****

**YAŞAR UNIVERSITY**

**FACULTY OF ENGINEERING**

**DEPARTMENT OF COMPUTER ENGINEERING**

**COMP4920 Senior Design Project II, Fall 2023**

**Advisor: Mehmet Ufuk Çağlayan**

**WEBREM: A Remembrances or Condolences Web Service**

**Final Report**

**(Bachelor of Science Thesis)**

**22.05.2023**

**By:**

**Atakan Aktakka, Student ID: 18070001032**

**Alican Daşdemir, Student ID: 18070001041**

**Doğukan Tez, Student ID: 18070001051**

**Kadir Ülge, Student ID: 18070001014**

# PLAGIARISM STATEMENT

This report was written by the group members and in our own words, except for quotations from published and unpublished sources which are clearly indicated and acknowledged as such. We are conscious that the incorporation of material from other works or a paraphrase of such material without acknowledgement will be treated as plagiarism according to the University Regulations. The source of any picture, graph, map or other illustration is also indicated, as is the source, published or unpublished, of any material not resulting from our own experimentation, observation or specimen collecting.

**Project Group Members:**

|  |  |  |  |
| --- | --- | --- | --- |
| Name, Lastname | Student Number | Signature | Date |
| Atakan Aktakka | 18070001032 |  |  |
| Alican Daşdemir | 18070001041 |  |  |
| Doğukan Tez | 18070001051 |  |  |
| Kadir Ülge | 18070001014 |  |  |

**Project Supervisors:**

|  |  |  |  |
| --- | --- | --- | --- |
| Name, Lastname | Department | Signature | Date |
| Prof. Dr. Mehmet Ufuk Çağlayan | Computer Engineering |  |  |
|  |  |  |  |

# ACKNOWLEDGEMENTS

First and foremost, I would like to express my utmost gratitude to my supervisor, Mehmet Ufuk Çağlayan, for his invaluable guidance throughout the course of these projects. His unwavering support, valuable insights, and assistance during challenging times were instrumental in our triumph. The project's successful completion owes much to his unwavering motivation and aid.

# KEYWORDS

* WEBREM
* Remembrances
* Condolences
* Memory

# ABSTRACT

WEBREM is a profound Remembrances or Condolences Web Service that endeavors to create an exceptional platform. The primary objective of this remarkable project is to craft and refine the WEBREM service, encompassing a plethora of functions. These functions encompass the ability to effortlessly upload cherished memories, be it in the form of texts, captivating photographs, poignant audio recordings, or captivating video recordings, all securely stored within a robust database through an intuitive web interface. Notably, these memories are meticulously moderated, ensuring a curated collection. Furthermore, the system employs an advanced indexing mechanism, facilitating seamless searchability and effortless retrieval. Ultimately, the aim is to provide users with unfettered access to these cherished memories, granting them solace and a means to cherish their experiences.

# ÖZET

WEBREM, derin bir Anma veya Başsağlığı Web Hizmeti olarak adlandırılan özel bir platformu yaratmayı hedefleyen bir hizmettir. Bu önemli projenin temel amacı, çok çeşitli işlevleri olan WEBREM hizmetini oluşturmak ve geliştirmektir. Bu işlevler, kullanıcıların metinleri, etkileyici fotoğrafları, dokunaklı ses kayıtlarını veya büyüleyici video kayıtlarını kolaylıkla yükleyebilecekleri ve güvenli bir veritabanında saklanabilecekleri kullanıcı dostu bir web arayüzü üzerinden gerçekleştirilir. Bu anılar özenle yönetilerek bir koleksiyon halinde sunulur. Ayrıca, sistem, gelişmiş bir indeksleme mekanizması kullanarak sorunsuz arama ve kolay erişim sağlar. Temel olarak, kullanıcılara bu değerli anılara sınırsız erişim imkanı sunmak ve onlara teselli etmek ve deneyimlerini yaşatmak için bir araç sağlamak amaçlanır.

# TABLE OF CONTENTS

[PLAGIARISM STATEMENT ii](#_Toc137218926)

[ACKNOWLEDGEMENTS iii](#_Toc137218927)

[KEYWORDS iv](#_Toc137218928)

[ABSTRACT v](#_Toc137218929)

[ÖZET vi](#_Toc137218930)

[TABLE OF CONTENTS vii](#_Toc137218931)

[LIST OF FIGURES x](#_Toc137218932)

[LIST OF TABLES xi](#_Toc137218933)

[LIST OF ACRONYMS/ABBREVIATIONS xii](#_Toc137218934)

[1. INTRODUCTION 13](#_Toc137218935)

[1.1. Description of the Problem 13](#_Toc137218936)

[1.2. Project Goal 13](#_Toc137218937)

[1.3. Project Output 13](#_Toc137218938)

[1.4. Project Activities and Schedule 13](#_Toc137218939)

[2. REQUIREMENTS 14](#_Toc137218940)

[3. DESIGN 15](#_Toc137218941)

[3.1. High Level Design 15](#_Toc137218942)

[3.2. Detailed Design 16](#_Toc137218943)

[3.2.1. WEBREM Main Class: 16](#_Toc137218944)

[3.2.2WEBREM Subsystem Classes: 16](#_Toc137218945)

[3.2.2.1. NonRegisteredUser Class: 16](#_Toc137218946)

[3.2.2.2. RegisteredUser Class: 16](#_Toc137218947)

[3.2.2.3. Admin Class: 17](#_Toc137218948)

[3.3. Realistic Restrictions and Conditions in the Design 18](#_Toc137218949)

[4. IMPLEMENTATION, TESTS and TEST DISCUSSIONS 18](#_Toc137218950)

[4.1. Implementation of the Product 18](#_Toc137218951)

[4.2. Tests and Results of Tests 18](#_Toc137218952)

[5. CONCLUSIONS 19](#_Toc137218953)

[5.1. Summary 19](#_Toc137218954)

[5.2. Cost Analysis 19](#_Toc137218955)

[5.3. Benefits of the Project 20](#_Toc137218956)

[5.4. Future Work 21](#_Toc137218957)

[References 21](#_Toc137218958)

[APPENDICES 21](#_Toc137218959)

[APPENDIX A: REQUIREMENTS SPECIFICATION DOCUMENT 21](#_Toc137218960)

[Revision History 23](#_Toc137218961)

[Table of Contents 24](#_Toc137218962)

[Table of Figures 25](#_Toc137218963)

[1. Introduction to WEBREM 26](#_Toc137218964)

[2. WEBREM Requirements - Use Case Table 26](#_Toc137218965)

[3. Actors and Use Case Diagram 27](#_Toc137218966)

[3.1. Actors in WEBREM 27](#_Toc137218967)

[3.2. WEBREM Use Case Diagram 28](#_Toc137218968)

[3.3. WEBREM User Activity Diagram 28](#_Toc137218969)

[3.4. WEBREM Admin-Moderators Activity Diagram 29](#_Toc137218970)

[3.5. WEBREM Use Case Description Table 30](#_Toc137218971)

[4. Non-Functional Requirements 30](#_Toc137218972)

[4.1. Development Environment 30](#_Toc137218973)

[4.2. Security 31](#_Toc137218974)

[4.3. Multiple Language Support 31](#_Toc137218975)

[4.4. WEBREM Administrators 31](#_Toc137218976)

[4.5. Logging 31](#_Toc137218977)

[4.6. Testing 31](#_Toc137218978)

[4.7. Business Rules 31](#_Toc137218979)

[4.8. Copyright Rules 32](#_Toc137218980)

[5. References 32](#_Toc137218981)

[APPENDIX B: DESIGN SPECIFICATION DOCUMENT 32](#_Toc137218982)

[Revision History 34](#_Toc137218983)

[Table of Contents 35](#_Toc137218984)

[Table of Figures 36](#_Toc137218985)

[1. Introduction 37](#_Toc137218986)

[2. WEBREM Software Design 37](#_Toc137218987)

[2.1. WEBREM Software System Architecture 37](#_Toc137218988)

[2.2. WEBREM Software System Structure 38](#_Toc137218989)

[2.2.1. Controller Layer 38](#_Toc137218990)

[2.2.2. Model Layer 38](#_Toc137218991)

[2.2.3. View Layer 39](#_Toc137218992)

[2.3. WEBREM Software System Environment 39](#_Toc137218993)

[3. WEBREM Software System Detailed Design: 40](#_Toc137218994)

[3.1. WEBREM Main Class 40](#_Toc137218995)

[3.2. WEBREM Subsystem Classes 40](#_Toc137218996)

[3.2.1. NonRegisteredUser Class 40](#_Toc137218997)

[3.2.2. RegisteredUser Class 41](#_Toc137218998)

[3.2.3. Moderator Class 43](#_Toc137218999)

[3.2.4. MasterModerator Class 43](#_Toc137219000)

[3.2.5. Admin Class 43](#_Toc137219001)

[3.2.6. MasterAdmin Class 43](#_Toc137219002)

[4. Testing Design 44](#_Toc137219003)

[4.1 Testing Strategy 44](#_Toc137219004)

[References 44](#_Toc137219005)

[APPENDIX C: PRODUCT MANUAL 45](#_Toc137219006)

[Revision History 46](#_Toc137219007)

[Table of Contents 47](#_Toc137219008)

[1. Introduction 48](#_Toc137219009)

[2. WEBREM Software Subsystem Implementation 48](#_Toc137219010)

[2.1. Source Code 48](#_Toc137219011)

[2.1.1 Controller Layer 48](#_Toc137219012)

[2.1.2 Model Layer 48](#_Toc137219013)

[2.1.3 View Layer 48](#_Toc137219014)

[2.2 Executable Organization 49](#_Toc137219015)

[2.2.1 System Environment 49](#_Toc137219016)

[2.2.2 User Interaction 49](#_Toc137219017)

[2.2.3 Testing 49](#_Toc137219018)

[2.3. Software Development Tools 49](#_Toc137219019)

[2.4. Hardware and System Software Platform 50](#_Toc137219020)

[3. WEBREM Software Subsystem Testing 50](#_Toc137219021)

[4. WEBREM Installation, Configuration and Operation 50](#_Toc137219022)

[References 51](#_Toc137219023)

[APPENDIX D: SOURCE CODE/EXECUTABLES/SCRIPTS IN CD/DVD 52](#_Toc137219024)

# LIST OF FIGURES

[Figure 1. WEBREM Software System Structure 15](#_Toc136202125)

[Figure 2. Use Case Diagram 28](#_Toc136202126)

[Figure 3. User Activity Diagram 29](#_Toc136202127)

[Figure 4. Admin-Moderators Activity Diagram 29](#_Toc136202128)

[Figure 5. WEBREM Software System Structure 38](#_Toc136202129)

# LIST OF TABLES

Table 1. Requirements …………………………………..………………………………14

# LIST OF ACRONYMS/ABBREVIATIONS

WEBREM A Remembrances or Condolences Web Service

RSD Requirements Specifications Document

DSD Design Specifications Document

PM Product Manuel

MVC Model View Controller

# 1. INTRODUCTION

# 1.1. Description of the Problem

When it comes to sharing their priceless memories or expressing genuine condolences, people typically struggle in today's digital era with a lack of understanding and support. The main worry is from trepidation in communicating their own experiences and feelings due to uncertainty about the reaction they could get from their audience. As a result, they travel with a strong sense of isolation, which makes it harder for them to get the help and understanding they need when things go tough. Therefore, there is an innate need for a kind and welcoming platform that promotes a secure and nurturing environment, enabling people to openly express their experiences and emotions without worrying about being judged while also getting the compassion and consolation they rightfully deserve.

# 1.2. Project Goal

The proposed project aims to develop a Remembrances or Condolences Web Service, known as WEBREM. The goal of WEBREM is to provide individuals with a platform to upload and share memories, specifically condolences about deceased individuals, in various formats such as texts, photographs, audio recordings, or video recordings. The service will ensure a moderated approach to curating and storing these memories within a secure database, accessible through a user-friendly web interface. Additionally, WEBREM will implement efficient indexing and updating mechanisms to facilitate easy search and retrieval of the stored memories. The ultimate objective is to create a supportive and empathetic environment where individuals can freely express their personal experiences and emotions, finding solace, support, and understanding during difficult times.

# 1.3. Project Output

The WEBREM project is expected to produce a list of potential outputs, including:

1. WEBREM-RSD-Rev-2.0
2. WEBREM-DSD-Rev-1.3
3. WEBREM Poster
4. WEBREM-PM-Rev-1.0

# 1.4. Project Activities and Schedule

* Requirements specifications document (RSD) report version 2.00 Written on 15 April 2022
* Design specifications document (DSD) report version 2.00 was written on 29 April 2022
* Product Manuel Document (PM) report version 1.00 written on 25 March 2023

# 2. REQUIREMENTS

|  |  |  |
| --- | --- | --- |
| No. | Requirements | Use Case |
| 1 | To enter the system and share my memories of famous people or institutions, I need to be registered. After entering my e-mail, I can set my password and register. Identity verification by e-mail is required for all registered users. Users need to accept personal data protection authority article via checkbox to create an account. | Create an account |
| 2 | After registration, the user can activate their account and start using the application with the activation link sent to their registered mail account as a result of mail verification. | Activate account |
| 3 | All WEBREM Users will be able to view all pages opened in the WEBREM service through shared links. However, non-registered WEBREM users will only be able to preview the memories on the opened page. To see the entire shared memory, it is necessary to be a WEBREM registered user. | Surf in the site and make queries within the site |
| 4 | The registered user will fill out a form to sign in fill the email and password boxes and push the sign in button | Sign in to WEBREM |
| 5 | Button I FORGOT MY PASSWORD should mail the password of the user to his/her first entered email address (not the address entered on update screen). If user has forgotten his/her initial email address, there is no chance that he/she will retrieve the password | Update registration information |
| 6 | Registered user can upload memories about a particular person/institution. Registered user will be able to share memories by pressing the SHARE button. | Share memories |
| 7 | Registered WEBREM users will be able to update their memories on request by using the UPDATE button. WEBREM The registered user will be able to delete the shared memory during the update with the DELETE button | Update shared memories |
| 8 | All WEBREM Users will be able to browse through the memories stored in the WEBREM service, make searches about who has written/uploaded what or who has written/uploaded what, and will be able to download the memories in the form of text, photo, sound recording, video recording to their own computer by pressing download button | Download the memories |
| 9 | Admin in order to create memorial page about a particular person/institution, in his admin screen selects add memorial page and types memorial page name then creates memorial page by clicking add new memorial page. | Create a memorial page |
| 10 | Admin has the ability to ban users based on posts with inappropriate content. | Ban users |

Table 1. Requirements

In the final report, certain components have been removed from the design. Firstly, the process of assigning moderators for new pages after the Master Moderator's approval has been eliminated. This means that there won't be a specific step to appoint one or more moderators for each person or institution on WEBREM. Additionally, the procedure of filling out an application form for opening a page and having it examined and approved by the Master Moderator has been excluded from the system. As a result, the check for page opening forms by the Master Moderator is no longer part of the functionality. In addition, in order to improve the operation of the application and prevent spam, the memorial page creation will be done by the admin instead of the users. These modifications have been reflected in the updated final report.

# 3. DESIGN

# 3.1. High Level Design

The WEBREM web service will be developed using Node.js. Based on the project requirements and the chosen development environment, a layered architecture is deemed the most suitable option for the system architecture.

diyagram içeren bir resim

Açıklama otomatik olarak oluşturuldu

Figure 1. WEBREM Software System Structure

# 3.2. Detailed Design

The WEBREM Software System Detailed Design describes the architecture and functionality of the application. It consists of various classes and methods that contribute to its overall operation. Among these classes, the RegisteredUser, NonRegisteredUser, Moderator, and Admin classes play key roles in different aspects of the system.

The RegisteredUser class handles functionalities such as making queries, signing in, updating registration information, sharing and updating memories, downloading shared memories, and creating memorial pages. The NonRegisteredUser class focuses on creating accounts for non-registered users and activating those accounts through email verification. The Moderator class is responsible for reporting inappropriate user behavior and content to the admin, while the Admin class deals with actions like banning users based on reports.

These classes and their respective methods provide the necessary functionality for users to interact with the WEBREM Software System effectively. They enable users to navigate the site, manage their accounts and information, share memories, report issues, and administer the system. The detailed design section of the final report provides further information on these classes, including the inputs, outputs, and specific functionalities associated with each method.

# 3.2.1. WEBREM Main Class:

This class represents the main program or process for the developments in WEBREM.

# 3.2.2WEBREM Subsystem Classes:

# 3.2.2.1. NonRegisteredUser Class:

* This class is responsible for handling non-registered user functionalities.

Method: createAnAccount()

* Takes a NonRegisteredUser object as input and creates an account for non-registered users.
* Requires the user to fill out a form with information such as email, password, firstname and surname.
* Provides submit and cancel buttons for sign-up.

Method: activateAccount()

* Takes a NonRegisteredUser object as input and sends an activation email to the non-registered user.
* After creating an account, an email is sent to the user for account activation.

# 3.2.2.2. RegisteredUser Class:

* This class handles functionalities for registered users.

Method: makeQuery()

* Allows registered users to make queries on the site using the search bar and navigate within the site.

Method: signIn()

* Takes a RegisteredUser object as input and logs the user into the system.
* Activated registered users can sign in by clicking the sign-in button.

Method: updateRegistrationInfo()

* Takes a RegisteredUser object as input and updates the registered user's information.
* Requires the user to fill out a form with information such as email, password, and provides buttons to show registration info, cancel the update, and recover a forgotten password.

Method: shareMemories()

* Takes RegisteredUser and Memory objects as input and allows the user to share memories.
* The user can fill out a form with information such as adding photos, text, audio recordings, and video recordings.

Method: updateSharedMemories()

* Takes RegisteredUser and Memory objects as input and allows registered users to update shared memories.
* Provides the ability to update or delete shared memories without going through moderator approval.
* Requires the user to fill out a form with information such as photo/text/audio recordings/video recordings.

Method: downloadMemories()

* Takes a Memory object as input and allows registered users to download shared memories.

# 3.2.2.3. Admin Class:

Method: banUsers()

* **Takes a RegisteredUser object as input and bans the user after evaluating reports about them.**

Method: createMemorialPage()

* Opens the page by typing the information of the person or institution to be created
* Takes necessary steps in the admin panel to create a memorial page. By clicking "Add Memorial Page," they enter the name and select "Add New Memorial Page," establishing a dedicated tribute page for a person or institution.

# 3.3. Realistic Restrictions and Conditions in the Design

* The application can only serve up to 1000 users simultaneously and will not be able to handle more users without crashing or experiencing performance issues.
* The application does not support distributed files, and all data must be stored on a single server.
* The application does not include any features for data migration or data transfer, and all data must be manually transferred if the application is moved to a different server or database.

# 4. IMPLEMENTATION, TESTS and TEST DISCUSSIONS

# 4.1. Implementation of the Product

A web application called WEBREM, created with NodeJS, intends to give users a place to collect and share memories about significant individuals or organizations. Users can upload memories in a variety of formats, such as text, images, audio files, or video files, thanks to the application's straightforward and user-friendly design.

The model layer takes charge of data management, utilizing a MongoDB database to store and retrieve information related to tasks, users, and other crucial aspects of the application. By providing the necessary methods for data access and manipulation, this layer ensures the integrity and persistence of the system's data.

The view layer serves as the interface between the different components, facilitating effective communication and seamless integration. Developed using HTML, CSS, and ReactNative, this layer enables users to interact with the system, initiate requests, and receive appropriate responses.

Throughout the implementation process, rigorous testing methodologies are employed to ensure the system's reliability and functionality. Unit testing, integration testing, and system testing are conducted using tools such as Jest and Puppeteer, verifying the correct operation of individual components, their interactions, and the overall performance of the subsystem.

For a more comprehensive understanding of the WEBREM Software Subsystem and its implementation details, please refer to the Product Manual in Appendix C: Product Manual, v1.0.

# 4.2. Tests and Results of Tests

The testing phase in the development of the WEBREM software system involves different levels of testing, including unit testing, integration testing, and end-to-end testing. The purpose of these tests is to ensure the correct functioning and integration of the components, as well as the overall functioning of the service.

* Unit testing will be performed on individual components or classes to verify their correct functioning in isolation. This testing will involve creating test cases that cover the various methods and features of each component. The expected results will be defined, and the tests will validate that the components produce the expected outputs.
* Integration testing will be conducted to verify that the different components of the service work together correctly. This testing phase will focus on testing the interactions between the view layer, the model layer, and the controller layer. The tests will ensure that data flows correctly between these components and that they cooperate effectively.
* End-to-end testing will be performed to verify the overall functioning of the service as a whole. This testing phase simulates user actions and assesses the service's response to these actions. Different user scenarios will be created to test the service comprehensively, and the tests will cover various user interactions. The goal is to ensure that the service behaves as expected from a user's perspective.

The detailed design and specifications for testing are provided in Appendix B: Design Specifications Document, 2.0, Section 4 - Testing Design. This section outlines the testing approach and strategies to be used in the testing phase.

The software testing environment is specified in Appendix B: Design Specifications Document, 2.0, Section 4.1 - WEBREM Software Testing Environment. This section provides information about the environment in which the testing will be conducted, including hardware, software, and any additional tools or resources required.

The Tests and Results of Tests section will be completed in COMP 4920, where the actual testing will take place and the test cases and results will be documented.

# 5. CONCLUSIONS

# 5.1. Summary

The WEBREM service is an innovative web-based platform designed to facilitate the sharing of cherished memories and heartfelt condolences concerning significant individuals or institutions. Users will have the ability to seamlessly upload various forms of content, including texts, photographs, audio recordings, and video recordings, each of which will be meticulously stored and moderated within a comprehensive database. Accessible through a user-friendly web interface, the service will empower users to effortlessly search for and retrieve shared memories. The High-Level Design of the WEBREM service incorporates a comprehensive diagram illustrating the key components of the system and their interconnectedness. To successfully develop the WEBREM service, the project team must diligently address the technical challenges and prerequisites, while taking into account the realistic constraints and contextual factors that may influence the system's design.

# 5.2. Cost Analysis

To calculate the cost of the project, you will need to multiply the number of hours worked by the hourly rate for each team member. For example, if the hourly rate for each team member is $50, the total cost for one team member working on the project for 560 hours will be 560 x $20 = $11,200. If there are four team members working on the project, the total cost will be 4 x $11,200= $44,800.

# 5.3. Benefits of the Project

The benefits of the WEBREM project are likely to be numerous, as the service has the potential to provide a valuable resource for the community. Some potential benefits of the project include:

* Creating a platform for people to express their condolences and memories: The WEBREM service will give people a place to express their condolences and recollections about significant individuals or institutions, which may help to maintain their legacies and let others to remember and respect them.
* Establishing a user community: By offering a platform for sharing condolences and memories, the WEBREM service has the potential to establish a user community where users may interact and support one another by sharing their memories.
* Giving users access to a searchable database of memories: The WEBREM service will come with a searchable database of memories, making it simple for users to locate and access the shared experiences. This will make it easier for the community to access and use the memories by preserving and organizing them.
* Supporting research and education: Researchers and educators may utilize the WEBREM service to access and use the shared memories, which may offer insightful details and data for their work.
* Preserving history and culture: By providing a platform for sharing memories and condolences, the WEBREM service has the potential to help preserve the history and culture of the community, and to provide a valuable resource for future generations.

Overall, the WEBREM project has the potential to serve the community in a number of ways by giving people a place to share their memories and condolences and by developing a priceless resource for keeping and retrieving those memories.

# 5.4. Future Work

The future prospects for the WEBREM project are promising, with several potential areas of development and expansion. These include:

1. Integration with social media platforms: By integrating WEBREM with popular social media platforms such as Facebook or Instagram, users can conveniently share their memories and connect with a broader audience, fostering a sense of community and engagement.
2. Implementation of memory organization features: A valuable future direction would involve incorporating functionality that allows users to organize their memories into collections or albums. This enhancement would streamline the management of memories, enabling users to easily categorize and share their experiences with others.
3. Establishing strategic partnerships: Exploring partnerships with institutions or organizations, such as universities or historical societies, can unlock opportunities for collaboration. WEBREM could serve as a dedicated platform for these entities to store and share memories specific to their institutions, enriching their archival efforts and fostering a stronger sense of historical preservation.
4. Introduction of premium services or subscription models: To ensure the sustainable development and maintenance of WEBREM, a viable option is to offer special memory storage services to users through subscription fees or by providing premium features. This approach would not only generate revenue but also incentivize continuous improvement and ongoing support for the platform.

By pursuing these future avenues, the WEBREM project can evolve into a dynamic and comprehensive platform, enhancing user experiences, fostering broader engagement, and solidifying its position as a leading provider of remembrance and condolence services.

# References

1. System Requirements Specification Document, Revision 2.0, in file WEBREM-RSD-2022-11-15-Rev-2.0.doc.
2. High Level Design, Design Specifications Document, Revision 1.3, in file WEBREM-DSD-Rev-1.3.doc.
3. Product Manuel, Revision 1.0, WEBREM-PM-Rev-1.0.doc

# APPENDICES

# APPENDIX A: REQUIREMENTS SPECIFICATION DOCUMENT

**COMP4910 Senior Design Project 1, Fall 2022**

**Advisor: Prof. Dr. Mehmet Ufuk Çağlayan**

**WEBREM**

**Requirements Specifications Document**

**19.12.2022**

**Revision 2.1**

**By:**

**Atakan Aktakka, 18070001032**

**Kadir Ülge, 18070001014**

**Doğukan Tez, 18070001051**

**Alican Daşdemir, 18070001041**

# Revision History

|  |  |  |
| --- | --- | --- |
| **Revision** | **Date** | **Explanation** |
| 1.0 | 08.10.2022 | Initial requirements |
| 1.1 | 15.10.2022 | Requirements 2.2.2. Function 1.2. and 2.2.8. Function 2.6. are modified |
| 1.2 | 24.10.2022 | Requirements 2.2.7. Function 2.5. and 2.2.17. Function 6.1. are added |
| 1.3 | 05.11.2022 | Requirements 3.3. and 3.4. are modified |
| 1.4 | 11.11.2022 | References are modified |
| 1.5 | 15.11.2022 | Requirement 3.8. is added |
| 1.6 | 25.11.2022 | Requirements 2.2.9. Function 3.1., 2.2.10. Function 3.2.,  2.2.12. Function 4.1. and 2.2.13. Function 4.2. are deleted. |
| 1.7 | 29.11.2022 | Requirements 2.2.12. Function 4.2. and 2.2.14. Function 6.1. are added. |
| 1.8 | 02.12.2022 | 2.3. Short use case description for all use functional requirements and 2.4. Use Case Diagram are added |
| 1.9 | 04.12.2022 | 4. UML Activity Diagram, 4.1. User Activity Diagram and 4.2. Admin-Moderators Activity Diagram are added. |
| 2.0 | 07.12.2022 | Table of figures added. 3.7. Requirement 3.7. Business Rules and 3.1 Requirement 3.1. Development Environment are modified. |

# Table of Contents

[Revision History 2](#_Toc122535184)

[Table of Contents 3](#_Toc122535185)

[Table of Figures 4](#_Toc122535186)

[1. Introduction to WEBREM 5](#_Toc122535187)

[2. WEBREM Requirements - Use Case Table 5](#_Toc122535188)

[3. Actors and Use Case Diagram 6](#_Toc122535189)

[3.1. Actors in WEBREM 6](#_Toc122535190)

[3.2. WEBREM Use Case Diagram 7](#_Toc122535191)

[3.3. WEBREM User Activity Diagram 7](#_Toc122535192)

[3.4. WEBREM Admin-Moderators Activity Diagram 8](#_Toc122535193)

[3.5. WEBREM Use Case Description Table 9](#_Toc122535194)

[4. Non-Functional Requirements 9](#_Toc122535195)

[4.1. Development Environment 9](#_Toc122535196)

[4.2. Security 9](#_Toc122535197)

[4.3. Multiple Language Support 10](#_Toc122535198)

[4.4. WEBREM Administrators 10](#_Toc122535199)

[4.5. Logging 10](#_Toc122535200)

[4.6. Testing 10](#_Toc122535201)

[4.7. Business Rules 10](#_Toc122535202)

[4.8. Copyright Rules 10](#_Toc122535203)

[5. References 11](#_Toc122535204)

# Table of Figures

[Figure 1. Use Case Diagram 7](#_Toc122535205)

[Figure 2. User Activity Diagram 8](#_Toc122535206)

[Figure 3. Admin-Moderators Activity Diagram 8](#_Toc122535207)

# 1. Introduction to WEBREM

The purpose of the software project is developing a web application in an object-oriented programming language Java, in LINUX/Windows/MAC and MySQL or PostgreSQL environment.

WEBREM is Remembrances or Condolences Web Service. The goal of the project is to develop the WEBREM service, which has functions such as uploading the memories about important people or institutions as texts, photographs, audio recordings or video recordings to a database with a web interface, storing them in the database in a moderated way, indexing and updating them in a way that can be easily searched and found, and put them at the access of users.

# 2. WEBREM Requirements - Use Case Table

|  |  |  |
| --- | --- | --- |
| No. | Requirements | Use Case |
| 1 | To enter the system and share my memories of famous people or institutions, I need to be registered. After entering my e-mail, country, and Telephone number, I can set my password and register. Identity verification by e-mail and mobile phone is required for all registered users. Users need to accept personal data protection authority article via checkbox to create an account. | Create an account |
| 2 | After registration, the user can activate their account and start using the application with the activation link sent to their registered mail account as a result of mail verification. | Activate account |
| 3 | All WEBREM Users will be able to view all pages opened in the WEBREM service through shared links. However, non-registered WEBREM users will only be able to preview the memories on the opened page. To see the entire shared memory, it is necessary to be a WEBREM registered user. | Surf in the site and make queries within the site |
| 4 | The registered user will fill out a form to sign in fill the email and password boxes and push the sign in button | Sign in to WEBREM |
| 5 | Button I FORGOT MY PASSWORD should mail the password of the user to his/her first entered email address (not the address entered on update screen). If user has forgotten his/her initial email address, there is no chance that he/she will retrieve the password | Update registration information |
| 6 | Registered user can upload memories about a particular person/institution. Registered user will be able to share memories by pressing the SHARE button. | Share memories |
| 7 | Registered WEBREM users will be able to update their memories on request by using the UPDATE button. WEBREM The registered user will be able to delete the shared memory during the update with the DELETE button | Update shared memories |
| 8 | All WEBREM Users will be able to browse through the memories stored in the WEBREM service, make searches about who has written/uploaded what or who has written/uploaded what, and will be able to download the memories in the form of text, photo, sound recording, video recording to their own computer by pressing download button | Download the memories |
| 9 | Registered User in order to create memorial page about a particular person/institution. Registered user must request to share memories. The user fills out an application form about the deceased famous person or institution that he/she wants to be opened, and the moderator examines and approves this application. | Create a memorial page |
| 10 | Moderator reports to the admin about users who make inappropriate comments, upload images, and behave. | Report users to admin |
| 11 | After the Master Moderator reviews and approves the application, one or more Moderators will be appointed for that person/institution, and basic information about that person/institution will be displayed on the WEBREM service. | Assign moderator for new pages |
| 12 | After evaluating the report about the user, it decides the steps to be taken. | Ban users |
| 13 | The user fills out an application form about the deceased famous person or institution that he/she wants to be opened, and the Master Moderator examines and approves this application. | Check Page Opening Form |
| 14 | Master admin assigns admin and master moderator. Master admin can change admin and master moderator later. | Assign admin and master moderator |

# 3. Actors and Use Case Diagram

# 3.1. Actors in WEBREM

|  |  |
| --- | --- |
| Actor | Description |
| Registered User | The user currently registered in WEBREM can perform any function that a user can do in the system. |
| Non-Registered User | A user who is not registered with WEBREM, unlike registered users, they only have access to browse the system. |
| Master Admin | It is the master admin of the system. At the beginning of the system, the master admin is assigned. can access everything in the system. |
| Admin | It is the admin of the system. Appointed by the master admin. It can ban users on the system. |
| Master Moderator | It is the master moderator of the system. Appointed by the master admin. It can assign moderator for new pages and check opening page form. |
| Moderator | It is the moderator of the system. Appointed by the master moderator. It can report users to admin. |

# 3.2. WEBREM Use Case Diagram

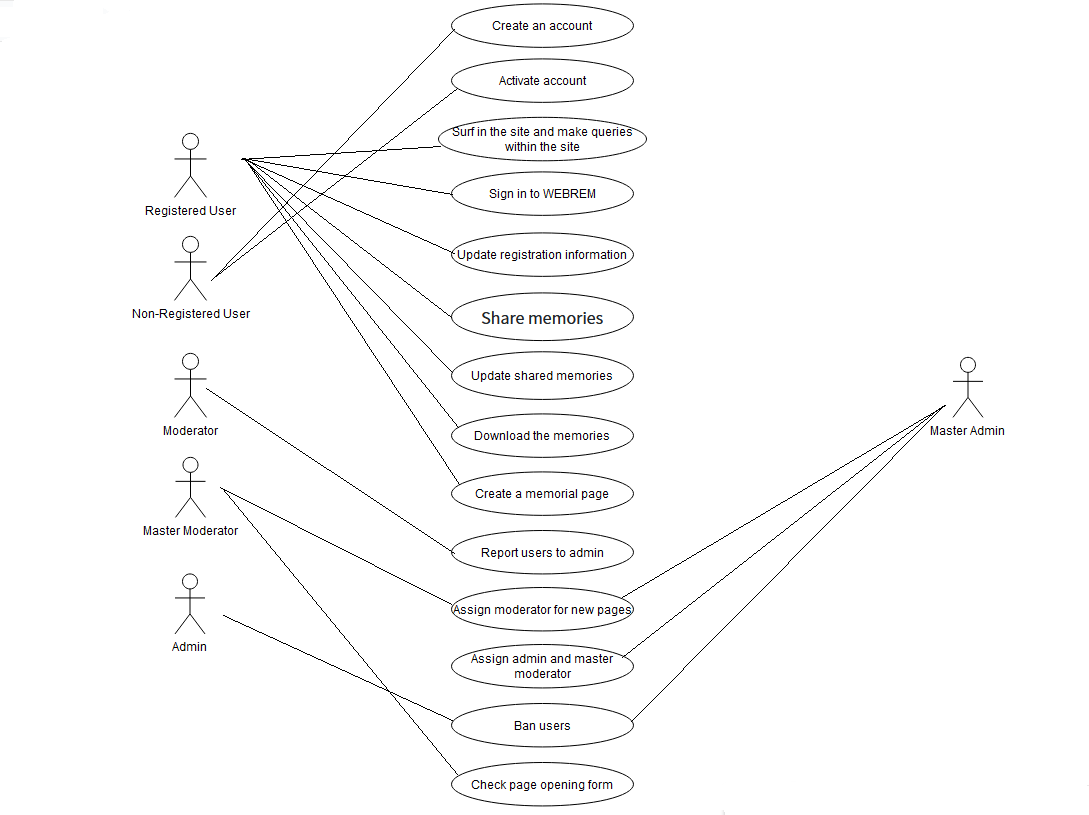


Figure 2. Use Case Diagram

# 3.3. WEBREM User Activity Diagram

Diagram

Description automatically generated

Figure 3. User Activity Diagram

# 3.4. WEBREM Admin-Moderators Activity Diagram

Diagram

Description automatically generated

Figure 4. Admin-Moderators Activity Diagram

# 3.5. WEBREM Use Case Description Table

|  |  |  |
| --- | --- | --- |
| No. | Use Case | Description |
| 1 | Create an account | This function helps the registered users to create account. |
| 2 | Activate account | When the user creates his/her account then this function helps the registered users to activate his/her account. |
| 3 | Surf in the site and make queries within the site | Registered or non-registered user make queries within the site by this function. |
| 4 | Sign in to WEBREM | When registered user activate his/her account. This function can sign in him/her. |
| 5 | Update registration information | Registered user can update registration information by this function. |
| 6 | Share memories | Registered user can share memories by this function. |
| 7 | Update shared memories | Registered user can update shared memories by this function. |
| 8 | Download the memories | Registered user can download shared memories by this function. |
| 9 | Create a memorial page | Registered user can create a memorial page by this function. |
| 10 | Report users to admin | Moderator can report user to admin by this function. |
| 11 | Assign moderator for new pages | Master moderator and master admin can assign moderator for new pages by this function. |
| 12 | Ban users | Master admin and admin can ban reported users by this function. |
| 13 | Check Page Opening Form | Master moderator can check page opening for by this function. |
| 14 | Assign admin and master moderator | Master admin can assign admin and master moderator by this function. |

# 4. Non-Functional Requirements

# 4.1. Development Environment

The software will be developed in React and Java. The database development and operation environment are any PostgreSQL and MySQL.

# 4.2. Security

All client-server operations through the Internet will be through SSL to provide security. All sensitive information regarding the users must be in encrypted form so that sensitive information cannot be accessed by any OS and/or MySQL sysadmins.

# 4.3. Multiple Language Support

WEBREM web application can support Turkish and English. WEBREM service menus will be in Turkish and English. Any user will be able to choose one of these two languages at any time and the web menus will be instantly displayed in the selected language.

# 4.4. WEBREM Administrators

When the server side of the WEBREM web service is first installed and/or started with a new and empty database, it should allow the WEBREM name, WEBREM web address and up to 10 administrator names, e-mail addresses and passwords to be entered. It is mandatory to define a WEBREM Master Admin when installing the system. The first administrator in the list of all WEBREM administrators is the main WEBREM administrator. WEBREM identification is made according to the WEBREM web address.

# 4.5. Logging

Actions of all WEBREM administrators regarding participant deletion, participant name, last name, email updates must be logged to a file and info about such actions must be emailed to the first administrator in the list of all WEBREM administrators, the WEBREM administrator.

# 4.6. Testing

The test document must cover the details of testing the correct operation and boundary conditions of each requirement 3 Section 2. The details of the testing document will be produced during the design. No acceptance test is specified as part of the requirements specification.

# 4.7. Business Rules

* The admin cannot remove or open the page of the famous person or institution before moderators prepares a report.
* Admin cannot ban users without moderators reporting.
* If users’ information is shared or illegally sold, necessary legal actions will be taken.
* Only admin and master admin can unban users.
* The admin can give suggestions to the moderator about preparing a report.

# 4.8. Copyright Rules

* Wikipedia rules will be the basis for copyrights.

# 5. References

* ANSI/IEEE Std 830-1984.
* CWA: A Conference Web Application Requirements Specifications Document, Revision 0.3, in file Caglayan-CWA-RSD-2022-10-29-Rev-0.3.doc.
* CWA: A Conference Web Application Requirements Specifications Document, Revision 1.3, in file Caglayan-CWA-RSD-2020-11-18-Rev-1.3.doc.

# APPENDIX B: DESIGN SPECIFICATION DOCUMENT

**COMP4920 Senior Design Project II, Spring 2023**

**Advisor: Prof. Dr. Mehmet Ufuk Çağlayan**

**WEBREM: A Remembrances or Condolences Web Service**

**Requirements Specifications Document**

**WEBREM**

**High Level Design**

**Design Specifications Document**

**Revision 2.0**

**19.03.2023**

**By:**

**Atakan Aktakka, 18070001032**

**Alican Daşdemir, 18070001041**

**Doğukan Tez, 18070001051**

**Kadir Ülge, 18070001014**

# Revision History

|  |  |  |
| --- | --- | --- |
| **Revision** | **Date** | **Explanation** |
| 1.0 | 08.12.2022 | Initial high-level design |
| 1.1 | 10.12.2022 | 3 Software Subsystem Design added |
| 1.2 | 11.12.2022 | 5 Testing Design and References added |
| 1.3 | 18.12.2022 | 4.1 and 4.2 sections updated |
| 1.4 | 20.12.2022 | Update: WEBREM software system structure |
| 1.5 | 20.12.2022 | Update: WEBREM software system environment |
| 1.6 | 20.12.2022 | Update: WEBREM software system detailed design |
| 2.0 | 19.03.2023 | Update: 1, 2 and 4.1 sections |

# Table of Contents

[Revision History 2](#_Toc130133211)

[Table of Contents 3](#_Toc130133212)

[Table of Figures 4](#_Toc130133213)

[1. Introduction 5](#_Toc130133214)

[2. WEBREM Software Design 5](#_Toc130133215)

[2.1. WEBREM Software System Architecture 5](#_Toc130133216)

[2.2. WEBREM Software System Structure 6](#_Toc130133217)

[2.2.1. Controller Layer 6](#_Toc130133218)

[2.2.2. Model Layer 6](#_Toc130133219)

[2.2.3. View Layer 7](#_Toc130133220)

[2.3. WEBREM Software System Environment 7](#_Toc130133221)

[3. WEBREM Software System Detailed Design: 9](#_Toc130133222)

[3.1. WEBREM Main Class 9](#_Toc130133223)

[3.2. WEBREM Subsystem Classes 9](#_Toc130133224)

[3.2.1. NonRegisteredUser Class 9](#_Toc130133225)

[3.2.2. RegisteredUser Class 9](#_Toc130133226)

[3.2.3. Moderator Class 11](#_Toc130133227)

[3.2.4. MasterModerator Class 11](#_Toc130133228)

[3.2.5. Admin Class 11](#_Toc130133229)

[3.2.6. MasterAdmin Class 11](#_Toc130133230)

[4. Testing Design 12](#_Toc130133231)

[4.1 Testing Strategy 12](#_Toc130133232)

[References 12](#_Toc130133233)

# Table of Figures

[Figure 1. WEBREM Software System Structure 6](#_Toc130133197)

[Figure 2. WEBREM Software System Environment 7](#_Toc130133198)

# 1. Introduction

The purpose of the software project is developing a web application named WEBREM in an object-oriented programming language NodeJS, in LINUX/Windows/MAC and MongoDB environment. WEBREM has some main functions which are listed below.

* Allowing users to upload memories about important people or institutions, in the form of text, photos, audio recordings, or video recordings.
* Storing these memories in a database, in a moderated way to ensure that only appropriate content is included.
* Providing a searchable index of the memories, allowing users to easily find and access the memories they are looking for.
* Providing a web-based user interface for accessing and interacting with the memories in the database.

The design is based on WEBREM Requirements Specification Document, Revision 1.5, in file WEBREM -RSD-Rev-1.5.doc [1].

The software architecture and overall high-level structure of WEBREM Service are given in Section 2, 3 and 4 and design details of all application functions and the user interface in terms of methods of all classes are given in Section 5 of this document.

# 2. WEBREM Software Design

# 2.1. WEBREM Software System Architecture

MVC architecture is the most suitable option for this system architecture. MVC (Model-View-Controller) architecture has several benefits, including separation of concerns, testability, reusability, maintainability, scalability, and flexibility. This architecture divides an application into three components, allowing for independent development and easier management of complex applications. MVC promotes code reusability, making it easier to build new applications or add new features to existing ones. Additionally, it allows developers to choose different technologies for each component, providing flexibility and scalability. Overall, MVC provides a structured approach to developing applications that is reliable, robust, and scalable.

# 2.2. WEBREM Software System Structure

diyagram içeren bir resim

Açıklama otomatik olarak oluşturuldu

Figure 5. WEBREM Software System Structure

WEBREM services use an MVC architecture that consists of three layers: the controller layer, model layer, and view layer.

# 2.2.1. Controller Layer

The controller layer is responsible for implementing the core functionality of the application and contains classes and methods that handle tasks such as creating and updating tasks, managing user accounts, and generating reports.

# 2.2.2. Model Layer

The model layer is responsible for managing data storage and retrieval. It uses a database to store information about tasks, users, and other aspects of the application. It provides the controller layer with methods for accessing and updating this data.

# 2.2.3. View Layer

The view layer defines the interface that the other layers in the system use to communicate with one another. It typically defines a set of functions, methods, or classes that the other layers can use to access the services provided by the layers below it. This layer acts as an interface between the different components of the system and plays a crucial role in ensuring that the different components can work together effectively.

# 2.3. WEBREM Software System Environment

A WEBREM service is designed to run on a browser that is running the Windows/linux/Mac operating system. The service is built using the ExpressJS, NodeJS and it uses a MongoDB database to store data.

The software system environment for this service includes Linux/windows/Mac operating system, ExpressJS framework, NodeJS and MongoDB database. It also includes all external services to which the service is connected, such as a database service for images, sound recordings, text messages etc. sent by the user, or a database service for downloading the desired post.

The user makes a request to the program by communicating from the view layer. The controller layer receives this appropriate response from the model layer after the necessary controls and transmits it to the view layer.

In this environment, the service must be able to function properly on the Windows/linux/Mac operating system, using the ExpressJS framework, NodeJS and the MongoDB database. It must also be able to interact with any external services that it depends on, such as the posting and downloading services.

To ensure that the web service functions properly in this environment, the developers may need to conduct tests to verify that the application works as intended on the Windows/linux/Mac operating system, with the ExpressJS framework, NodeJS and the MongoDB database. They may also need to test the service's interactions with any external services to ensure that they are working properly.

Overall, the software system environment is the set of external factors and conditions that can impact the functioning and performance of a software system. It is important to consider the environment when designing a software system to ensure that it can function properly in its intended environment.

**WEBREM Software System Back-End Environment**

* + - NodeJS 18.13.0v

**WEBREM Software System Front-End Environment**

* + - ReactJS 18
    - Html & CSS
    - Javascript

**WEBREM Software System Database Environment**

* + - MongoDB 6.04v

**WEBREM Software Testing Environment**

* + - JestJS 29.5.0v
    - Puppeteer

**WEBREM Software System Frameworks Environment**

* + - ExpressJS 4.18.2v

# 3. WEBREM Software System Detailed Design:

# 3.1. WEBREM Main Class

Main class is the main program or main process for developments in WEBREM.

# 3.2. WEBREM Subsystem Classes

# 3.2.1. NonRegisteredUser Class

**NonRegisteredUser Class - Method createAnAccount()**

* This method takes NonRegisteredUser class as an input parameter, then creates account for non-registered users.
* If non-registered user wants to create an account, the user will fill out a form containing the following information. Fields that are marked with a \* are mandatory (There should be a message about this to the user).
  + Email\*
  + Country\*
  + Password\*
  + Last name\*
  + Name\*
  + GSM Phone\*
  + Username

Sign-up Buttons: Submit and Cancel

**NonRegisteredUser Class - Method activateAccount()**

* This method takes NonRegisteredUser class as an input parameter, then sends an activation e-mail to non-registered user.
* After non-registered user creates an account, an e-mail is sent to the user to activate the account.

# 3.2.2. RegisteredUser Class

**RegisteredUser Class - Method makeQuery()**

* Registered user makes queries site by search bar and surfs within the site by this function.

**RegisteredUser Class - Method signIn()**

* This method takes RegisteredUser class as an input parameter, then logs user into the system.
* When registered user activates his/her account, this function can sign in him/her. User signs in by clicking sign in button.

**RegisteredUser Class - Method updateRegistrationInfo()**

* This method takes RegisteredUser class as an input parameter, then updates registered user’s information.
* The registered user will fill out a form containing the following information to update registration information.
  + Email\*
  + Password\*
  + Buttons: SHOW MY REGISTRATION INFO, CANCEL and I FORGOT MY PASSWORD

**RegisteredUser Class - Method shareMemories()**

* This method takes RegisteredUser and Memory classes as an input parameter, then shares memory.
* The user will fill out a form containing the following information to share memories. At least one field must be filled (There should be a message about this to the user).
  + Add Photo
  + Add Text (memories)
  + Audio recordings
  + Video recordings

**RegisteredUser Class - Method updateSharedMemories()**

* This method takes RegisteredUser and Memory classes as an input parameter, then updates shared memories.
* Registered WEBREM users will be able to update their memories on request by using the UPDATE button. WEBREM The registered user will be able to delete the shared memory during the update with the DELETE button. This action does not go through moderator approval.

The registered user will fill out a form containing the following information.

* + Photo/Text/Audio Recordings/Video Recordings\*

Buttons: UPDATE, DELETE and CANCEL

**RegisteredUser Class - Method downloadMemories()**

* This method takes Memory class as an input parameter, then downloads shared memories.
* The registered user can download the shared memories by pressing the download button whenever he/she wants.

**RegisteredUser Class - Method createMemorialPage()**

* This method takes RegisteredUser class as an input parameter, then creates memorial page.
* The user fills out an application form about the deceased famous person or institution that he/she wants to be opened, and the Master Moderator examines and approves this application.
* The user will fill out a form containing the following information. Fields that are marked with a \* are mandatory (There should be a message about this to the user).

(Their means here famous person or institutions).

For persons,

* + Their First Name\*
  + Their Last Name\*
  + Their Date of Birth (year – month - day) \*
  + Their Date of Death (year – month - day) \*
  + Their City of Birth\*
  + Gender (female – male – other) \*
  + Add Photo
  + Add Information\*

For institutions,

* + Their Name\*
  + Their Date of Establish (year – month - day) \*
  + Their Date of Bankruptcy (year – month - day) \*
  + Their City of Birth\*
  + Add Photo (logo)
  + Add Information\*

# 3.2.3. Moderator Class

**Moderator Class - Method report()**

* This method takes RegisteredUser and Memory classes as an input parameter, then reports users and memories to admin.
* Moderator reports to the admin about users who make inappropriate comments, upload images, and behave.

# 3.2.4. MasterModerator Class

**MasterModerator Class - Method assignModerator()**

* This method takes Moderator and MemorialPage classes as an input parameter, then assigns moderator to memorial pages.
* After the Master Moderator reviews and approves the application, one or more Moderators will be assigned for that person/institution page, and basic information about that person/institution will be displayed on the WEBREM service.

**MasterModerator Class - Method checkPageOpeningForm()**

* This method takes MemorialPage class as an input parameter, then brings page opening form to master moderator for checking.
* The user fills out an application form about the deceased famous person or institution that he/she wants to be opened, and the master moderator examines and approves this application.

# 3.2.5. Admin Class

**Admin Class - Method banUsers()**

* This method takes RegisteredUser class as an input parameter, then bans user.
* After evaluating the report about the user, it decides to ban or not.

# 3.2.6. MasterAdmin Class

**MasterAdmin Class - Method assignAdminMasterModerator()**

* This method takes Admin, MasterModerator and MemorialPage classes as an input parameter, then assigns admin and master moderator for memorial pages.
* Master admin can change admin and master moderator later.

These classes are written to give information about the general architecture and operation of the project. More detailed explanations will be added during the COMP4920 course.

# 4. Testing Design

The purpose of this testing design is to describe the approach and strategies that will be used to test the web-based task management services.

# 4.1 Testing Strategy

* Unit testing: Jest will be used to perform unit testing on individual components or classes in the service to verify their correct functioning in isolation. This will involve creating test cases that exercise the different methods and features of each component and verifying that they produce the expected results.
* Integration testing: Jest will be used to perform integration testing to verify that the different components of the service work together correctly. This will involve testing the interactions between the view layer, the model layer, and the controller layer.
* End-to-end testing: Jest and Puppeteer will be used to perform end-to-end testing to verify the overall functioning of the service as a whole. This will involve testing the service from the perspective of a user, simulating different user actions and verifying that the service responds as expected.

# References

1. System Requirements Specification Document, Revision 2.0, in file WEBREM-RSD-2022-12-19-Rev-2.1.doc.
2. IEEE Std 1016-2009

# APPENDIX C: PRODUCT MANUAL

**COMP4920 Senior Design Project II, Spring 2023**

**Advisor: Prof. Dr. Mehmet Ufuk Çağlayan**

**WEBREM: A Remembrances or Condolences Web Service**

**Requirements Specifications Document**

**Product Manual**

**Revision 1.0**

**04.04.2023**

**By:**

**Atakan Aktakka, 18070001032**

**Alican Daşdemir, 18070001041**

**Doğukan Tez, 18070001051**

**Kadir Ülge, 18070001014**

# Revision History

|  |  |  |
| --- | --- | --- |
| **Revision** | **Date** | **Explanation** |
| 1.0 | 04.04.2023 | All sections update |

# Table of Contents

[Revision History 2](#_Toc131538390)

[Table of Contents 3](#_Toc131538391)

[1. Introduction 4](#_Toc131538392)

[2. WEBREM Software Subsystem Implementation 4](#_Toc131538393)

[2.1. Source Code 4](#_Toc131538394)

[2.1.1 Controller Layer 4](#_Toc131538395)

[2.1.2 Model Layer 4](#_Toc131538396)

[2.1.3 View Layer 4](#_Toc131538397)

[2.2 Executable Organization 4](#_Toc131538398)

[2.2.1 System Environment 5](#_Toc131538399)

[2.2.2 User Interaction 5](#_Toc131538400)

[2.2.3 Testing 5](#_Toc131538401)

[2.3. Software Development Tools 5](#_Toc131538402)

[2.4. Hardware and System Software Platform 5](#_Toc131538403)

[3. WEBREM Software Subsystem Testing 6](#_Toc131538404)

[4. WEBREM Installation, Configuration and Operation 6](#_Toc131538405)

[References 7](#_Toc131538406)

# 1. Introduction

WEBREM is a web application developed using NodeJS, which aims to provide a platform for users to store and share memories about important people or institutions. The application is designed to be simple and easy-to-use, allowing users to upload memories in various formats, including text, photos, audio recordings, or video recordings.

The memories are stored in a moderated way to ensure that only appropriate content is included. This ensures that the platform remains safe and appropriate for users of all ages. The application also provides a searchable index of the memories, allowing users to easily find and access the memories they are looking for.

WEBREM is developed to work on LINUX/Windows/MAC operating systems and uses MongoDB as the database. The web-based user interface is designed to be intuitive, making it easy for users to access and interact with the memories in the database.

This product manual report aims to provide detailed information on how to use WEBREM and its various features. It includes step-by-step instructions on how to upload memories, search for memories, and interact with the memories in the database. Additionally, the report provides information on the system requirements, installation instructions, and troubleshooting tips.

# 2. WEBREM Software Subsystem Implementation

# 2.1. Source Code

The source code for the WEBREM system is organized into three main layers: the controller layer, the model layer, and the view layer.

# 2.1.1 Controller Layer

The controller layer is responsible for implementing the core functionality of the application. It contains classes and methods that handle tasks such as creating and updating tasks, managing user accounts, and generating reports. The controller layer is implemented using ExpressJS and NodeJS frameworks.

# 2.1.2 Model Layer

The model layer is responsible for managing data storage and retrieval. It uses a MongoDB database to store information about tasks, users, and other aspects of the application. It provides the controller layer with methods for accessing and updating this data.

# 2.1.3 View Layer

The view layer defines the interface that the other layers in the system use to communicate with one another. It typically defines a set of functions, methods, or classes that the other layers can use to access the services provided by the layers below it. This layer acts as an interface between the different components of the system and plays a crucial role in ensuring that the different components can work together effectively. The view layer is implemented using HTML, CSS, and JavaScript.

# 2.2 Executable Organization

The WEBREM system is designed to run on a browser that is running the Windows/Linux/Mac operating system. The service is built using the ExpressJS and NodeJS frameworks, and it uses a MongoDB database to store data.

# 2.2.1 System Environment

The software system environment for this service includes the Windows/Linux/Mac operating system, ExpressJS framework, NodeJS, and MongoDB database. It also includes all external services to which the service is connected, such as a database service for images, sound recordings, text messages, etc., sent by the user, or a database service for downloading the desired post.

# 2.2.2 User Interaction

The user makes a request to the program by communicating from the view layer. The controller layer receives this request, performs the necessary controls, and sends an appropriate response back to the view layer.

# 2.2.3 Testing

To ensure that the web service functions properly in this environment, the developers may need to conduct tests to verify that the application works as intended on the Windows/Linux/Mac operating system, with the ExpressJS framework, NodeJS, and the MongoDB database. They may also need to test the service's interactions with any external services to ensure that they are working correctly.

In summary, the WEBREM system is organized into three main layers: the controller layer, the model layer, and the view layer. The system is designed to run on a browser that is running the Windows/Linux/Mac operating system and uses the ExpressJS and NodeJS frameworks, and a MongoDB database. The user interacts with the program through the view layer, and the controller layer responds to user requests. The developers need to conduct tests to ensure that the system works correctly in its intended environment.

# 2.3. Software Development Tools

The following software tools were used during the development and testing of WEBREM Software Subsystem:

* ReactJS
* Visual Studio
* GitHub for version control
* NodeJS
* Github Desktop
* Visual Studio
* Postman
* MongoDB Compass

for database managementused during your project implementation and testing, that is, your source code production, testing, configuration, etc.

# 2.4. Hardware and System Software Platform

WEBREM Software Subsystem is designed to run on a standard desktop computer

# 3. WEBREM Software Subsystem Testing

WEBREM software subsystem testing will be performed to ensure that the different software components and subsystems work correctly and reliably. The testing approach will follow a comprehensive and iterative process, which includes the following steps:

1. Test Planning: A detailed test plan will be developed, outlining the testing objectives, scope, and approach. This will involve identifying the subsystems and components to be tested, the testing tools and techniques to be used, and the test data required.
2. Unit Testing: Jest will be used to perform unit testing on individual software components, such as modules, functions, and classes. This will involve creating test cases that exercise the different methods and features of each component and verifying that they produce the expected results.
3. Integration Testing: Jest will be used to perform integration testing to verify that the different software subsystems and components work together correctly. This will involve testing the interactions between the different layers of the application, including the view layer, the model layer, and the controller layer.
4. System Testing: End-to-end testing will be performed using Jest and Puppeteer to verify the overall functioning of the software subsystems as a whole. This will involve testing the subsystems from the perspective of a user, simulating different user actions, and verifying that the subsystems respond as expected.

# 4. WEBREM Installation, Configuration and Operation

**Installation and Configuration:**

1. First, download the latest version of WEBREM from our website.
2. Unzip the downloaded file and upload it to your web server using an FTP client.
3. Create a new MySQL database and user on your web server.
4. Navigate to the installation URL in your web browser to start the installation wizard.
5. Follow the instructions in the wizard to configure the database connection and other system settings.

Note: The installation and configuration process may require advanced knowledge of web server administration and database management.

**Operation:** Once WEBREM is installed and configured, users can access the system by navigating to the site's URL in their web browser. The user interface is similar to Instagram, with a timeline displaying the latest posts from the users the user follows. Users can view and interact with posts by liking, commenting, and sharing them.

To create a new post, the user can click the "New Post" button and enter the post content, including text, images, and videos. They can also add hashtags to make their post more discoverable.

In addition to creating posts, users can also edit their profile information, view their follower/following list, and search for other users and posts by keyword or hashtag.

**Error Messages:** If an error occurs during installation or operation, WEBREM will display an error message indicating the issue. The error message will provide a brief explanation of the problem and suggest a solution. Common errors include incorrect database connection settings, insufficient file permissions, and incompatible web server software.

**Help Facility:** WEBREM includes an extensive online help facility accessible from the user interface. The help facility provides detailed instructions and explanations for all system features and functions. Users can access the help facility by clicking the "Help" button in the top menu.

# References

1. System Requirements Specification Document, Revision 2.0, in file WEBREM-RSD-2022-12-19-Rev-2.1.doc.
2. IEEE Std 1016-2009

# APPENDIX D: SOURCE CODE/EXECUTABLES/SCRIPTS IN CD/DVD