**Uncommon Solutions**

**Group 3**

**Design Document**

**UNCOMMON SOLUTIONS DESIGN DOCUMENT**

**Prepared By**

|  |  |
| --- | --- |
| Document Owner(s) | Project Role |
| Hither Guzha | Technical Writer |

**Requirements Version Control**

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Date | Author | Change Description |
| 1.0 | 11/16/2019 | Hither Guzha | Document creation |

**Summary Details**

|  |  |
| --- | --- |
| Participants | Name(s) |
| Project Manager: | Michael Kiefer |
| Project Team: | Hither Guzha – Technical Writer  Andrew Benson - Software Engineer  Donn Eddy - UX/HCI  Sean Mooneyham - Integration Engineer  Chase Thorpe - Test Engineer |
| End Users: | HR Departments |
| Description w/ Goal: | The purpose of this project is the implementation of an HR database and front end for personnel tracking. This document is to detail the design and architecture of the Uncommon Solutions Application. |

[INTRODUCTION 4](#_Toc24810567)

[1.1 Purpose 4](#_Toc24810568)

[1.2 Background 4](#_Toc24810569)

[1.3 Scope 4](#_Toc24810570)

[1.4 Assumptions 4](#_Toc24810571)

[1.5 Constraints 4](#_Toc24810572)

[1.6 Risks 4](#_Toc24810573)

[1.7 Design Considerations 4](#_Toc24810574)

[2 SYSTEM OVERVIEW 5](#_Toc24810575)

[2.1 System Design 5](#_Toc24810576)

[3 SYSTEM ARCHITECTURE DESIGN 6](#_Toc24810577)

[3.1 Database Design 6](#_Toc24810578)

[3.2 Class Design 6](#_Toc24810579)

[3.3 User Interface 6](#_Toc24810580)

[4 SYSTEM INTERGRITY CONTROLS 7](#_Toc24810581)

[5 APPROVALS 9](#_Toc24810582)

# 

# INTRODUCTION

## Purpose

The purpose of this document is to detail the design and architecture of the Uncommon Solutions HR Management System. The Uncommon Solutions HR Management software is being developed using an Agile SDLC framework. While this document describes the design aspects of certain features within the Uncommon Solutions HR Management System, specific details about the requirements and design considerations that were made to meet those requirements can be found in the system requirement specifications document. This document contains the necessary information required to effectively define the architecture and system design to give the project team clearly defined guidance on the architecture of the system to be developed.

## Background

The Uncommon Solutions HR Management System will be designed in a way that makes it easy to support multiple platforms such as Windows, macOS, iOS and Android. This web-based tool provides a direct method for storing and providing access to individual personnel records, and for all processes required for HR tracking and data aggregation requirements. The HR system will be implemented using AWS Elastic Compute Cloud (EC2) and Amazon’s Relational Database Service (RDS) in order to allow for universal deployability and access.

## Scope

This document describes the overarching design and guiding principles of the Uncommon Solutions HR Management System. This is a living document and will be updated as changes are made to the HR Management Systems design.

## Assumptions

The following assumptions are relevant to the design of the proposed system:

* The proposed new system will leverage the Uncommon Solutions HR architecture.
* The existing architecture and system design will be used including all existing components and sub-systems.
* It is assumed that additional functionality will be added to the proposed solution as required during development and testing.

## Constraints

* There are no hardware or software technical constraints identified with this project.
* System interoperability may be a constraint since the design will leverage free tier AWS EC2 instance and RDS with the potential to expand to paid utilization at a larger-scale fielding.

## Risks

There are very minimal risks associated with the system design.  This is primarily because the existing system design and architecture will not be modified to meet the needs of the proposed solution. Ongoing maintenance of the system will also be a concern.

## Design Considerations

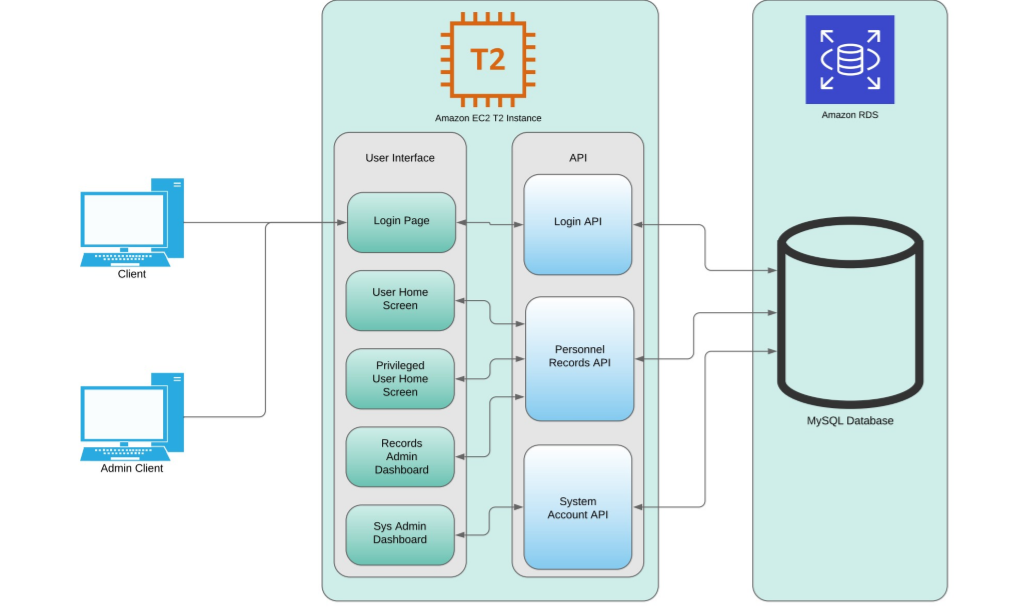
The major design considerations for the proposed solution are related to system performance and scalability of the solution. The data center is hosted in AWS which provides a tremendous amount of flexibility in terms of scaling for performance and storage requirements. Processor speed, memory, peripherals, and stakeholder support will be factored in the design.

# SYSTEM OVERVIEW

## System Design

The product system design utilizes two different AWS services, Elastic Compute Cloud (EC2) and Amazon’s Relational Database Service (RDS). Using AWS as a platform for fielding the application is the most cost-effective way to host the application while also providing access to the application for all project team members and product owners. AWS is an excellent platform for quickly spinning up application prototypes and rapidly deploying solutions to multiple customers.

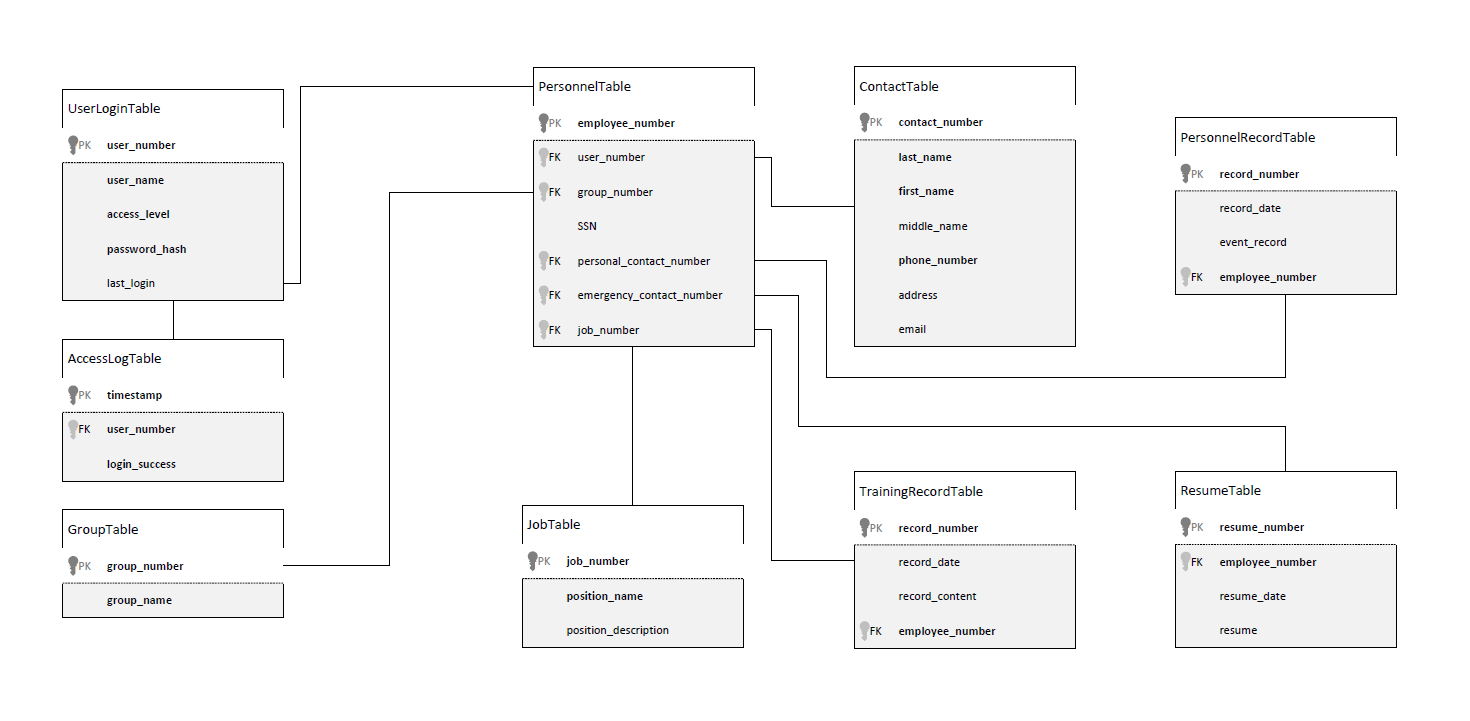
The application uses Apache version 2.4.29 as a web server and is installed on the EC2 T2 instance and uses Ubuntu 18.04 as an operating system. All source code including the user interface and API will be on the apache server. The application uses a MySQL database installed on an RDS instance to house all the application data. To access the application the client or admin client simply connects to the EC2 instance currently located at “**ec2-54-145-217-172.compute-1.amazonaws.com**” OR “**54.145.217.172**”.



**Figure 1: The System Design Diagram**

# SYSTEM ARCHITECTURE DESIGN

## Database Design



**Figure 2: The Data Design Diagram (DDD)**

The database design for this project is divided into two main sections; authentication data and personnel data. There is a relational connection between the two sections for most accounts, except those accounts for system administrators. The Data Design Diagram (DDD) provides a visual of the tables and their relationships, but a basic breakdown is described in the following sections.

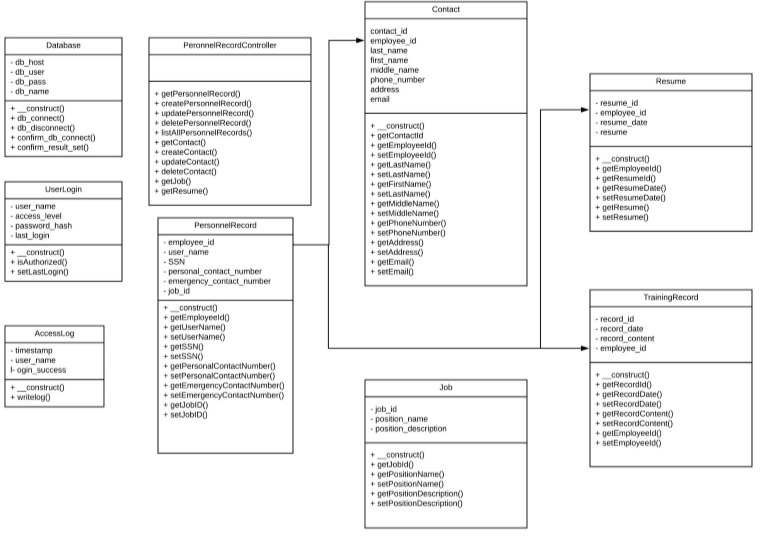
The authentication data section is simple, containing the table with information on the individual user accounts and a linked table to log authentication attempts in. The presence of the AccessLogTable allows for database level tracking of frequency of failed login attempts as well as an audit log of all successful system logins. At this time, this is the only database level logging implemented for this program, though future revisions may incorporate this capability to all data tables for record-level audit tracking. For all users who are not system administrator accounts, the table entry in UserLoginTable will link to an entry in PersonnelTable in a one-to one relationship.

The personnel data section is a little more complicated than that of the authentication data, but only to allow for more flexibility in data storage for individual records. In much the same way, as personnel change jobs, the jobs themselves are still present and can be reassigned to a new personnel record to reflect the change in who is occupying that position without requiring the re-entry of the job title or job description. As individuals progress within the company, the number of training records, personnel records, and resumes on file can shift and grow with the individual, in order to effectively capture all their experience and capabilities. The same table is used to store contact information for both the employees and their emergency contacts, as we will have instances where an emergency contact is another employee of the company, and there’s no need to duplicate data in those cases.

Overall, it’s a simple database that takes advantage of the capabilities of the relational model to streamline data storage and avoid space allocation for empty records, while still allowing the flexibility for a large-scale employee record in the cases that warrant.

## Class Design

The design of the API uses a simple MVC pattern where the models use an Object-Oriented design and have standard attributes such as private properties that are accessible with getter and setter methods. Since the scope of the application is small only one controller will be used to handle all the database operations. Classes will be implemented to handle database connectivity, user login, and audit logging.



**Figure 3: The Class UML Diagram**

## User Interface

The User Interface for the will adopt a flat design. Not only would this provide the design with a modern aesthetic, but would also minimize the system resources consumed to render the web pages expanding the compatibile range of platforms. The design will be seamless between pages to ensure that there are no sudden shifts in design. Refer to the Appendix for page wireframes.

The User Interface presented after completion of the login process will depend on the level of access associated with the individual user account. The interface presented to a standard user will vary significantly from that presented to a system administrator as outlined in the wireframes presented in the appendix.

# SYSTEM INTERGRITY CONTROLS

The following security and integrity controls are relevant to the design of the proposed system:

* Source code for the HR software components and products will be stored securely with need-to-know access controls applied.
* The system will have log files for all modification that will be maintained and preserved for future analysis.
* User passwords will be encrypted in the database utilizing a secure hashing algorithm.
* Internal security will be implemented to restrict access of critical data items to only Data Administrators and users with the required access levels.
* Each employee will be restricted to only access their personnel data unless they have higher level accesses.
* Audit procedures will be implemented to meet control, reporting, and retention period requirements for operational and management reports
* Verification processes for additions, deletions, or updates of critical data
* The system will have the ability to identify all audit information by user identification, network terminal identification, date, time, and data accessed or changed.

# APPROVALS

I have read the above Project Plan and will abide by its terms and conditions and pledge my full commitment and support for the Project Plan.

Sign-off Sheet

**Prepared by** \_\_\_Hither Guzha\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Technical Writer – Hither Guzha

**Approved by**\_\_\_Michael Kiefer\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Project Manager– Michael Kiefer

**Approved by**\_\_\_Andrew Benson\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Project Team Member– Andrew Benson

**Approved by**\_\_\_Donn Eddy\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Project Team Member– Donn Eddy

**Approved by**\_\_\_Sean Mooneyham\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Project Team Member– Sean Mooneyham

**Approved by**\_\_\_Chase Thorpe\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Project Team Member– Chase Thorpe

**References:**

Maryland GOV. Waterfall Templates. Retrieved from: <https://doit.maryland.gov/SDLC/Pages/Templates.aspx>

SDLC Forms. What is the Waterfall Software Development Life Cycle (SDLC)? Retrieved from: <https://www.sdlcforms.com/UnderstandingSDLC.html>

Alexsoft. Technical Documentation in Software Development: Types, Best Practices, and Tools. Retrieved from: <https://www.altexsoft.com/blog/business/technical-documentation-in-software-development-types-best-practices-and-tools/>

Agile Modeling. Core Practices for Agile/Lean Documentation. Retrieved from: <http://www.agilemodeling.com/essays/agileDocumentationBestPractices.htm>

Dreamcatcher. A Practical Approach for Managing Agile Technical Design. Retrieved from: <http://dreamcatchersoftware.com/agile-technical-design/>

**Appendix:**

Login Page

Username:

Password:

Uncommon Solutions HR Management System

Login

LOGO

*Access of this system constitutes consent to monitoring*

User Home Screen

LOGO

Logout

Edit Record

Home

Welcome, ${Username}

**Your Record**

*Table populated with user details: name, address, contact numbers,*

*job identifier, etc*

Current Role: Level 1 User

*Table populated with user details: name, address, contact numbers,*

*job identifier, etc*

LOGO

Logout

Edit Record

Home

Welcome, ${Username}

**Your Record**

View Users

Current Role: Level 2 User

Privileged User Home Screen

LOGO

Logout

Home

Welcome, ${Username}

**Users Table**

Current Role: Records Admin

User name

User name

User name

Contact

Contact

Contact

Job Info

Job Info

Job Info

Edit User

Edit User

Edit User

Delete User

Delete User

Delete User

Records Admin Dashboard

Sys Admin Dashboard

LOGO

Logout

Home

Welcome, ${Username}

**Users Table**

Current Role: System Admin

User name

User name

User name

Contact

Contact

Contact

Job Info

Job Info

Job Info

Create New User

+