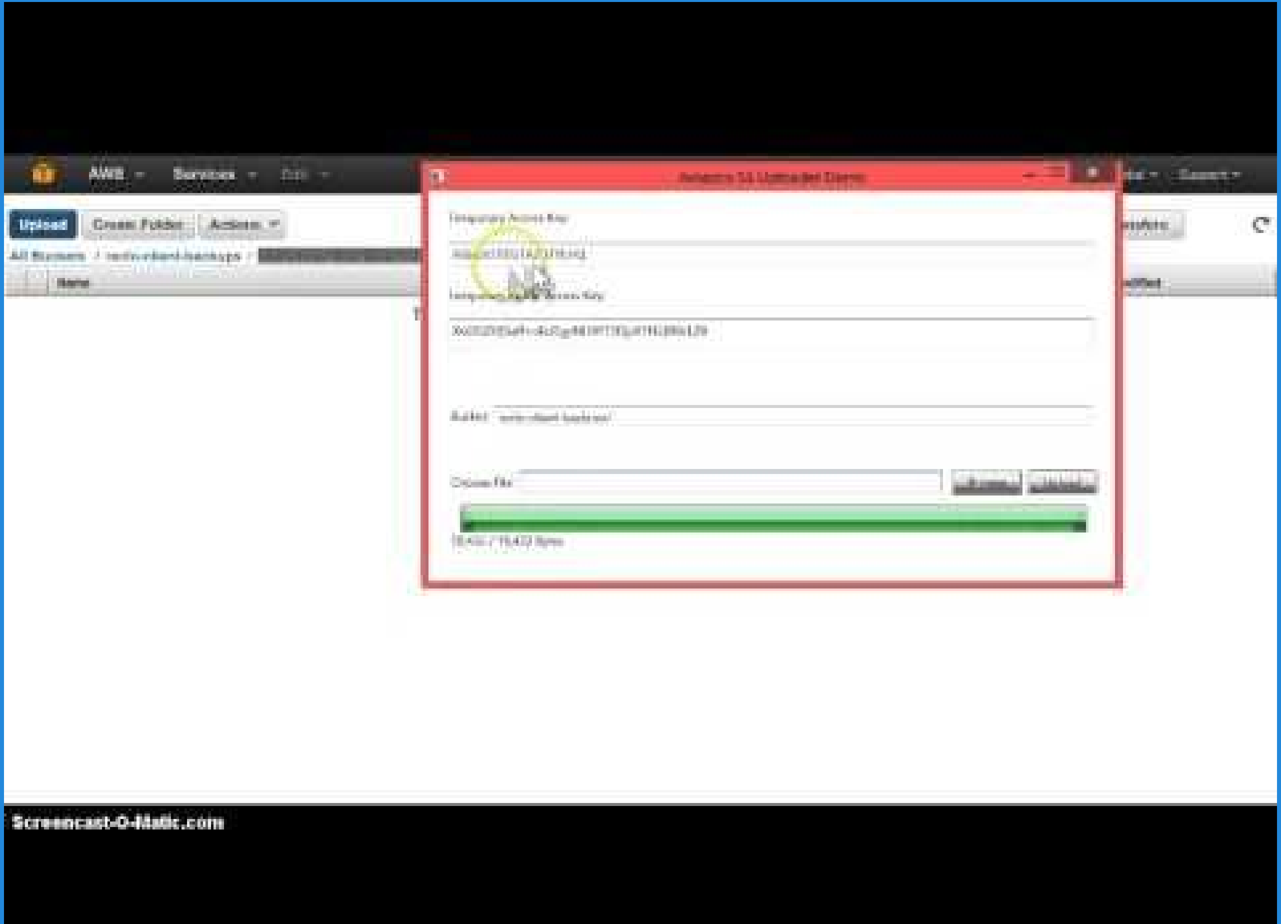
S3 Upload

- Temporary Access token to user
- Upload is done from the C# client using AWS SDK
- Upload is asynchronous
- Needs to be merged with GUI

  and api client code

Amazon S3

S3 Upload Video Demo



A Perspective

Language	Lines of Code Written
Node.js	1610
C#	779
HTML & CSS	220
TOTAL	**2609**

2609 lines of ...

1. Debugged
2. Running
3. Tested

… code written!

Next Sprint: Goals

- Upload backups to S3
- Start new EC2 instances from vhd backups
- VPN Connection to restored instances
- Finish Backup Manager back-end
- Recovery Manager back-end