CHATRASAHAYA DEVELOPER GUIDE

VERSION 1.0

Team Details :

Vikas Sabbi (2836919)

Keshav Singhal (2822290)

Tanmay Rajgor   (2827074)

1.0 Introduction and Background

1.1 Introduction

This software will help the Chatrasahaya to be more efficient in submitting of their requests such as money,food and other needs…and tracking of submission on the maps. It also sends the mails to the internal team to check the submissions quickly. The purpose of this project is to computerize all details regarding submissions from request us,Users can submit their required needs likes education,financial,Health…from the **Request Us for Self /For Others** Page by using some fields such as **Name** -- Full Name of the requester , **Phone Number** -- Phone/mobile Number of the Requester , **Location** -- The place of the request where books,chairs,tables are needed for the schools , **Category--** Its was type of the request such as education,financial,Health **Description** -- Its was the description/overview of the needs , **Aaadhar Number** -Its was unique identity number of the requester, **Documents** --- Proof of the documents needs to be uploaded here.

Modules/Features as list below:

- **Home**-It helps to access the other modules in the application from the home page
- **About**-It helps to know the chatrasahaya website details and activities
- **Join Us**-Its helps the people/end users to join in the team by submitting the form
- **Programs** -It helps to know the chatrasahaya programs conducted
- **Contact Us**-Its was useful to the people/end users to contact the team for any issues/queries
- **Request Us-**Its was useful to request the service from the chatrasahaya by submitting the form based on the category
- **Search** -Its was use to search any features or modules.

1.2 Objectives

As state earlier, chatrasahaya is trust related management software. That said, chatrasahaya implements most of the expected functionality of trust services management software. It does not implement the functionality found in a product like Blackboard, instead it focuses more on a subset of the functionality provided by a product like provides the following functionality:

- End/Users can request their services like category food,health,education:
    - Request Us Page Submission

- Mail Notification
  Map and Count View

- Internal Team can manage their Programs by:
  - Scheduling of the Programs
  - Checking of the Services requested by the End/Users
  - Viewing of the Website whenever needs to check the map & count view

- Internal users can manage their Join Us & Contact Us:
  - Internal users can check the emails where new internal team can join into the team
  - Internal users can check the emails receives from the end users related to contact us page such as issues/queries raised by the end users

1.3 Risks

This section discusses project risks and the approach to managing them.

Project risks are events or circumstances that can potentially have a negative impact on a project's objectives, such as its scope, schedule, budget, or quality. Identifying and assessing project risks is a crucial part of project management.

- ❖ Scope Creep
- ❖ Resource Constraints
- ❖ Schedule Delays
- ❖ Quality Issues
- ❖ Technical Challenges
- ❖ Communication Breakdown

Name of risk	**probability**	**Impact**	**RM3 pointer**
Scope Creep	less	Impact on Design of the requirement	Initial
Resource Constraints	less	Impact on Schedule Timings	Optimized

Schedule Delays	medium	Impact on Timely Deliverable	Defined
Quality Issues	medium	Impact on Output of the requirement	Optimized
Technical Challenges	medium	Impact on Proper Coding & testing	Initial
Communication Breakdown	medium	Impact on Design of the requirement	Initial

2.0 Architecture

2.0 Software Architecture

When designing chatrasahaya , we explicitly chose to utilize a Layered architecture. I had briefly considered employing the MVC architecture pattern, however time constraints on the project alongside my inexperience with the pattern in Java led to me choosing a Layered architecture instead. This has led to some curious implementation details whereby the system mimics an MVC architecture (stressing that this is not an implementation of MVC).

The architecture diagram for chatrasahaya is shown below. Proceeding the architecture diagram is a description of all the components that make up the diagram.

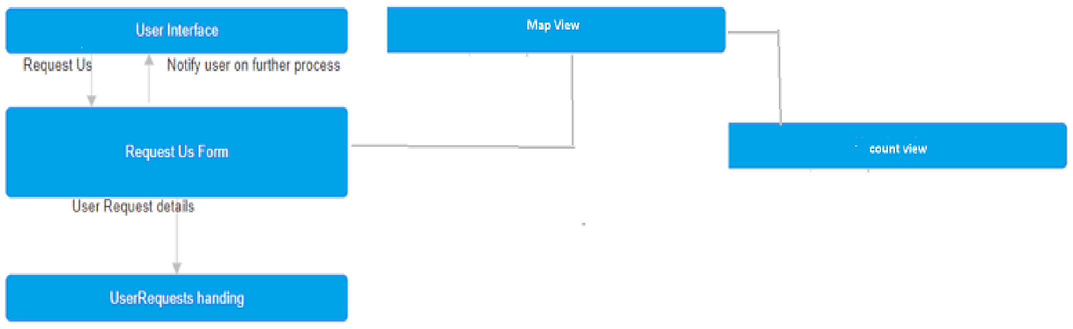
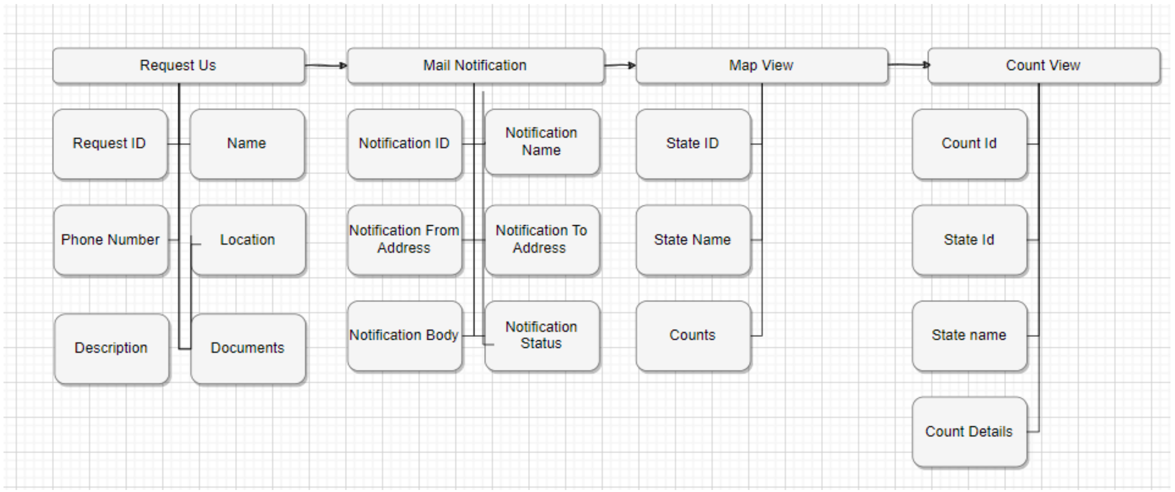


Figure 1: chatrasahaya Architecture Diagra

2.1 Database Design

chatrasahaya follows a relational database model that approximates third-normal form. There were some decisions regarding certain tables where I though it would be more prudent to just repeat data in a column rather than create a small (3-7 row) table for it (such as with days, and years). This data is checked for correctness before it is entered into the database anyway, so no invalid values are can exist in the affected tables.

chatrasahaya database design schema is listed below:



Following is a description of the tables in presented in the ERD as well as their relations:

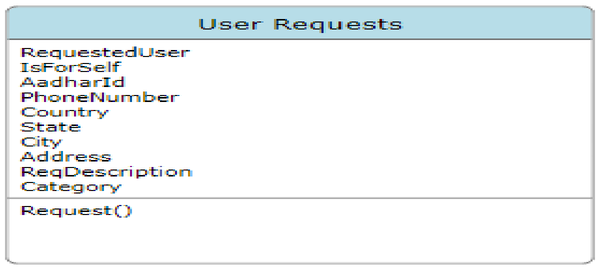
User Requests	**Mail Notification**	**Map View**	**Count View**
Customer ID: Int	Notification ID : Int	State ID: Int	Count ID: Int
Name: String	Notification Name: String	State Name: String	State ID: Int
Phone Number: BigInt	Notification To Address : String	Count: Int	State Name:String
Location: String	Notification From Address : String		Count Details : Table with Request Name,Request Phone Number,Location,Description,Documents View
Description: String	Notification Body: String		
Documents: Img,Pdf Uploads	Notification Status : String		

Table No 4: Database Design

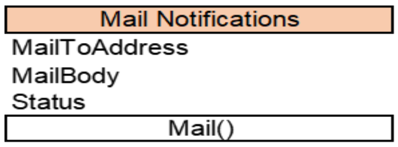
2.2 Class Level Design

This section explains every class present in chatrasahaya , as well as the interactions between them. I will also provide rationale behind the design of several of the more complicated classes in the system. The classes listed below are organized as they appear in the Visual Studio solution: First by folder, then by class.

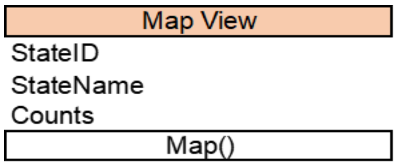
Here is the below is the Class diagram for the User Request



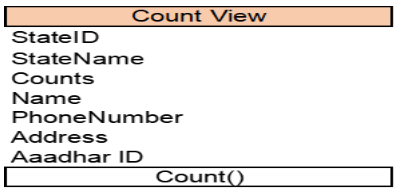
Here is the below is the Class diagram for the Mail Notification



Here is the below is the Class diagram for the Map View



Here is the below is the Class diagram for the Count View



2.2.1 Cache

This folder contains classes related to an in-memory cache.

2.2.1.1 Cache Item

Implements a key-value pair cache item. Both the key and value are generic for increased flexibility. A cache item implements methods for accessing and mutating both the key and value. It is the primary class utilized by the MemoryCache class.

2.2.1.2 MemoryCache

Implements an in-memory cache that stores key-value pair CacheItems. The MemoryCache class implements an aging mechanism for paging. It also utilizes the C# delegate design model to dynamically invoke a method handler for page events. This provides a direct benefit for classes that use the MemoryCache class in that they can catch objects as they are paged out of cache and handle them appropriately. The MemoryCache class employs all of the standard operations one would normally expect from a software-based cache.

2.2.2 Managers

This folder contains classes related to managing the interactions between the front-end graphical user interface and the back-end database. These classes are also used to cache the classes they are responsible for maintaining.

2.2.2.1 Maps Manager

This class is responsible for providing access to and maintaining synchronicity across maps and sections. It provides middleware versions of the course and section methods found in the DatabaseManager class. The GUI interacts with this class to request information regarding courses and sections.

2.2.2.2 DatabaseManager

This class does all the heavy lifting involved with interacting with the database. This is the DatabaseManagers sole responsibility. Most exceptions regarding the data classes can usually be traced back to this class. The CourseManager, UniversityManager, and UserManager classes make use of this class to indirectly interact with the database. Many of the methods in this class are wrappers around SQL code.

2.3 Software Interfaces

chatrasahaya utilizes several outside software packages to accomplish its objectives. When installing chatrasahaya , the maintainer must be certain (in most cases) that the following software packages are installed on the hosting system:

- Hosted Wordpress Website: Offically Hosted Wordpress details should have to access the application for the furture changes. chatrasahaya was implemented inside Hosted Wordpress and Python. It is recommended that the maintainer utilize at least this version of the software,
- SQL Server Express: This software ships with Visual Studio (if the proper packages are installed with it). SQL Server Express was used during development as the database provider for chatrasahaya . It is highly suggested that the maintainer transfer the database schema to a full-fledged SQL Server or Azure instance on dedicated hardware. It is <u>not recommended</u> to run chatrasahaya on the same physical hardware that the database exists on.
- JQuery*: Provides advanced JavaScript functionality to the system and supports the bootstrap theming system.
- Google Maps Api: chatrasahaya utilizes Google Maps API for using the india map to show the counts based on requests and should have google consule developer account and should generate the api key from the google site and can be used in the hosted wordpress application.
- DIA diagramming tool: All of the models for the system were built using DIA. DIA is free software distribute by the GNU project. It is available on both Windows and Linux and is a great alternative to Microsoft Visio. In order to work with the models and physically alter them, you will need DIA installed on your system. I have also taken care
- Github Version Control: Available at https://github.com, this web service hosts the project. The maintainer must be familiar with its operation.

*Note: JQuery and Bootstrap are currently setup to pull from official Content Distribution Networks for the respective software systems. It is not necessary to do anything during installation of the chatrasahaya software to ensure either system will function.

3.0 Conclusion

3.1 Remarks on Implementation

This section contains my remarks regarding the current implementation of chatrasahaya. This implementation is nearly complete (I estimate that ~95% of the functionality outlined in the Requirements Spec. is met). For instance, advisors cannot currently override prerequisites for their advisees. Likewise, I ran out of time to complete the theming of the system. It has partially met the layout specified in the Design Specification, however the arrangement of the controls within the system are not as I had specified.

Also, due to the time constraints and limited time I personally had to work on this project, I made some implementation decisions that are not optimal from a performance standpoint. I have tried to outline these areas within the source code of the system. However, I may have missed some areas. I have tried my best to provide developer comments for every method but there are some methods that I have missed (the stubs are there, they but they are not filled in).

Also, due to the time constraints and limited time its was taken more time to get the Google API Approved from the Google team to integrate into the application for the map view

3.2 Possible Future Improvements

We will plan to the develop the Mobile Application for the same website with the same features one by one