*Work Request Application*

Product Design Specification

Version *1.1*

*Dec 12 2021*

VERSION HISTORY

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Version #** | **Implemented**  **By** | **Revision**  **Date** | **Approved**  **By** | **Approval**  **Date** | **Reason** |
| 1.0 | Ian Oliver | Nov 11 2021 | Dave Leake | Nov 13 2021 | Initial Design Definition draft |
| 1.1 | Ian Oliver | Dec 12 2021 | Dave Leake | Dec 13 2021 | Update Design Definition |
|  |  |  |  |  |  |

TABLE OF CONTENTS

[1 Introduction 5](#_Toc180482593)

[1.1 Purpose of The Product Design Specification Document 5](#_Toc180482594)

[2 General Overview and Design Guidelines/Approach 5](#_Toc180482595)

[2.1 Assumptions / Constraints / Standards 5](#_Toc180482596)

[3 Architecture Design 5](#_Toc180482597)

[3.1 Logical View 5](#_Toc180482598)

[3.2 Hardware Architecture 5](#_Toc180482599)

[3.3 Software Architecture 5](#_Toc180482600)

[3.4 Security Architecture 5](#_Toc180482601)

[3.5 Communication Architecture 5](#_Toc180482602)

[3.6 Performance 6](#_Toc180482603)

[4 System Design 6](#_Toc180482604)

[4.1 Use-Cases 6](#_Toc180482605)

[4.2 Database Design 6](#_Toc180482606)

[4.3 Data Conversions 6](#_Toc180482607)

[4.4 Application Program Interfaces 6](#_Toc180482608)

[4.5 User Interface Design 6](#_Toc180482609)

[4.6 Performance 6](#_Toc180482610)

[4.7 Section 508 Compliance 6](#_Toc180482611)

[5 Product Design Specification Approval 7](#_Toc180482612)

[Appendix A: References 8](#_Toc180482613)

[Appendix B: Key Terms 9](#_Toc180482614)

# Introduction

## Purpose of The Product Design Specification Document

The Product Design Specification Document documents and tracks the necessary information required to effectively define architecture and system design in order to give the development team guidance on architecture of the system to be developed. The Product Design Specification Document is created during the Planning Phase of the project. Its intended audience is the project manager, project team, and development team. Some portions of this document such as the user interface (UI) may on occasion be shared with the client/user, and other stakeholder whose input/approval into the UI is needed.

# General Overview and Design Guidelines/Approach

This section describes the principles and strategies to be used as guidelines when designing and implementing the system.

## Assumptions

There is a general assumption that this product is a proof of concept and not a polish/published work. The team working on this product are preforming an exploratory design.

There is a general understanding of a major time constraint and not all of the of the desired features will be finished to the polished level that the team may want. As this product is proof of concept this should not prove to be an issue.

## Constraints

The product is written and maintained in the Java programming language and is only intended to run in a Java Runtime Environment (JRE). The development team designed, implemented, and tested only in the Windows Operating System (OS).

The SQL Database in use is based on a government owned system. Much of which is not accessible from outside a secured network. Limiting the usability of much of the database. Two of the tables have been scrubbed and cleansed of protected/sensitive information and are the focus of this proof of concept. As such much of the full capabilities of the project are not available during the proof of concept, such a user lookup, external emailing’s services, or a full catalogue of previous requests.

# Architecture Design

This section outlines the system and hardware architecture design of the system that is being built. For the proof of concept to be considered successful the Work Request Application is required to be able to:

* Add a new record
* View all retrievable records from database
* Access localhost **and** ability to access internet hosted server
* Have some visual breakdowns of the information present in the record

The Work Request Application is designed with a Model-View architecture or Model – View – Controller (MVC) as its more commonly known. Allowing for intermittent connectivity to the database and eliminating the need to continuous connection to function correctly. For this use case the Graphical User Interface will function as both the controller and view for internal model. See figure 1. found below, for visual reference.

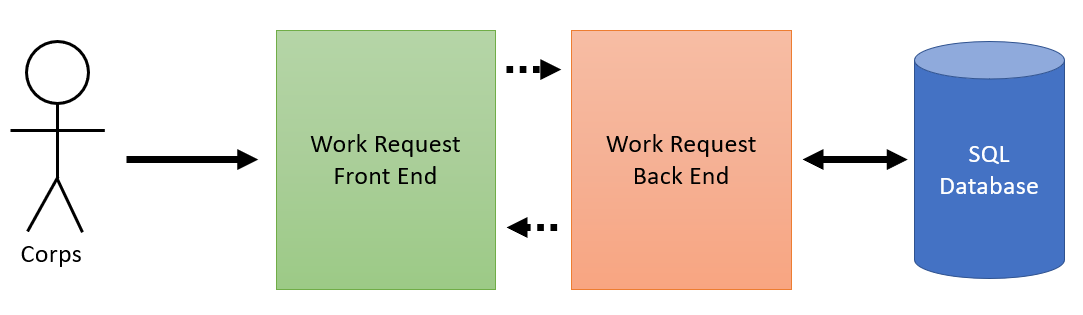


Figure . Work Request Application High Level Concept

## Logical View

The Work Request Application will be able to navigate between multiple tabs. One section to display the records stored in the SQL Database, using an internal model as intermediator. And to send requests to SQL Database to store a new record in specified tables.

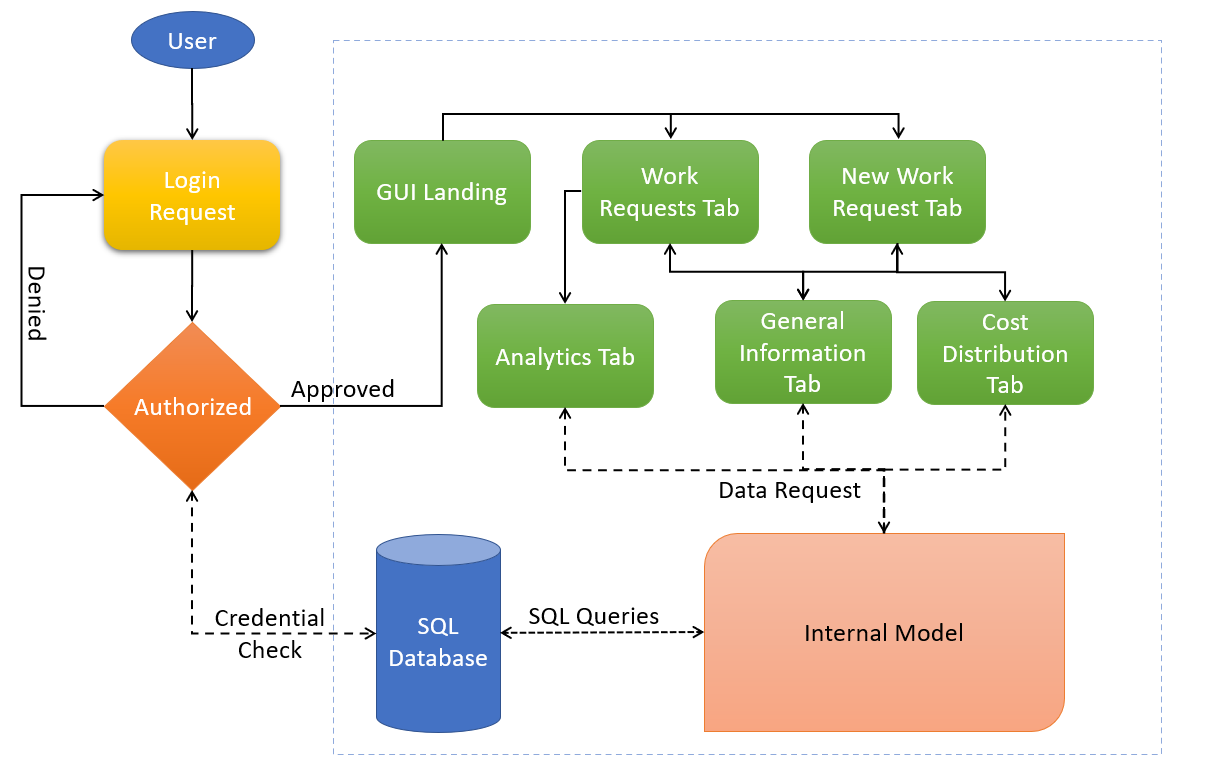


Figure . Work Request Application High Level Logic Diagram

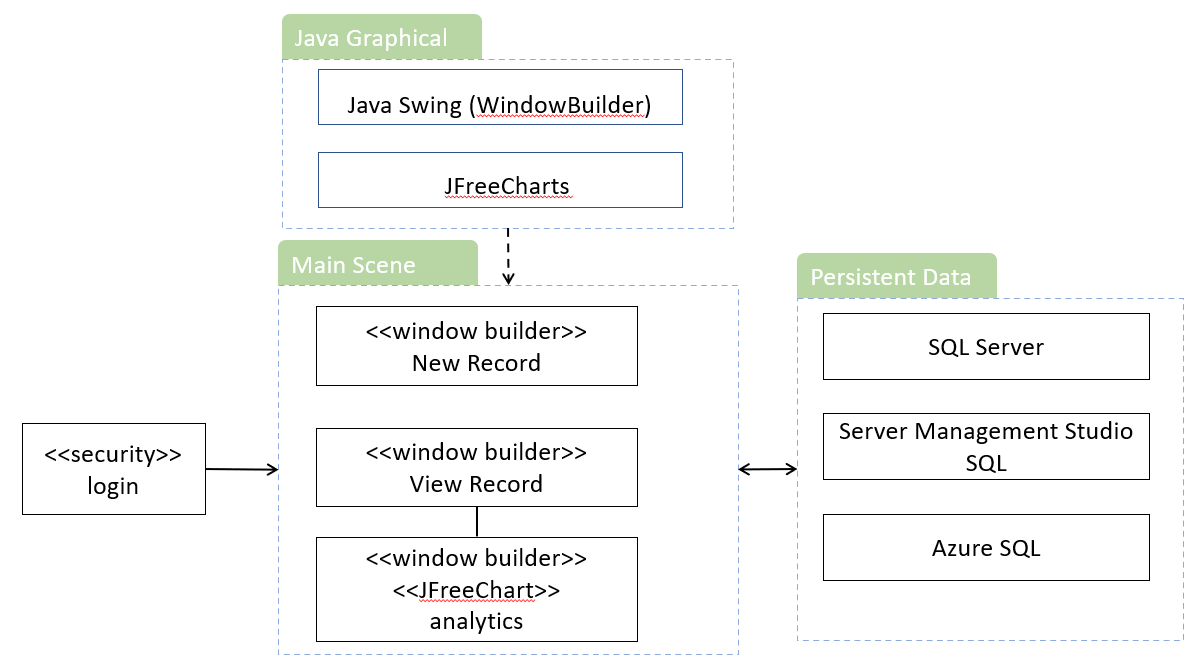


Figure . Work Request Application Component Diagram

## Software Architecture

The Work Request Applications main function is for proof of concept for portability of existing software and database to a Java/SQL system. With no official funding for this project all software requirements are free and/or open source frameworks. Including Java Development Kit 17, Eclipse Community Edition 4.21.0, JFreeChart 1.5.3, MSSQL JDBC 9.4.0, WindowBuilder, Microsoft SQL Server Management Studio v18.0, and Github.

* Language: Java
* Development Environment: Eclipse IDE Version 4.21.0
* Development GUI: WindowBuilder Eclipse Plugin
* Development GUI: JFreeChart 1.5.3
* Database: SQL Server
* Operating System: Windows with JRE 16+
* Source Control: Github

## Performance

There are no performance requirements for the Work Request Application as it is mainly a proof-of-concept project. Functionally each screen, data request, and analytics request would take less than a 3 second lag time. But no hard requirements have been set for performance for this project.

## Alternate architecture (Depricated)

The previous design of this project included functionality that is either unobtainable or adds no benefit to the project as an exploratory design. As such it has been deprecated and replaced with a new design that meets all expected requirements.

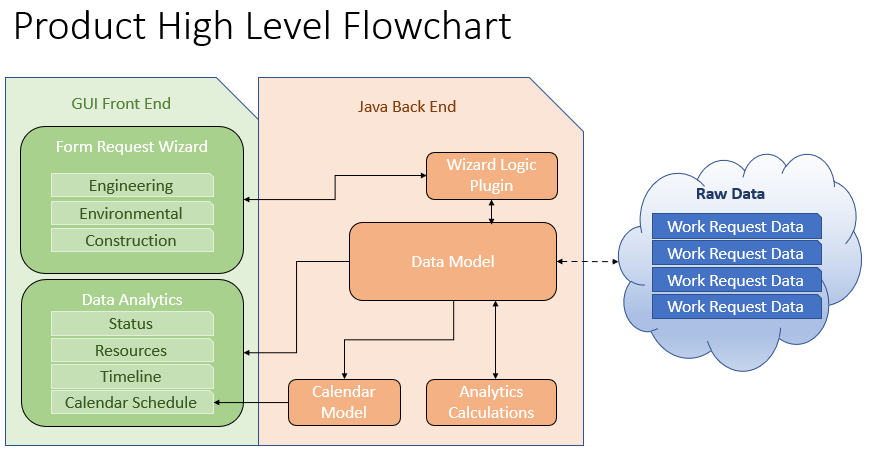


Figure . Deprecated High Level Flowchart and Design

# System Design

## Use-Cases

Use cases are described in the Team3WorkRequestHighLevelRequirements.docx.

## Database Design

The Database design and decencies can be found in the files USACE\_ECWR\_Dependencies.pdf and USACE\_ECWR\_DependencyDiagram.pdf

## User Interface Design

The user interface design is focused on two components. A record viewer and a record insertion. The record viewer also has a visual analytics component. For the purposes of this exploratory project, the analytics tab has been limited to a few fields and only two charts. These can be expanded at a later date if the finding are positive.

### WORK REQUEST TAB

The work request tab uses an internal model of ArrayList<Map<String, Object>> to store each data point in the records from the SQL Database. This allows for the connection to be made and verified once per update and only for short burst communications. Allowing for more limited or unstable internet connection to be usable for the application.

The viewing tab for work requests only needs a connection to the Database to request information. This allows for a shorter connection time than inserting a new request into the database.

There is also an analytics tab that take the stored information in the internal model and creates two different visual charts. A pie chart of cost distribution breakdown and a gnatt chart for overall workflow progress. The estimation dates are all that is required for this exploratory phase.

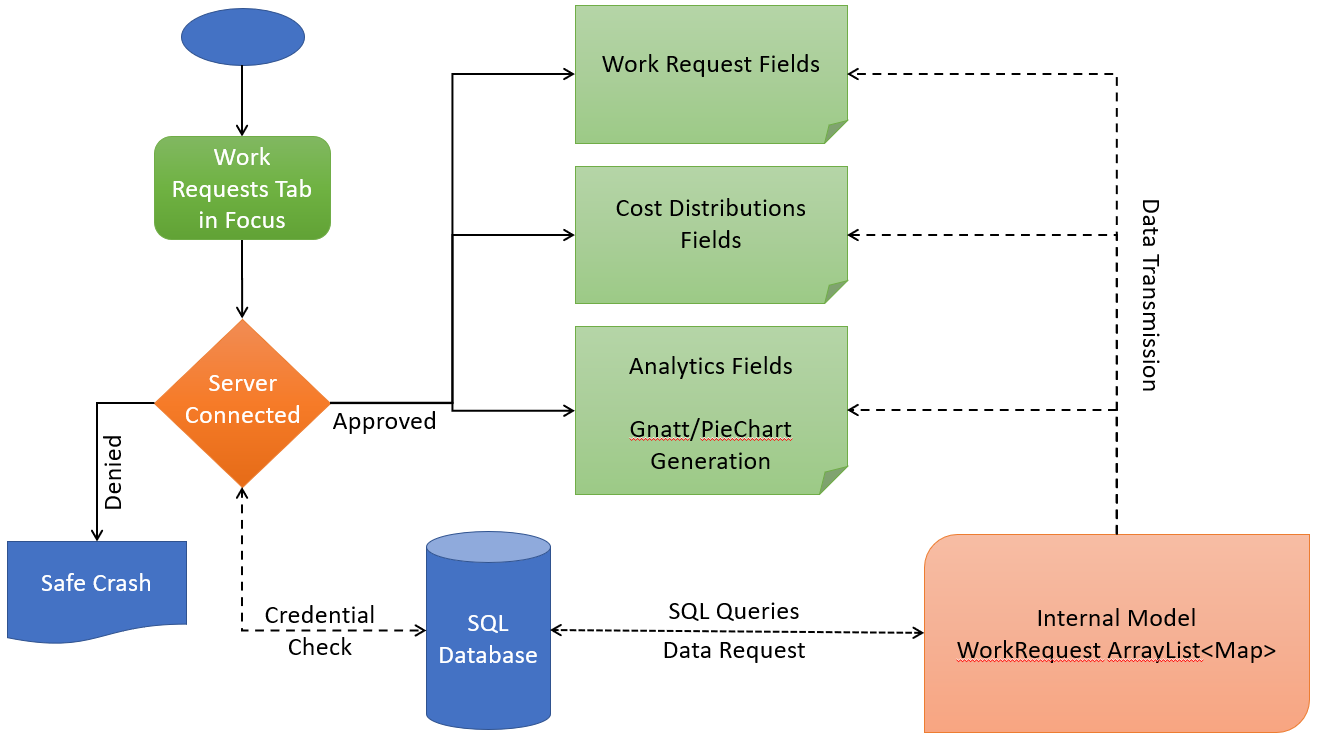


Figure . View Work Requests Tab Diagram

### NEW WORK REQUEST TAB

The New Work Request tab is for inserting a new record into the database. Utilizing a sendmap internal model to store the data from SQL server, an insertion request is made from this data to the database. This requires a longer connection on average.

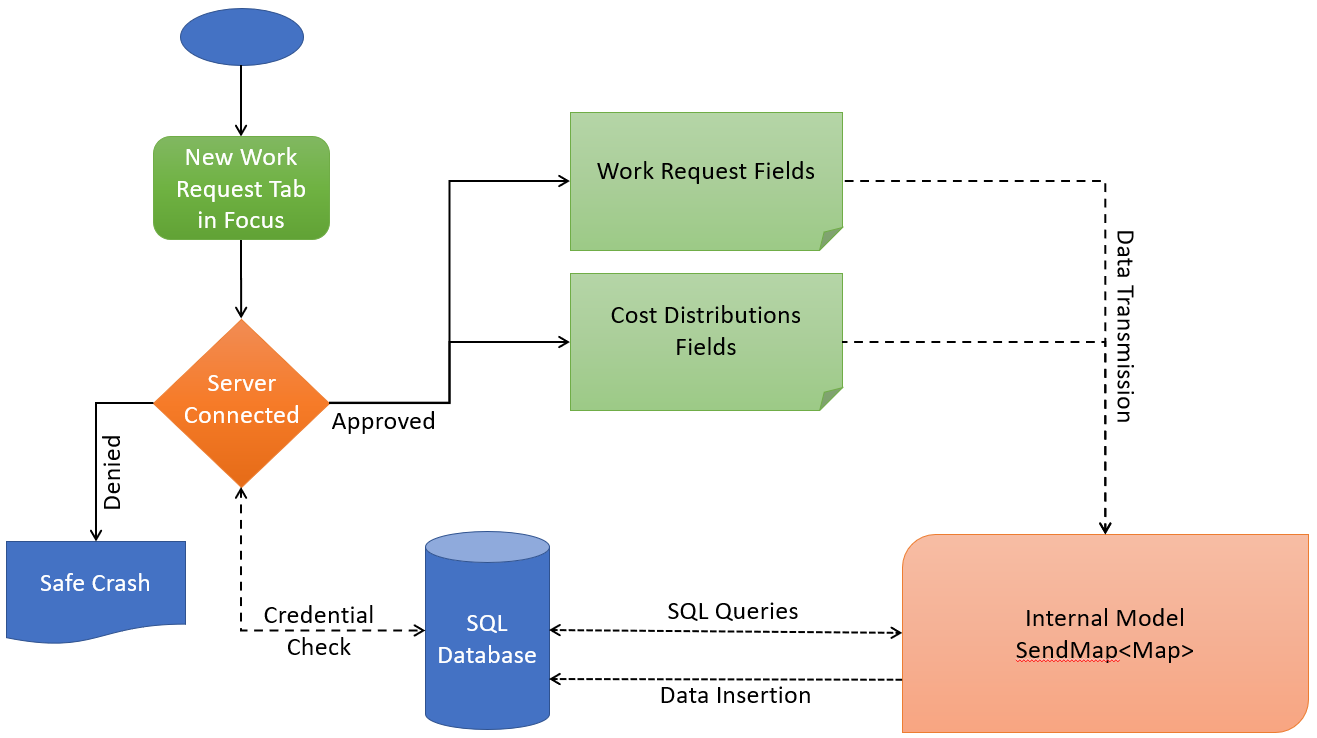


Figure . Insert New Work Request Tab Diagram