# Data Corruption Recovery

Project Documentation Submitted

To the Faculty of School of

Computing and Information Technologies

Of

Asia Pacific College

In Partial Fulfillment of the Requirements for the subject

Applied Projects 2 or Software Development

By

Alberca , Reginald John Steven

Miculob, Kent Michael

Laureta, Maria Letty

Table of Contents

[Project Title 1](#_Toc454543889)

[Abstract 3](#_Toc454543890)

[List of Figures, List of Tables, List of Notations 4](#_Toc454543891)

[I. Introduction 5](#_Toc454543892)

[1.1 Project Context 5](#_Toc454543893)

[1.2 Purpose and Description 5](#_Toc454543894)

[1.3 Objectives 5](#_Toc454543895)

[1.4 Scope and Limitations 5](#_Toc454543896)

[II. Related Literature 5](#_Toc454543897)

[III. Technical Background 5](#_Toc454543898)

[3.1 Include in-depth discussion on relevant technical aspects of this project 5](#_Toc454543899)

[IV. Design and Methodology 5](#_Toc454543900)

[4.1 Include discussions on conceptual design/system architecture/block diagrams and algorithms 5](#_Toc454543901)

[V. Results and Discussion 5](#_Toc454543902)

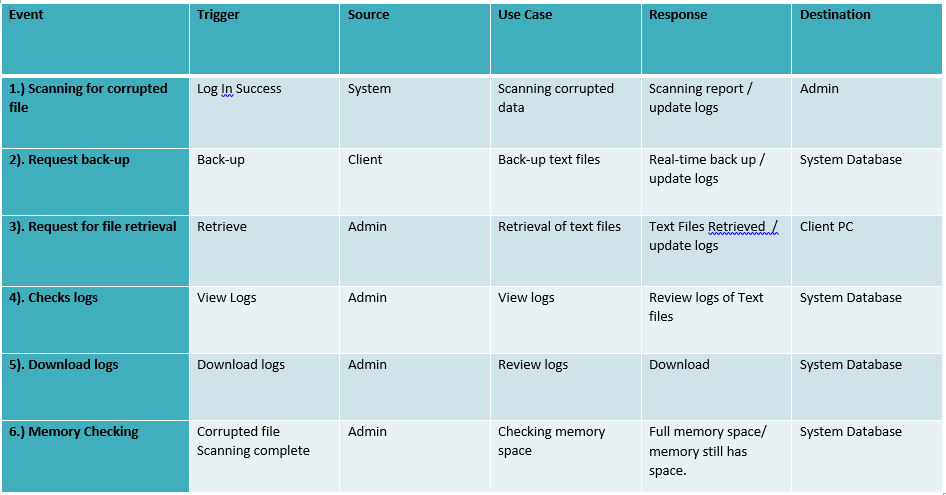
[VI. Conclusions and Recommendations 5](#_Toc454543903)

[VII. Appendices 5](#_Toc454543904)

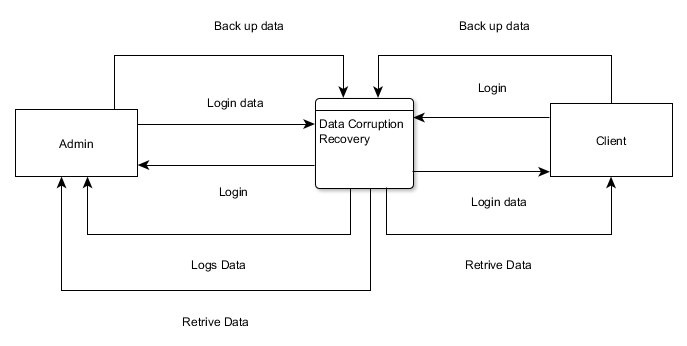
# Abstract

# List of Figures, List of Tables, List of Notations

Event Table

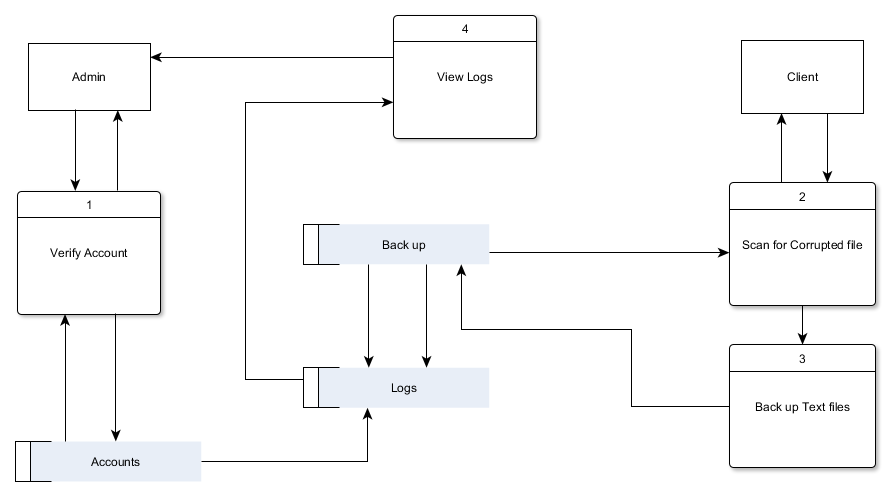


Context Diagram

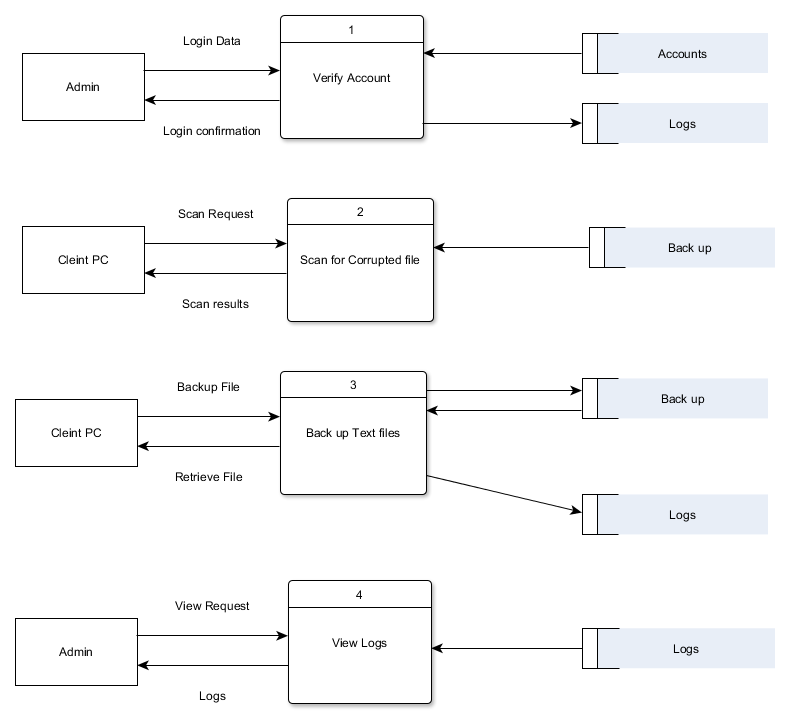


Data Flow Diagram

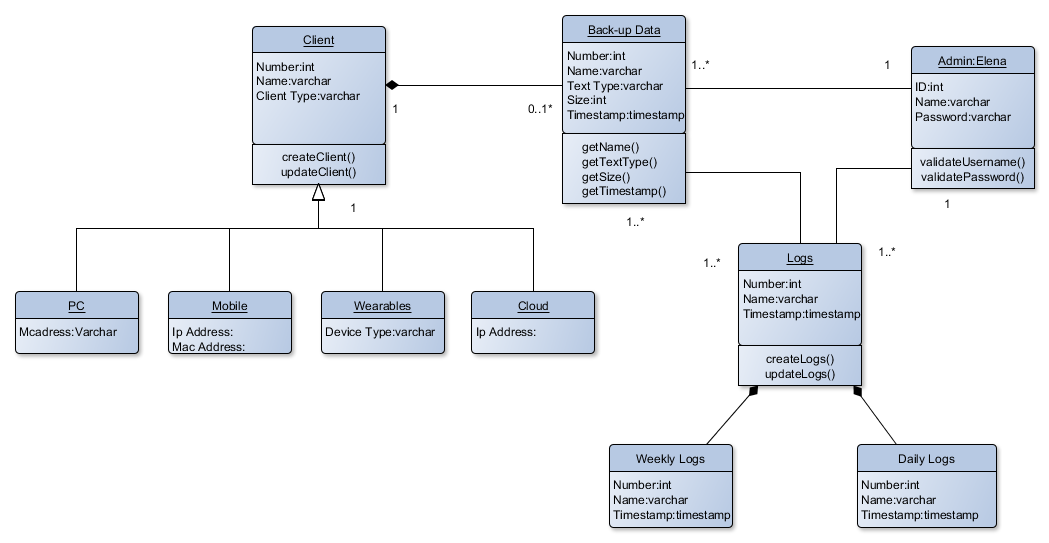
Leveled 0



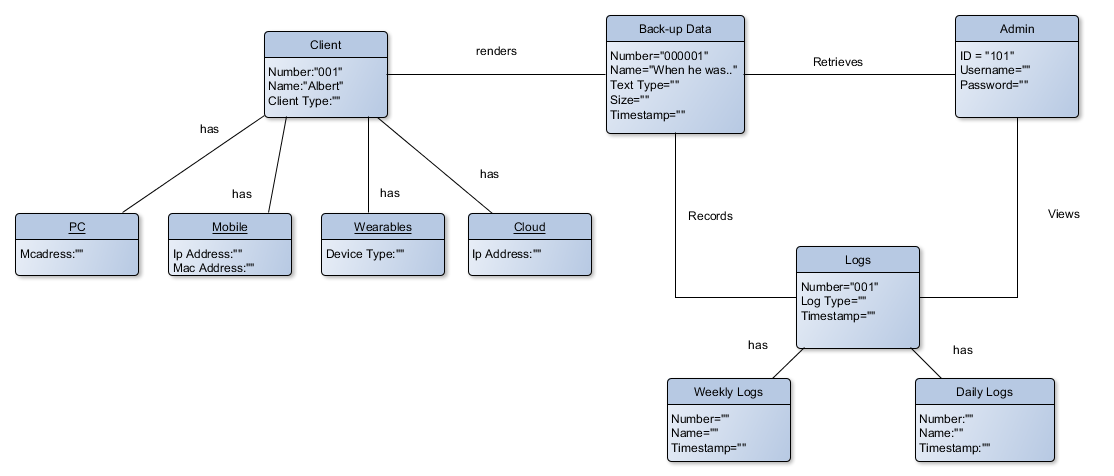
DFD Fragments



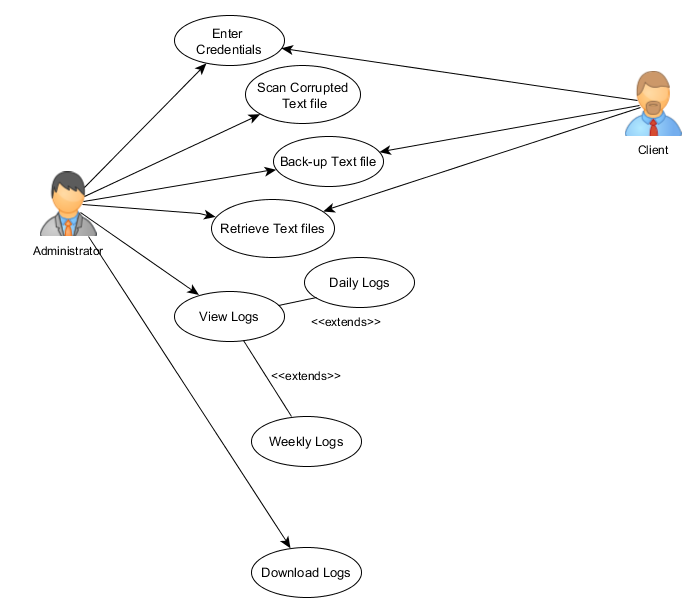
Class Diagram

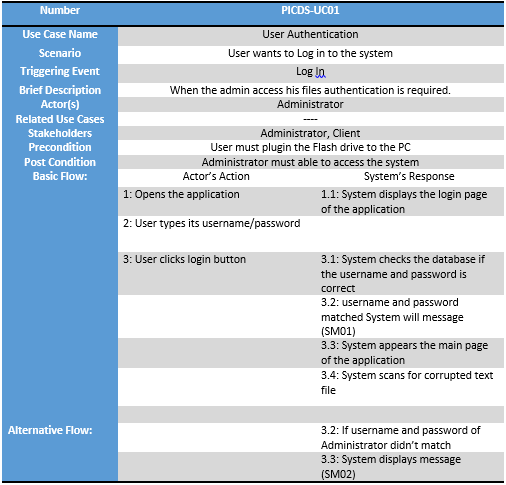


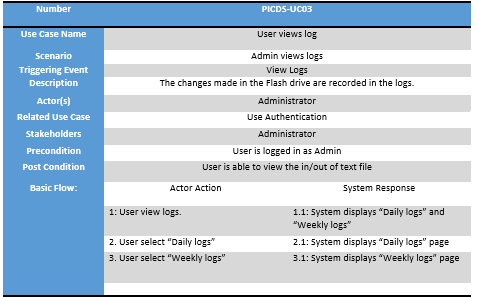
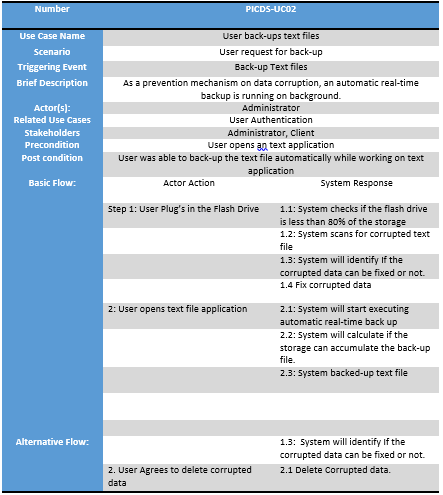
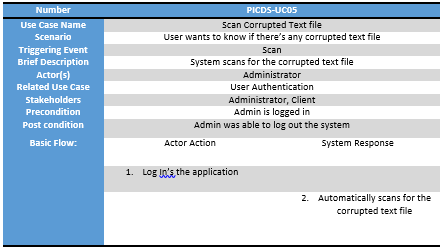
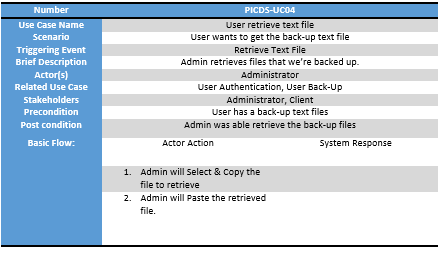
Object Diagram

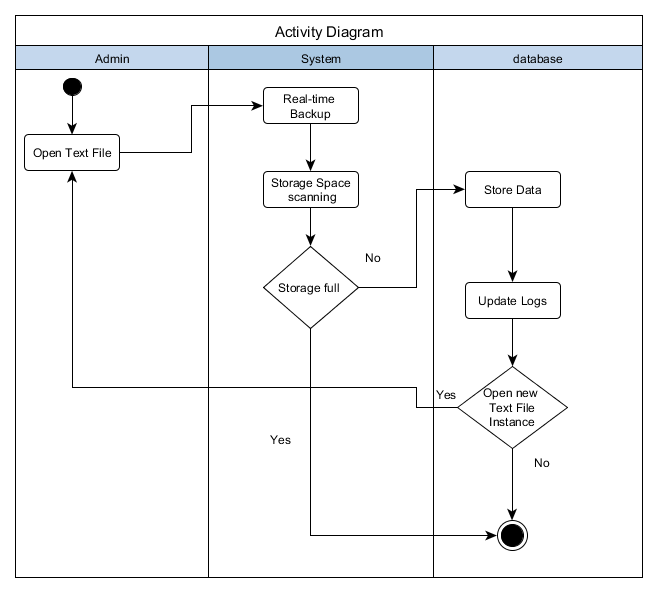
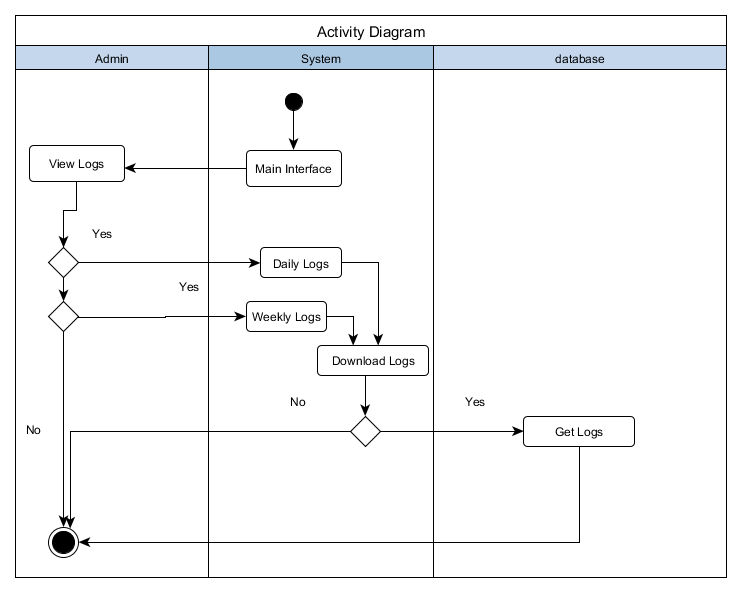
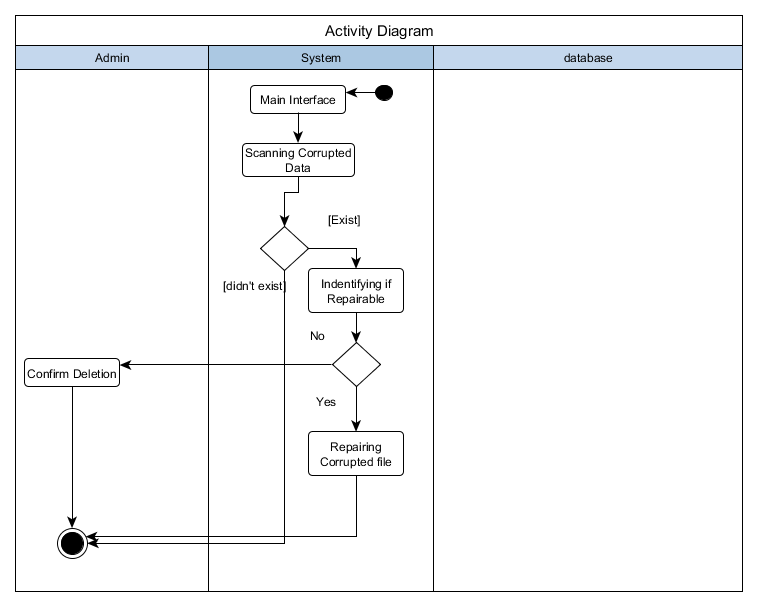


Use Case Diagram

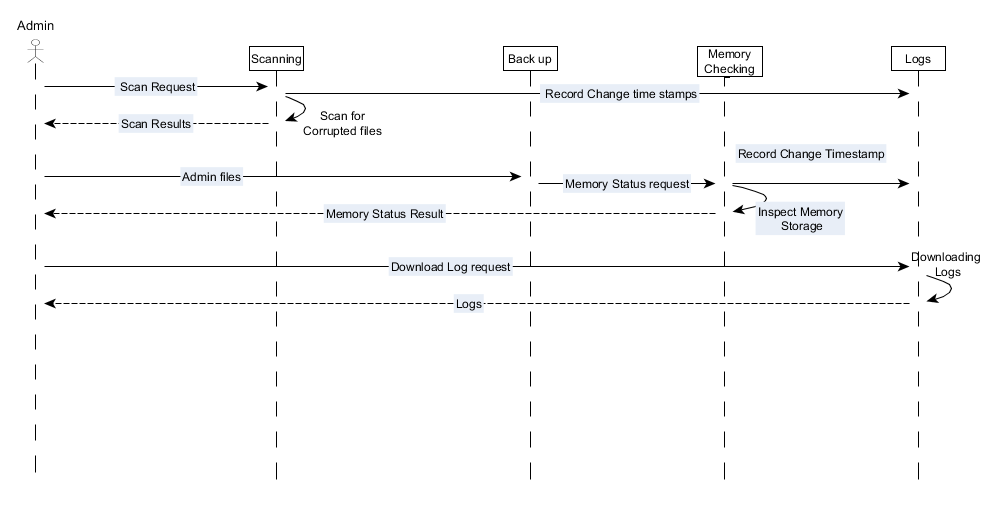


Use Case Full Description  


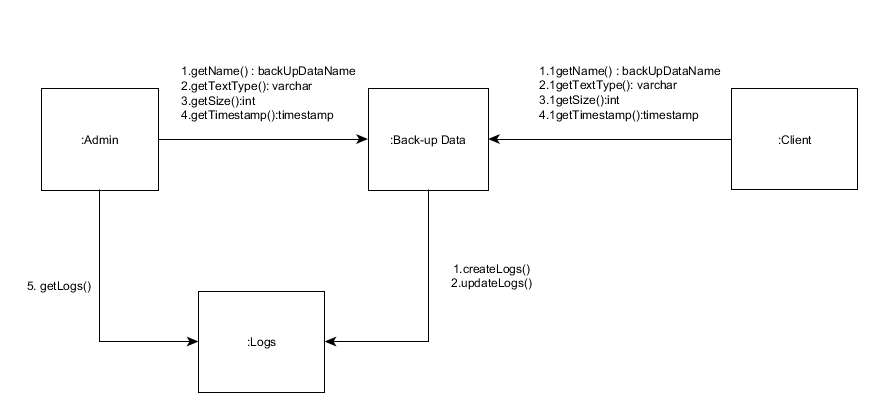
  


Activity Diagram  


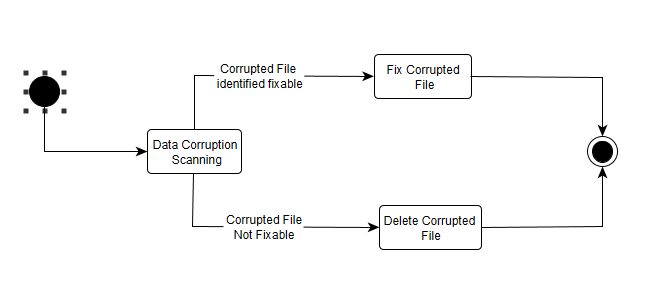
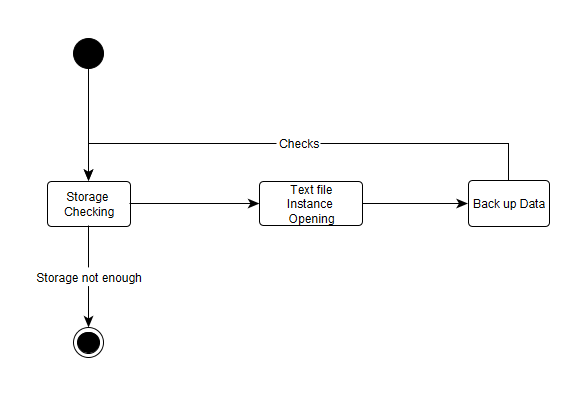
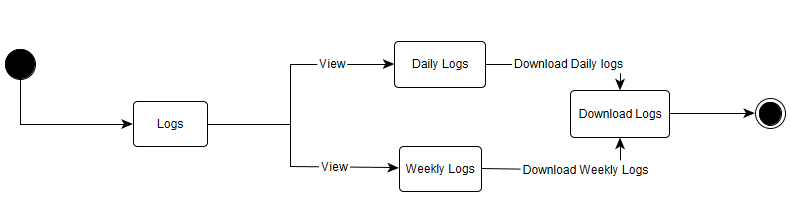
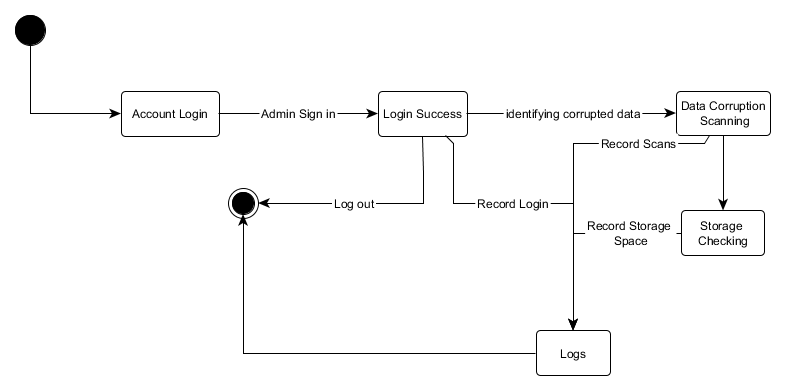
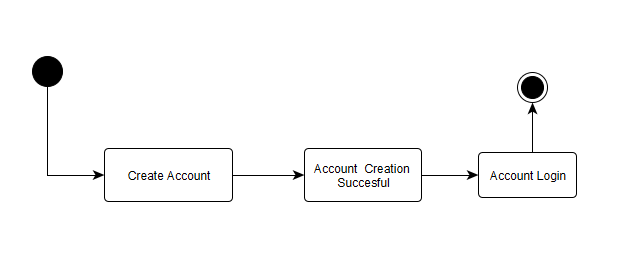
Sequence Diagram



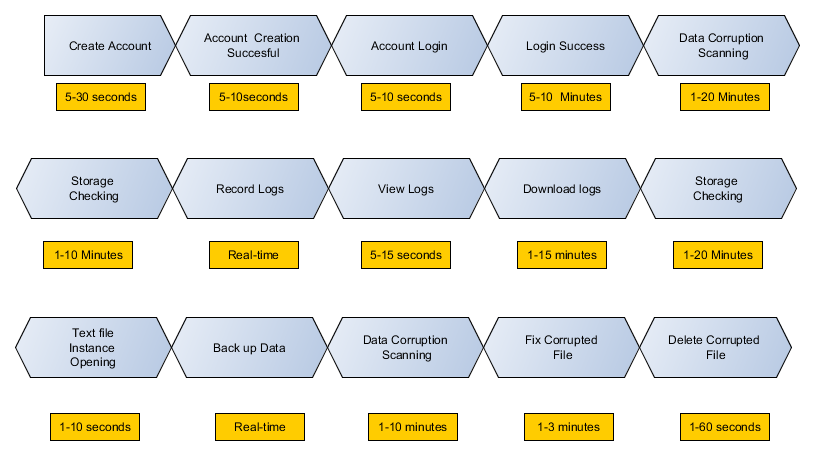
Communication Diagram



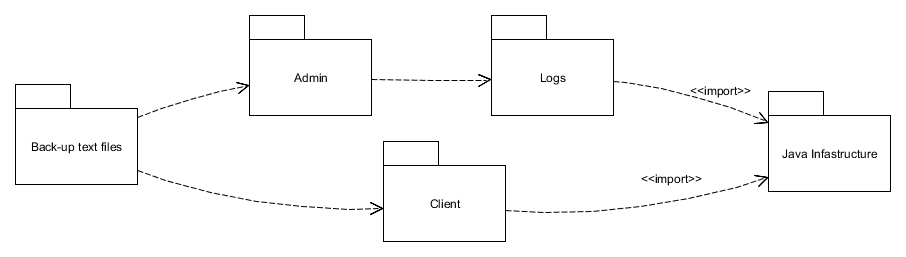
State Diagram



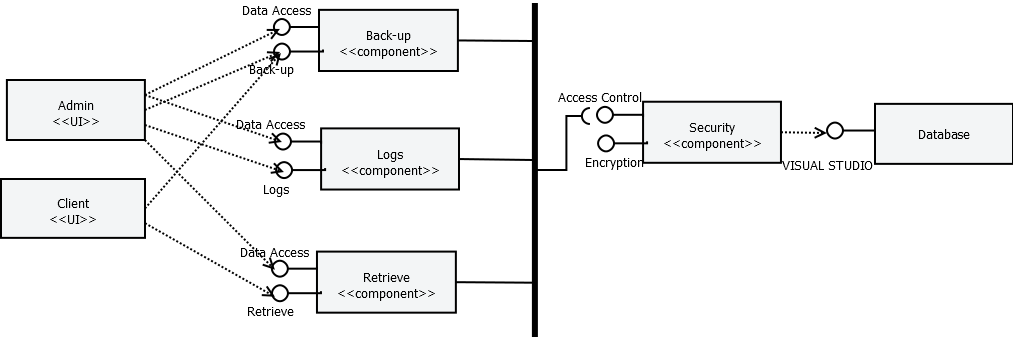
Timing Diagram



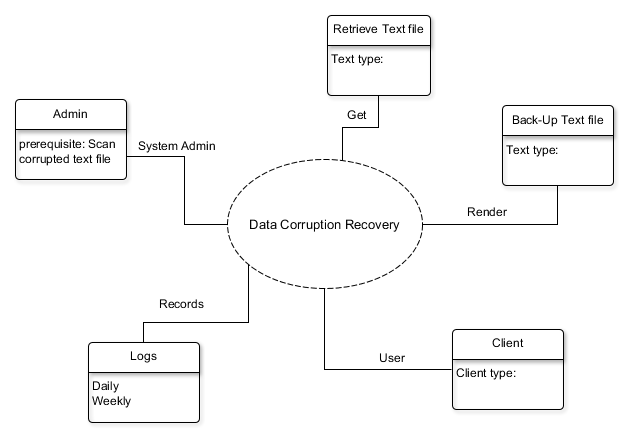
Package Diagram



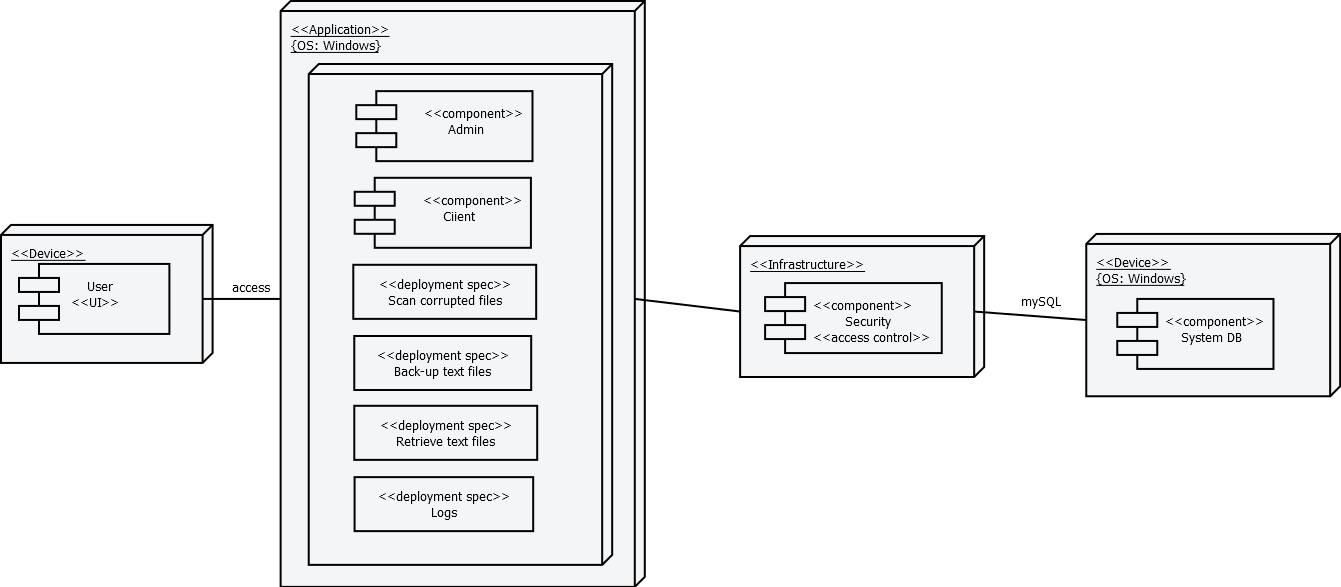
Component Diagram



Composite Structure Diagram



Deployment Diagram



Interaction Overview Diagram

# Introduction

## Project Context

Data corruption has always been a problem in computing. With the imminent implementation of internet of things, solutions on data corruption must be address to avoid future complications regarding data loss.

## Purpose and Description

People are expose to different security issues that revolves around the Security, Integrity and accessibility. The main essence of this research is to address availability issues that concerns data corruption. And to produce a tool capable of preventing the occurrence of corrupted data that can be easily used by non-IT people.

## Objectives

General Objectives:

* + - * This Project aims to create a security algorithm that could save critical information from data loss.
* Specific Objectives:
  + - * To be able to provide an algorithm that could save someone’s information.
      * To be able to give confidentiality control on the user’s information.
      * To be able to create a low-learning curve security algorithm that is applicable to most people.

## Scope and Limitations

This Project aims to help students and/or office-workers to secure their information or data on availability issues, whom are using windows based OS- 7 and higher. The project primarily focuses on data corruption and prevention. It also assumes that the interaction are just between the players of the system such as the admin, system, and user interactions and no other irrelevant factors are included.

# Related Literature

**Causes of Data Corruption**

The common causes of corrupted data is power outages or other related to power failure, Inappropriate shutdown caused by the user, hardware problems or failures such as hard drive failure, bad sectors, and etc. another reason is software failures caused by bad programming, particularly if it results in either hard restarts or data that is saved incorrectly. Any of these causes can result in a corrupted hard drive directory. A corrupted hard drive directory can cause files to apparently "go missing" and lead to further data loss or corruption. (Wikipedia : Data Corruption, n.d.), (Smith, 2014)

**Data Corruption**

Some file systems, such as Btrfs, HAMMER, ReFS, and ZFS, use internal data and metadata check summing to detect silent data corruption. In addition, if a corruption is detected and the file system uses integrated RAID mechanisms that provide data redundancy, such file systems can also reconstruct corrupted data in a transparent way. This approach allows improved data integrity protection covering the entire data paths, which is usually known as end-to-end data protection, compared with other data integrity approaches that do not span different layers in the storage stack and allow data corruption to occur while the data passes boundaries between the different layers. (Wikipedia : Data Corruption, n.d.)

**Checksum**

A checksum is a count of the number of bits in a transmission unit that is included with the unit so that the receiver can check to see whether the same number of bits arrived. If the counts match, it's assumed that the complete transmission was received. Both TCP and UDP communication layers provide a checksum count and verification as one of their services. (Search Security, n.d.)

**File System**

A file system is a part of the operating system that determines how files are named, stored, and organized on a volume. A file system manages files and folders, and the information needed to locate and access these items by local and remote users. Microsoft Windows Server 2003 supports both the FAT and NTFS file systems. NTFS allows you to gain the maximum benefits for the needs of today’s enterprise business environments from Windows Server 2003, such as increased security, more robust and reliable performance, as well as a design for greater storage growth, features not found in FAT. (How to Repair Corrupted Windows System Files with the SFC and DISM Commands, n.d.)

**RAID Technology**

With RAID technology, data can be mirrored on one or more disks in the same array, so that if one disk fails, the data is preserved. RAID also offers the option of reading or writing to more than one disk at the same time in order to improve performance. In this arrangement, sequential data is broken into segments which are sent to the various disks in the array, speeding up throughput. A typical RAID array uses multiple disks that appear to be a single device so it can provide more storage capacity than a single disk. (Beal)

**Assembly Language**

In implementing the corrupted data retrieval we can use the Assembly language. It is a low leveled programming language for a computer, or other programmable device specific to a particular computer architecture meaning you can communicate through the device one on one. Each assembly language is specific to a particular computer architecture, in contrast to higher level of programming which are generally portable across multiple architectures, but require interpreting or compiling. Assembly language may also be called symbolic machine code. Assembly language also use mnemonic to show a list of each low leveled machine instruction or operation. Many operation takes one or more operands to be able to complete instruction. Assembly language is converted into executable machine code by a utility program referred to as an assembler like NASM, MASM, etc. (Assembly Language, n.d.) , (Assembly Programming Tutorial, n.d.)

**An Analysis of Data Corruption in the Storage Stack**

An important threat to reliable storage of data is silent data corruption. In order to develop suitable protection mechanisms against data corruption, it is essential to understand its characteristics. In this paper, we present the ﬁrst large-scale study of data corruption. We analyze corruption instances recorded in production storage systems containing a total of 1.53 million disk drives, over a period of 41 months. We study three classes of corruption: checksum mismatches, identity discrepancies, and parity inconsistencies. We focus on checksum mismatches since they occur the most.

We found more than 400,000 instances of checksum mismatches over the 41-month period. We ﬁnd many interesting trends among these instances including: (i) near line disks (and their adapters) develop checksum mismatches an order of magnitude more often than enterprise class disk drives, (ii) checksum mismatches within the same disk are not independent events and they show high spatial and temporal locality, and (iii) checksum mismatches across different disks in the same storage system are not independent. We use our observations to derive lessons for corruption-proof system design. (Bairavasundaram, Arpaci-Dusseau, Arpaci-Dusseau, Goodson, & Schroeder, 2008)

**Method and apparatus for recovering data from damaged or corrupted file storage media**

An automated method and apparatus for identifying and copying lost files from a mass data storage device of a computer when file system information (as opposed to the actual data files) stored on the mass data storage device has been corrupted or destroyed. The mass data storage device is scanned on a sector-by-sector basis in order to attempt to identify sectors containing file system data structures and file attributes. Identification is made using data signature and/or pattern matching filters. The location of, and any valid information found in, such sectors is used then to derive information useful in locating files to be copied to another storage device. For example, in a FAT, NTFS or other cluster-oriented file system, if information on the number of sectors per cluster (SPB) is not available from a boot directory, it and a cluster base (the starting sector of cluster 0) are calculated using the physical location of the beginning sectors of the directories or folders. When a starting cluster is known from a directory entry, but not additional file allocation information, a cluster chain may be reconstructed utilizing one or more of several disclosed methods.

**Systems and methods for automated maintenance and repair of database and file systems**

The present invention relates generally to database and file system management and, more particularly, to automatic database and file system maintenance and repair to ensure data reliability. Various aspects of the present invention relate to responding and correcting data corruptions at a data page level for all data page types, as well as to recovery (including rebuild or restore operations) for various scenarios including, without limitation, index page corruptions (clustered and non-clustered), data page corruptions, and page corruptions in the log file.

**System and method for detecting correcting and discarding corrupted data packets in a cable data delivery system**

A method and apparatus for detecting errors and improving quality in real-time data transmissions is provided. In one embodiment, the packet header checksum field is turned off to allow uninterrupted transmission of data packet payloads. A checksum added to each independent data segment in the datagram payload permits each data packet to be examined separately, resulting in improved transmission quality.

**VALIDATING POPULATION VIABILITY ANALYSIS FOR CORRUPTED DATA SETS**

Diffusion approximation (DA) methods provide a powerful tool for population viability analysis (PVA) using simple time series of population counts. These methods have a strong theoretical foundation based on stochastic age-structured models, but their application to data with high sampling error or age-structure cycles has been problematic. Recently, a new method was developed for estimating DA parameters from highly corrupted time series. We conducted an extensive cross-validation of this new method using 189 long-term time series of salmon counts with very high sampling error and non-stable age-structure fluctuations. Parameters were estimated from one segment of a time series, and a subsequent segment was used to evaluate the predictions regarding the risk of crossing population thresholds. We also tested the theoretical distributions of the estimated parameters. The distribution of parameter estimates is an essential aspect of a PVA because it allows one to calculate confidence levels for risk metrics. This study is the first data-based cross-validation of these theoretical distributions. Our cross-validation analyses found that, when parameterization methods designed for corrupted data sets are used, DA predictions are very robust even for problematic data. Estimates of the probability of crossing population thresholds were unbiased, and the estimated parameters closely followed the expected theoretical distributions.

# Technical Background

## Include in-depth discussion on relevant technical aspects of this project

# Design and Methodology

## Include discussions on conceptual design/system architecture/block diagrams and algorithms

# Results and Discussion

# Conclusions and Recommendations

# Appendices

- Relevant Source Code

- Evaluation Tool or Test Document

- Sample Input / Output Reports

- User Guides

- One-page Curriculum Vitae per team member