1. PROJECT OVERVIEW

1.1. Introduction to customer care management system

Customer Care Management (CCM) is the process of handling customer interactions, resolving issues, and ensuring customer satisfaction to build long-term relationships. It involves strategies, technologies, and best practices aimed at providing excellent customer service before, during, and after a purchase. Effective customer care management leads to stronger relationships, increased customer retention, and business growth.

1.2. Importance of Customer Care Management

1. Enhances Customer Satisfaction – Ensures customers receive quick and effective assistance.
2. Improves Business Reputation – Positive customer experiences lead to good reviews and word-of-mouth marketing.
3. Increases Revenue – Happy customers often make repeat purchases and spend more.
4.Reduces Customer Churn – Effective care prevents customers from switching to competitors.

1.3. Objectives of a proposed system:

The objective of a proposed system in customer care management is to enhance customer service operations by improving efficiency, responsiveness, and customer satisfaction. Now a days customer services is a key component for many companies. Whether you are serving consumer or other business, excellent customer service will help to ensure profitability. Understanding your customer's needs, as well as their frustrations, means that customer services also provide important feedback about business processes. Great customer services require organizational commitment.

Most business base customers service and meeting customer need and expectation as well as dealing productively with their complaints and suggestions employees who work directly with customers should know their employers' products use good phone and personal etiquette and acknowledge that their personal success depends on the quality of customer service they provide.

1. Automating Customer Interactions – Implementing chatbots, IVR systems, and AI-powered assistants to handle common inquiries and reduce response times.
2. Enhancing Customer Experience – Providing seamless support across multiple channels (phone, email, live chat, social media) for a more personalized experience.
3. Efficient Ticket Management – Automating ticket creation, categorization, and resolution tracking to ensure faster problem-solving.
4. Centralized Customer Data – Maintaining a unified database for customer interactions, preferences, and history to improve service quality.
5. Real-time Performance Analytics – Monitoring agent performance, customer satisfaction scores, and response times to optimize service.
6. Self-Service Options – Offering FAQs, knowledge bases, and community forums to allow customers to find solutions without agent intervention.
7. Integration with CRM & Other Systems – Synchronizing with customer relationship management (CRM) software for better service continuity.
8. Reducing Operational Costs – Streamlining workflows and reducing manual tasks to cut down costs while maintaining service quality.

9. Ensuring Data Security & Compliance – Protecting customer data and ensuring compliance with industry regulations (e.g., GDPR, HIPAA).
10. Scalability & Flexibility – Designing a system that can grow with the business and adapt to new customer service trends and technologies.

1.4. Project category:

A Customer Care Management System (CCMS) can fall under several project categories, depending on its focus and implementation. Here are some possible categories:

1. Customer Relationship Management (CRM)
   - Manages interactions with customers
   - Tracks customer history, feedback, and preferences
   - Example: Salesforce, HubSpot CRM
2. Help Desk & Ticketing System
   - Handles customer complaints and queries
   - Automates ticket creation, tracking, and resolution
   - Example: Zendesk, Freshdesk
3. Call Center Management
   - Manages inbound and outbound customer calls
   - IVR (Interactive Voice Response) and call routing
   - Example: Five9, Avaya
4. Live Chat & Chatbot Support
   - Provides real-time customer support via chat
   - Uses AI chatbots for automated responses
   - Example: Drift, Intercom
5. Feedback & Survey Management
   - Collects customer feedback through surveys
   - Analyzes customer satisfaction and Net Promoter Score (NPS)
   - Example: SurveyMonkey, Qualtrics
6. Knowledge Base & Self-Service Portals
   - Provides FAQs, guides, and troubleshooting articles
   - Reduces workload on support teams
   - Example: Confluence, Document360
7. Social Media Customer Support
   - Manages customer interactions on social media platforms
   - Tracks brand mentions and sentiment analysis
   - Example: Sprout Social, Hootsuite
8. Customer Loyalty & Retention Management
   - Tracks customer engagement and rewards loyal customers
   - Offers personalized deals and promotion
   - Example: Zoho CRM, Loyalty Lion
9. AI-Powered Customer Service Analytics
   - Uses AI to analyze customer trends and behavior
   - Helps in predictive support and decision-making
   - Example: IBM Watson, Google Contact Center AI

1.5. FEATURES:

Complete Account Rebuild

The first thing we do for a new client is completely overhaul their account. Using existing data, as well as our own research and experience, we spend time building a well-structured, high performing customer care services ensure you hit the ground running.

Direct Access to Google

The result in the page is shown directly from goggle. We are not managing anything in showing the result in google. Only the advertising are been managed.

Dedicated Account Managers

Each top click media client is assigned their own personal account manager responsible for both the day running of the account, as well as devising long term account strategies. “From keyword to click to search we monitor and refine at every step. That’s how we continually deliver report for our clients.

Increased Profits

Our experience in the design and implementation of PPC accounts ensures that only the most relevant internet users end up on your site. This can’t help but lead to increased sales.

1.6. Benefits of our CUSTOMER CARE MANAGEMENT (prime care) services

- Managed ads
- Set-it and forget it; we handle everything for you
- Free up you time
- Improvement of customer satisfaction
- Stop wasting money & time
- Start pulling-in low-cost, targeted traffic
- Receive monthly reports detailing all aspects of any changes, alterations and results
- Self-managed Accounts

We take care on CUSTOMER CARE MANAGEMENT (prime care)

- Creation of customer account
- Professional account management
- Structured campaign management
- Optimization of customer’s complain & suggestion
- Budget management
- Solving problem through online
- Improvement of quality score
- Giving suggestion on query

And much more

2.SOFTWARE AND HARDWARE REQUIREMENT

2.1. LANGUAGE REQUIREMENT SPECIFICATION

LANGUAGE REQUIREMENT SPECIFICATION		
1.	Frontend	React, typescript, tailwind CSS
2.	Backend	C#, Asp.net 6.0
3.	Server	Google firebase cloud server
4.	Programming language	C#, typescript, DHTML

TABLE 1

2.2. SOFTWARE REQUIREMENT SPECIFICATION

SOFTWARE REQUIREMENT SPECIFICATION		
1.	IDE	Visual studio 2022
2.	Framework	.net Framework 4.5
3.	Database	Google Firebase 2024
4.	O. S	Window 7,8,10,11, macOS, Linux
5.	Internet software	Chrome, firebox, safari

TABLE 2

2.3. MINIMUM HARDWARE REQUIREMENT

MINIMUM HARDWARE REQUIREMENT		
1.	PROCESSOR	Intel Pentium n series, IGH2
2.	RAM	1GB
3.	HARD DISK	2GB
4.	GRAPHIC	800 x 600, 256 colour recommended 1024 x 768, high colour 16 Bit
5.	OTHER	CD-ROM Driver or DVD-ROM Drive Microsoft mouse or compatible pointing device

TABLE 3

3.SYSTEM ANALYSIS

3.1. MODULARISATION DETAIL

In our project having 3 mains basic Module/user

1. Head office (admin)
2. Agent
3. Customer

Work detail of agent: -

- View own profile
- Change password
- Check complains tickets status
- Check daily complain info
- Check transaction detail
- View analytic and requirements
- View response tickets
- Send a requirement report
- View feedback

Work detail of head office: -

- ✓ View /Edit own profile detail
- ✓ Change password
- ✓ View/edit agent head profile
- ✓ Check customer complain maintenance status
- ✓ View customer complain
- ✓ Send customer complain to admin
- ✓ View feedback
- ✓ View complains and suggestion
- ✓ Send reply
- ✓ View transaction

Work Detail of Customer: -

- ➢ Customer registration
- ➢ Upload own profile info.
- ➢ View/edit own profile
- ➢ Change own profile
- ➢ Check own product status & maintenance report
- ➢ Make online payment
- ➢ View/download driver and software
- ➢ Send feedback
- ➢ Send complain suggestion

3.2 USER INTERFACE DESIGN

- Standard tools used
- Label
- Text Box
- Button
- Link Button
- Hyperlink
- Dropdown List
- Check Box List
- Radio Button List
- Image
- Image Button
- Table
- List Box
- Bulleted List
- File Upload

Data Tools Used

- Data List
- Detail List
- Detail View
- Form View
- Sigl Data Source

Validation control

- Required Field Validator
- Compare Validator
- Range Validator
- Regular Expression Validator
- Validation summary

Login Contral

- Login
- Login View
- Password Recovery
- Login Status
- Login Name
- Create user Wizard
- Change Password

& some other controls to make this site successful……

Much of the success of any Web application depends on the quality of its ser interface. A far as end-users are concerned, the user interface in the application: Users aren't interested in the details of the data model or the design of the data-access classes

In an ASP.NET Web application, the user interface consists of a series of .aspx pages that rendered to the browser using standard HTML. Designing the user interface is simply a matter of deciding which pages are required (and in what sequence) and populating those pages with the appropriate controls

Standard HTML has a surprisingly mined set of user-input controls:

- Buttons
- Text boxes
- Drop-down lists
- Check hoses
- Radio buttons

However, ASP.NET offers many controls that build on these haste controls. For example, you can use a Grid View control to present data from a database in a tabular format

All ASP.NET controls are eventually rendered to the browser, using standard HTML A result, even the most complicated ASP.NET controls are simply composites made of standard HTMI. controls and HTML. formatting elements (such as tables).

Designing the user interface can quickly become the most complicated aspect of a Web application. Although user interface design has no hard-and-fast rules, here are a few guidelines you should keep in mind

- Consider how frequently the user will use cach page and how familiar he or she woll be with the application. If the user works with the same page over and over again all day long, try to make the data entry as efficient as possible. However, if the user will use the page only once in a while, err on the side of making the page self-explanatory so the user doesn't have to struggle to figure out how to use the page
- Remember that the user is in control of the application and users are pretty unpredictable. Users might give up in the middle of a data-entry sequence, or unexpectedly hit the browser's Back button
- Some users like the mouse, others like the keyboard. Don't force your preference in the user: make sure your interface works well for mouse as well as keyboard users.
- Review prototypes of the user-interface design with actual users. Listen to their suggestions seriously. They probably have better ides than you do of what the user interface should look like and how it should behave.
- Study Web sites that you consider to have good interfaces

Designing the Business Rules Layer

Business rules are the portion of a program that implements the business policies dictated by the application. Here are some examples of business rules:

- Should a customer be granted a credit request?
- How much of a discount should be applied to a given order?
- How many copies of Form 10432 J need to be printed?
- How much shipping and handling should be tacked onto an invoice?
- When should an inventory item that is running low on stock be reordered?
- How much sick leave should an employee get before managers wonder whether he or she has been sking rather than staying home sick
- When should an account payable be paid to take advantage of discounts while maximizing float?

The key to designing the business-rules portion of an application is simply to identify the business rules that must be implemented and separate them as much as possible from other parts of the program. That way, if the rules change, only the code that implements the rules needs to be changed.

For example, you might create a class to handle discount policies. Then, you can call methods of this class whenever you need to calculate a customer's discount. If the discount policy changes, the discount class can be updated to reflect the new policy

Ideally, each business rule should be implemented only once, in a single class that's used by each program that needs it. All too often, business policies are implemented over and over again in multiple programs and if the policy changes, dozens of programs need to be updated. (That even hurts to think about, doesn't it?)

Designing the Data Access Layer

Much of the job of designing the Data Access Layer involves designing the database itself Here are some pointers on designing the Data Access Layer:

- For starters, you must decide what database server to use (for example, SQL Server or Oracle).
- You'll need to design the tables that make up the database and determine which columns each table will require.
- You must also decide what basic techniques you'll use to access the data. For example, will you write custom data-access classes that access the database directly. or will you use ASP.NET's SqlDataSource control to access the database? And will you use stored procedures or code the SQL statements used to access the data directly in the application code?

Validation Check

After you create a web form, you should make sure that mandatory fields of the form elements such as login name and password are not left blank, data inserted is correct and is within the specified range. Validation is the method of scrutinizing that the user has entered the correct values in input fields. In HTML you can perform validation either by checking the values at client-side or after submitting the form at the server-side. But these methods in HTML take

lots of time to create and maintain the code. Moreover, if the user has disabled JavaScript then he or she may not receive the message regarding the error.
Therefore, in ASP NET you can use ASP.NET Validation Controls while creating the form and specify what ASP.NET Validation Controls you want to use and to which server control you want bind this. ASP NET Validation Controls are derived from a common base class and share a common set of properties and methods. You just have to drag and drop the ASP NET Validation Control in the web form and write one line of code to describe its functionality

Types of Validation Controls

There are validation controls for specific types of validation, such as range checking of pattern matching The following table lists the validation controls:

Control Name:

Required FieldValidator- Ensures that the user does not skip an entry.

CompareValidator-Compares a user's entry with a constant value or a property value of another control using a comparison operator (less than, equal to, greater than, and so on).

Range Validator- Checks that a user's entry is between specified lower and upper boundaries. You can check ranges within pairs of numbers. alphabetic characters, or dates Boundaries can be expressed as constants.

Regular Expression Validator-Checks that the entry matches a pattern defined by a regular expression. This type of validation allows you to check for predictable sequences of characters, such as those in social security numbers, e-mail addresses, telephone numbers, postal codes, and so on.

CustomValidator- Checks the user's entry using validation logic that you code yourself. This type of validation allows you to check for values derived at run time.

ValidationSummary-Displays the validation errors in summary form for all of the validators on a page.

3.3. IDENTIFICATION OF NEED

User Interface principles in web design

There is much hype surrounding website UI design, and a large variety of tools claiming to help optimize the said interface, however, the most useful piece of theory I have found on the topic is dated 1998 and was written by somebody called Talin, whose home page now a a 404. It was written with reference to program design rather than website design, but I think the principles stated there apply cross-topic
Actually, to me, Talin's list read like a collection of my own disparate observations regarding what works and does not work when it comes to websites, what converts and what doesn't So, while I give the full credit to Talin, I will reiterate the principles and explain my taks on the way they should influence website design.
1.The principle of user profiling

- **Who are your users?**
- **How do they access the website?**
- **What are their goals in coming to your website?**

2. The principle of metaphor
3. The principle of feature exposure
4 The principle of coherence
5. The principle of state visualization
6. The principle of shortcuts
7. The principle of focus
8 The principle of grammar
9. The principle of help
    - **Goal-oriented:**
    - **Descriptive:**
    - **Procedural: "**
    - **Interpretive:**
    - **Navigational:**
10. The principle of safety
11. The principle of context
12. The principle of aesthetics
13 The principle of user testing
14 The principle of humility

OUTPUT DESIGN

Outputs from computer systems are required primarily to communicate the results of processing to users. They are also used to provide a permanent copy of the results for later consultation. The various types of outputs in general are:

- External Outputs, whose destination is outside the organization
- Internal Outputs whose destination is within organization and they are the
- User's main interface with the computer.
- Operational outputs whose use is purely within the computer department.
- Interface outputs, which involve the user in communicating directly with

INPUT DESIGN

Input design is a part of overall system design. The main objectives during the input design is as given below:

- To produce a cost-effective method of input
- To achieve the highest possible level of accuracy
- To ensure that the input is acceptable and understood by the user

ERROR AVOIDANCE

At this stage care is to be taken to ensure that input data remains accurate form the stage at which it is recorded up to the stage in which the data is accepted by the system. This can be achieved only by means of careful control each time the data is handled

ERROR DETECTION

Even though every effort is made to avoid the occurrence of errors, still a small proportion of errors is always likely to occur, these types of errors can be discovered by using validations to check the input data.

DATA VALIDATION

Procedures are designed to detect errors in data at a lower level of detail. Data validations have been included in the system in almost every area where there is a possibility for the user to commit errors. The system will not accept invalid data. Whenever an invalid data is keyed in, the system immediately prompts the user and the user has to again key in the data and the system will accept the data only if the data is correct. Validations have been included where necessary.

The system is designed to be a user friendly one. In other words the system has been designed to communicate effectively with the user. The system has been designed with pop-up menus.

ERROR MESSAGE DESIGN:

The design of error messages is an important part of the user interface design. As user is bound to commit some errors or other while designing a system the system should be designed to be helpful by providing the user with information regarding the error he/she has committed.

This application must be able to produce output at different modules for different Inputs.

3.4. Performance Requirements:

Performance is measured in terms of the output provided by the application.
Requirement specification plays an important part in the analysis of a system. Only when the requirement specifications are properly given, it is possible to design a system, which will fit into required environment. It rests largely in the part of the users of the existing system to give the requirement specifications because they are the people who finally use the system. This is because the requirements have to be known during the initial stages so that the system can be designed according to those requirements. It is very difficult to change the system once it has been designed and on the other hand designing a system. which does not cater to the requirements of the user, is of no use.

The requirement specification for any system can be broadly stated as given below:

- The system should be able to interface with the existing system.
- The system should be accurate
- The system should be better than the existing system

The existing system is completely dependent on the staff to perform all the duties,

3.5. PRELIMINARY INVESTIGATION

It offers many advantages which static websites cannot provide. In this kind of, users can interact with the site using various options and search criteria. Your online bank account web page is an example of a dynamic website. Search engines are also the examples of the same with the capabilities to search and present information as per your need. The information user enters in it can be stored in a database such as Oracle or DB2. It is developed using various web technologies such as Java, NET, PHP, and CGI Script. Dynamic website is interactive and fun for the surfer. On the contrary, static websites do not provide any options where users can key-in his views or questions. The user needs to go through the entire site to look for the

3.6. FEASIBILITY STUDY

Feasibility study is conducted once the problem is clearly understood. Feasibility study is a high-level capsule version of the entire system analysis and design process. The objective is to determine quickly at a minimum expense how to solve a problem. The purpose of feasibility is not to solve the problem but to determine if the problem is worth solving
The system has been tested for feasibility in the following points.

1. Technical Feasibility
2. Economic Feasibility
3. Operational Feasibility

3.7. PROJECT PLANNING

DESIGNING

With 40,000+ new websites every day, the internet is becoming virtually the only source to reach the world. Because of the demand, USM has attempted to get you started on owning and running your own personal or business website. Since there are many people willing to take your money for nothing, before we share some of the necessary steps used to create a professional web design and website, the following question must be addressed: What constitutes a "professional web design"? More often than not, a web designer or web design company claims to offer professional web design services for their clients. However, should they be promoting "web design services rather than "professional web design services"?

- Essential Information before Planning
- Card Sorting
- Content Inventories
- Paper and Sketch boards
- Site Map Diagrams
- The Architecture Is the Home, Not the Content Itself
- HTML Wireframes
- Plain Old Text
- Slides
- Jump chart
- Putting It All Together

Which to Choose?

There is no one right way to plan the architecture for a website. Depending on the size of the website, you might use all of these techniques. They're not opposed or mutually exclusive just different means to similar ends.

When picking your method of architecture planning, consider these things:

A few tips on architecture planning:

- Organize content according to user needs, not an organizational chart or how the client structures their company
- Give pages clear and succinct names.
- He sympathetic. Think of your typical users, called personas, and imagine them navigating the website. What would they be looking for?
- Consider creating auxiliary way-finding pages. These pages would lie beyond the main navigation of your website and structure various pages according to specific user needs.
- If you can't succinctly explain why a page would be useful to someone, omit it.
- Plan the architecture around the content. Don't write content to fit the architecture.
- When dealing with clients, especially client's at large companies with many departments, keeping egos in check can be tough. Keep everyone on point with constant reminders of the true goals of the website.
- Not everything has to be a page. Use your hierarchy of content as a guide. Some items might work better as an FAQ entry or as sidebar content. Make sure your architecture planning method does not blind you to this.

The Inevitable Revisions: Being Fleet of Foot

Clients change their minds. It's in their genes to be indecisive and difficult. If they knew what the heck they were doing, they wouldn't need us. Our job is to tum their mess into perfection. Despite the mess, budget and timeline, your work will be judged on its own ment. You either got it right or you didn't, and there's no passing the buck.

This Scylla and Charybdis are no reason to stop trying. What you need is a workflow that embraces change rather than resists it.

A Few Parting Practical Tips:

- Be specific about your wants with clients. Ask for digital text, Web-sized images, et
- Keep all deliverables in one place, and put them there as soon as you get them!
- Ask for written changes, preferably via email so that they're time-stamped
- Use Google's advanced site search to quickly learn about the current website's size and shape if your project is a redesign.
- Ask your client for access to old stats. Learning how people have been accessing content is important if you will be planning a new website.
- Avoid being too specific in the early stages. Work from general to specific, and don't get bogged down in details until they become important

3.8. PROJECT SCHEDULING

Software project scheduling is an activity that distributes estimated effort across the planned project duration by allocating the effort to specific software engineering tasks. However, the scheduled evolves over time. During early stages of project planning, s macroscopic scheduled is developed. This type of scheduled identifies all major process framework activities and the product functions to which they are applied. As the project gets under way, each entry on the macroscopic scheduled is refined into a detailed scheduled.

There are some basic principles guide software projects scheduling while implementing the project. Each of these principles is applied as the project scheduled evolves. They are:

- Compartmentalization.
- Interdependency,
- Time allocation.
- Effort validation.
- Defined responsibilities
- Defined outcomes
- Defied milestones

In project management, a schedule consists of a list of a project's terminal elements with intended start and finish dates. Terminal elements are the lowest element in a schedule, which is not further subdivided. Those items are often estimated in terms of resource requirements. budget and duration, linked by dependencies and scheduled

Overview

Before a project schedule can be created, a project manager should typically have a work breakdown structure (WBS), an effort estimates for cache task, and a resource list with availability for each resource. If these are not yet available, it may be possible to create something that looks like a schedule, but it will essentially be a work of fiction. They can be created using a consensus-driven estimation method like Wideband Delphi. The reason for this is that a schedule itself is an estimate: cache date in the schedule is estimated, and if those dates do not have the buy-in of the people who are going to do the work, the schedule will be inaccurate.

In many industries, such as engineering and construction, the development and maintenance of the project schedule is the responsibility of a full-time scheduler or team of schedulers, depending on the size of the project. And though the techniques of scheduling are well developed, they are inconsistently applied throughout industry. Standardization and promotion of scheduling best practices are being pursued by the Association for the Advancement of Cost Engineering (AACE), the Project Management Institute (PMI). In some large corporations, scheduling, as well as cost, estimating, and risk management are organized under the department of project controls.

4. SYSTEM DESIGN

4.1. SOFTWARE PROCESS MODELS

To solve actual problems in a software project setting, a software engineer or a team of engineers roust incorporate a development strategy that includes methods, tools and procedures. This strategy is referred to as a process model or a software engineering paradigm.

There are several such paradigms available. Which software paradigm is more suitable depends on lot of factors such as:

1. Nature of the Project
2. Type of Applications
3. Tools proposed to be used
4. Kind of controls and documentation required

4.2. CODE AND FIX LIFE CYCLE MODEL

As the name suggests, this model uses an adhoc approach for the software designing A very casual study is followed by cosling. The issues regarding specification or design are never addressed. Instead, the developers simply build a product that is re-build again and again until the customer is satisfied.

Limitations of code-and-fix cycle model

This approach may work well for small systems but is very unsatisfactory for larger systems. As the code size increases, the understandability and maintainability of the system decreases The Waterfall Model Linear Sequential Life Cycle Model. The simplest, oldest and most widely used process model for software designing is the waterfall model. It was proposed by Royce in 1970. The essence of this software paradigm is that the process of software designing consists of linear set of distinct phases These phases are

Stage-1: Feasibility Study

The goal of feasibility study is to evaluate alternative systems and to purpose the most feasible and desirable system for designing. Five types of feasibility are addressed in this
study.

1. Technical feasibility
2. Economic Feasibility
3. Motivational Feasibility
4. Schedule Feasibility
5. Operational Feasibility

Stage-2: Requirement Analysis and Specification

The goal of this phase is to understand the exact requirements of the customers and in document them properly. This activity is usually executed together with the customers, as the goal is to document all functions, performance and interfacing requirements for the **software designing and management**. The requirements describe "what" of a system. This phase

produces a large document containing a description of what the system will do without describing how it will be done. This document is known as **software requirement specification (SRS) document.**

Stage-3: Design

The goal of this phase is to transform the requirement specification produced in the requirement analysis phase into a structure that is suitable for implementation, in some programming language, here, overall software architecture is defined, and the product design and detailed design work is performed. This work is documented and is known as **software design description (SDD document).**

Stage-4: Coding and Unit Testing

The information contained in SDD is sufficient to begin the coding Phase. The coding Phase of **software designing** involves translation of design specification imo a machine-readable form. If design is performed in a detailed manner, code generation can be accomplished easily. This phase is also known as the implementation phase. Here, each component of the design is implemented as a program module, and cache of these program modules is unit tested. The purpose of unit testing is to determine the correct working of individual modules.

Stage-5: Integration and System Testing:

During this phase the different program modules are integrated in a planned way and then tested as a completed system to ensure that the designed system functions according to its requirements as specified in the SRS document. After testing, the software is delivered to the customer

Stage-6: Software Maintenance

This is the last phase of software designing which includes a broad set of activities such as error correction, enhancement of capabilities, deletion of obsolete capabilities and optimization.

Advantages

1.Simple and easy to use
2. Easily manageable
3. The phase of the model are processed and completed one at a time
4. Works very well for smaller software projects.

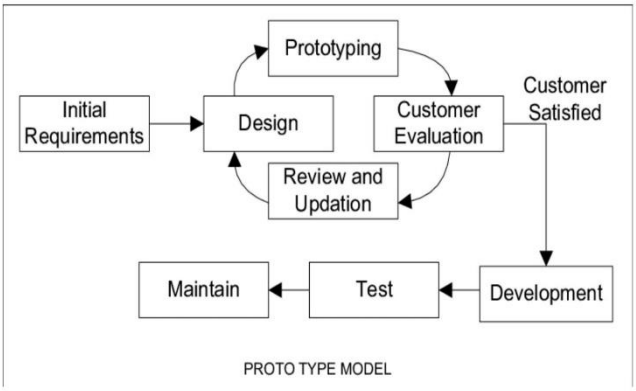
Disadvantages

1. It is often difficult for the customer to state all the requirements explicitly.
2. Real projects rarely follow the sequential flow that the software model proposes.
3. The customer must have patience, as the product is delivered very late in this software process.
4. The model is not suitable for long time software projects.

4.3. THE PROTOTYPING MODEL

Prototyping is a technique that provides a reduced functionality or limited performance version of the eventual software to be delivered to the user in the early stages of the software development process. If used judiciously, this approach helps to solidify user requirements earlier, thereby making the waterfall approach more effective.

What is done is that before proceeding with design and coding, a throw away prototype is built to give user a feel of the system. The development of the software prototype also involves design and coding, but this is not done in a formal manner. The user interacts with the prototype as he would do would therefore be in a better position to specify his requirements in a more detailed manner. The iterations occur to refine the prototype to satisfy the needs of the user, while at the same time enabling the developer to better understand what needs to be done.



PROTO TYPE MODEL

Disadvantages

1. In prototyping, as the prototype has to be discarded, so might argue that the cost involved is higher.
2. At times, while designing a prototype, the approach adopted is "quick and dirty with the focus on quick development rather than quality.
3. The developer often makes implementation compromises in order to get a prototype working quickly.

Maintenance:

The key to reducing need for maintenance, while working, if possible, to do essential tasks.
1. More accurately defining user requirement during system development.
2. Assembling better systems documentation.
3. Using more effective methods for designing, processing, login and communicating information with project team members.
4. Making better use of existing tools and techniques.
5. Managing system engineering process effectively.

Output Design.

One of the most important factors of an information system for the user is the output the system produces. Without the quality of the output, the entire system may appear unnecessary that will make us avoid using it possibly causing it to fail. Designing the output should process the in an organized well throughout the manner. The right output must be developed while ensuring that each output element is designed so that people will find the system easy to use effectively.

The term output applying to information produced by an information system whether printed or displayed while designing the output we should identify the specific output that is needed to information requirements select a method to present the formation and create a document report or other formats that contains produced by the system.

Types of output:

Whether the output is formatted report or a simple listing of the contents of a file, a computer process will produce the output

- A Document
- A Message
- Retrieval from a data store
- Transmission from a process or system activity
- Directly from an output sources

Layout Design

It is an arrangement of items on the output medium. The layouts are building a mockup of the actual reports or document, as it will appear after the system is in operation. The output layout has been designated to cover information. The outputs are presented in the appendix.

Input design and control

Input specifications describe the manner in which data enter the system for processing. Input design features will ensure the reliability of the systems and produce results from accurate data, or thus can be result in the production of erroneous information. The input design also determines whenever the user can interact efficiently with this system

Objectives of input design

Input design consists of developing specifications and procedures for data preparation, the steps necessary to put transaction data into a usable from for processing and data entry, the activity of data into the computer processing. The five objectives of input design are:

- Controlling the amount of input
- Avoiding delay
- Avoiding error in data
- Avoiding extra steps
- Keeping the process simple

Controlling the amount of input:

Data preparation and data entry operation depend on people, because labour costs are high, the cost of preparing and entering data is also high. Reducing data requirement expense. By reducing input requirement, the speed of entire process from data capturing to processing to provide results to users.

Avoiding delay:

The processing delay resulung from data preparation or data entry operations is called bottlenecks. Avoiding bottlenecks should be one objective of input.

Avoiding errors:

Through input validation we control the errors in the input data.

Avoiding extra steps:

The designer should avoid the input design that cause extra steps in processing saving or adding a single step in large number of transactions saves a lot of processing time or takes more time to process.

Keeping process simple:

If controls are more people may feel difficult in using the systems. The best-designed system fits the people who use it in a way that is comfortable for them.

4.4. DATA FLOW DIAGRAM

A data flow diagram is graphical tool used to describe and analyze movement of data through a system. These are the central tool and the basis from which the other components are developed. The transformation of data from input to output, through processed, may be described logically and independently of physical components associated with the system.

These are known as the logical data flow diagrams. The physical data flow diagrams show the actual implements and movement of data between people, departments and workstations. A full description of a system actually consists of a set of data flow diagrams. Using two familiar notations Yourdon, Gane and Sarson notation develops the data flow diagrams. Each component in a DFD is labelled with a descriptive name Process is further identified with a number that will be used for identification purpose The development of DFD's is done in several levels. Each process in lower-level diagrams can be broken down into a more detailed DFD in the next level. The lop-level diagram is often called context diagram. It consists a single process bit, which plays vital role in studying the current system. The process in the context level diagram is exploded into other process at the first level DFD.

The idea behind the explosion of a process into more process is that understanding at one level of detail is exploded into greater detail at the next level. This is done until further explosion is necessary and an adequate amount of detail is described for analyst to understand the process. Larry Constantine first developed the DFD as a way of expressing system requirements in a graphical from, this lead to the modular design. A DFD is also known as a "bubble Chart" has the purpose of clarifying system requirements and identifying major transformations that will become programs in system design. So it is the starting point of the design to the lowest level of detail. A DFD consists of a series of bubbles joined by data flows in the system.

CONSTRUCTING A DFD:

Several rules of thumb are used in drawing DFD's:

1. Process should be named and numbered for an easy reference. Each name should be representative of the process.

2. The direction of flow is from top to bottom and from left to right. Data Traditionally flow from source to the destination although they may flow back to the source. One way to indicate this is to draw long flow line back to a source: An alternative way is to repeat the source symbol as a destination. Since it is used more than once in the DFD it is marked with a short diagonal.

3. When a process is exploded into lower-level details, they are numbered.

4. The names of data stores and destinations are written in capital letters. Process and dataflow names have the first letter of each work capitalized.

A DFD typically shows the minimum contents of data store. Each data store should contain all the data elements that flow in and out.

Questionnaires should contain all the data elements that flow in and out. Missing interfaces redundancies and like is then accounted for often through interviews.

SAILENT FEATURES OF DFD's

1. The DFD shows flow of data, not of control loops and decision are controlled considerations do not appear on a DFD.
2. The DFD does not indicate the time factor involved in any process whether the data flows take place daily, weekly, monthly or yearly.
3. The sequence of events is not brought out on the DFD.

TYPES OF DATA FLOW DIAGRAMS

1. Current Physical
2. Current Logical
3. New Logical
4. New Physical

CURRENT PHYSICAL:

In Current Physical DFD process label include the name of people of their positions or the names of computer systems that might provide some of the overall system-processing label includes an identification of the technology used to process the data. Similarly, data flows and data stores are often labels with the names of the actual physical media on which data are stored such as file folders, computer files, business forms or computer tapes.

CURRENT LOGICAL:

The physical aspects at the system are removed as mush as possible so that the current system is reduced to its essence to the data and the processors that transform them regardless of actual physical form.

NEW LOGICAL:

This is exactly like a current logical model if the user were completely happy with he user were completely happy with the functionality of the current system but had problems with how it was implemented typically through the new logical model will differ from current logical model while having additional functions, absolute function removal and inefficient flows recognized.

NEW PHYSICAL:

The new physical represents only the physical implementation of the new systern.

RULES GOVERNING THE DED'S

PROCESS

1. No process can have only outputs.
2. No process can have only inputs. If an object has only inputs than it must be a sink.
3. A process has a verb phrase label.

DATA STORE

1. Data cannot move directly from one data store to another data store, a process must move data,
2. Data cannot move directly from an outside source to a data store, a process, which receives, must move data from the source and place the data into data store
3. A data store has a noun phrase label.

SOURCE OR SINK: The origin and/or destination of data

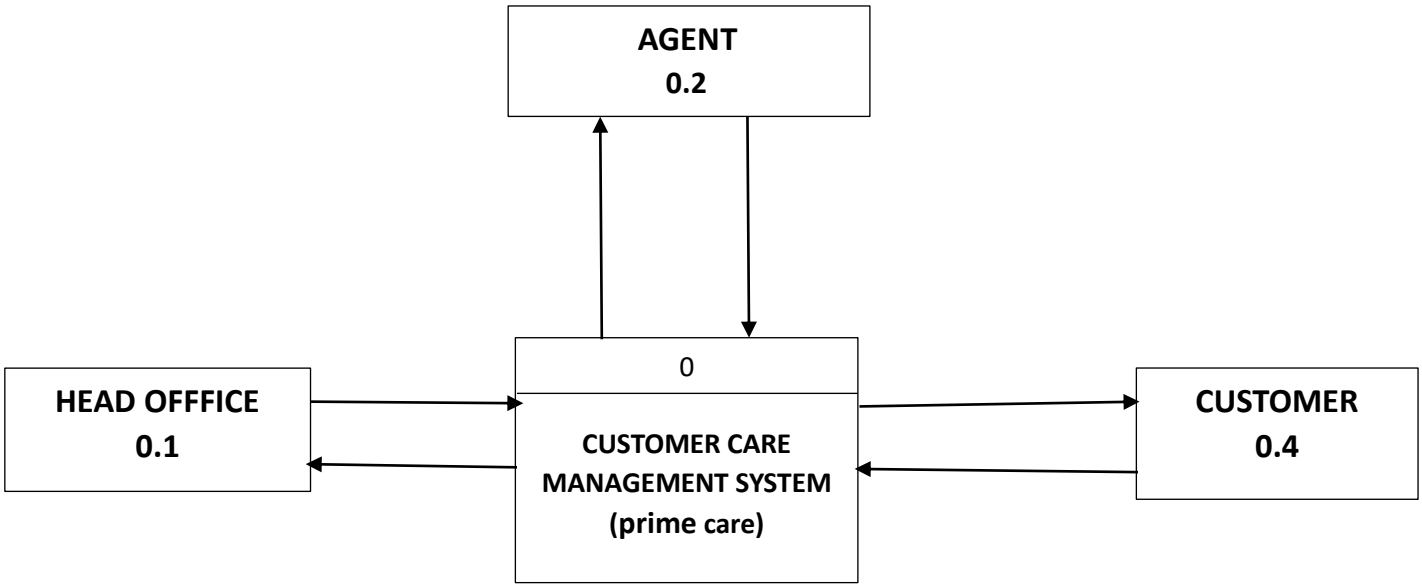
1. Data cannot move direly from a source to sink it must be moved by a process
2. A source and/or sink has a noun phrase land

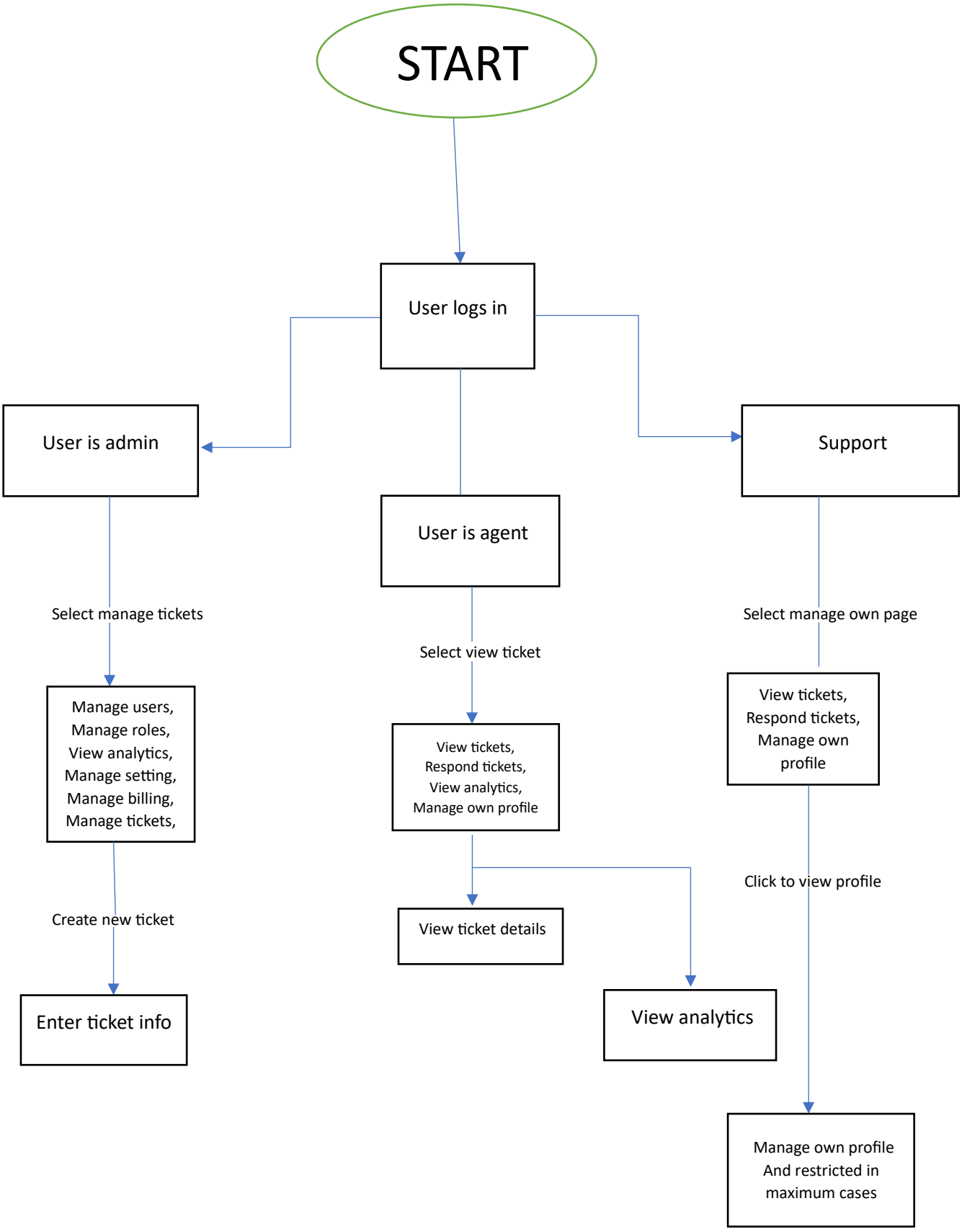
DATA FLOW

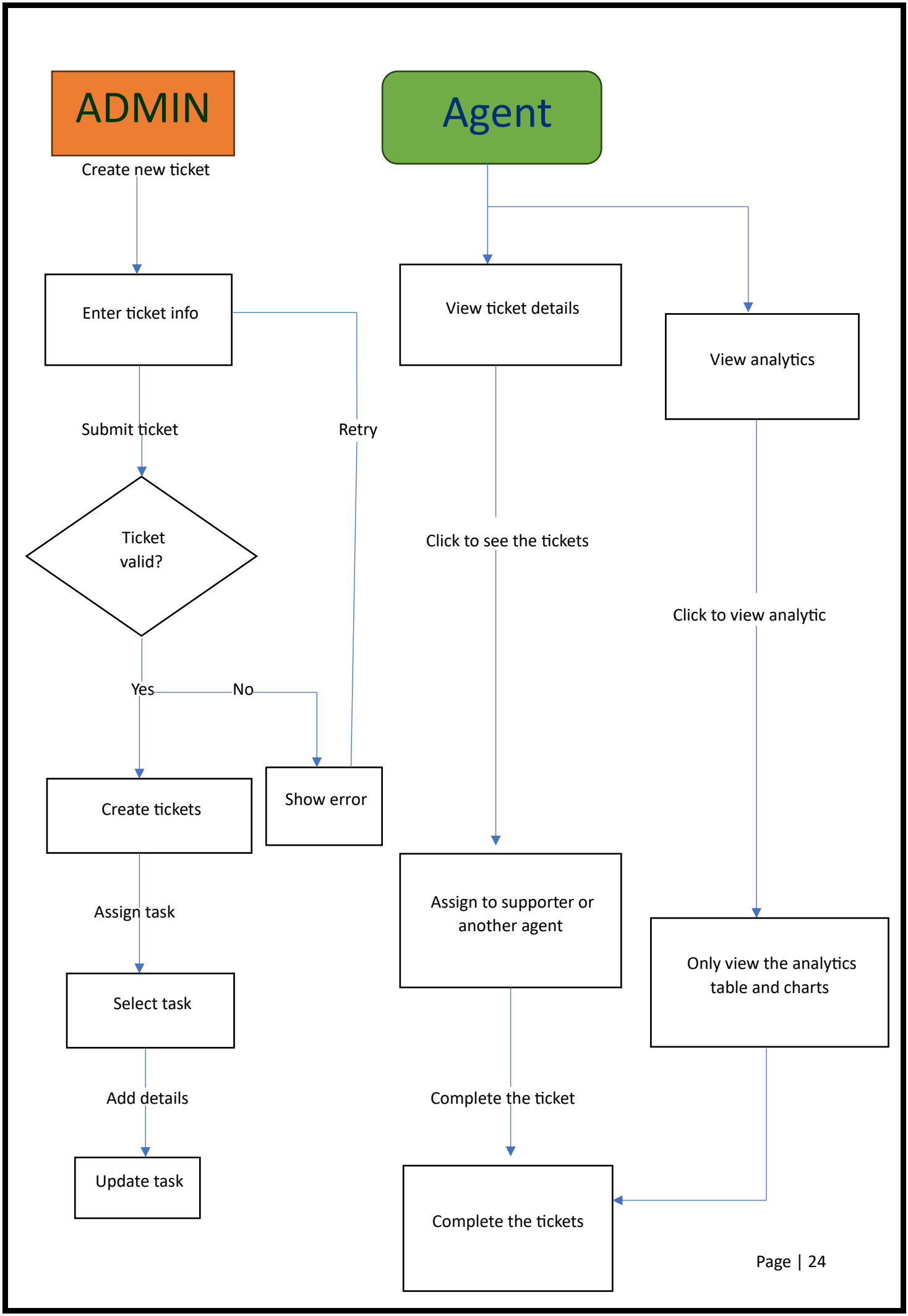
1. A Data Flow has only one direction of flow between symbol. It may flow in both directions between a process and a data store to show a read before an update. The later is usually indicated however by two separate arrows since these happen at different type.
2. A join in DFD means that exactly the same data comes from any of two or more different processes data store or sink to a common location.

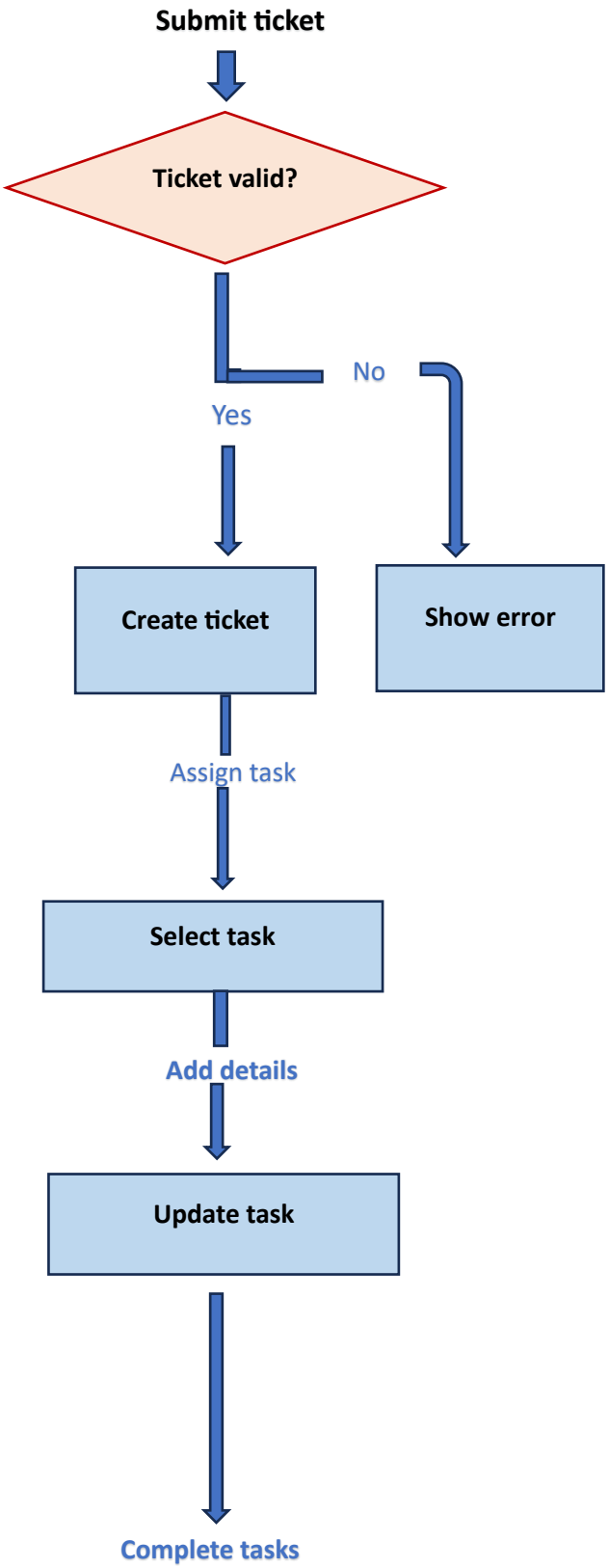
3. A data flow cannot go directly back to the same process it leads. There must be atleast one other process that handles the data flow produce some other data flow returns the original data into the beginning process.
4. A Data flow to a data store means update (delete or change).
5. A data Flow from a data store means retrieve or use.
A data flow has a noun phrase label more than one data flow noun phrase can appear on a single arrow as long as all of the flows on the same arrow move together as one package.

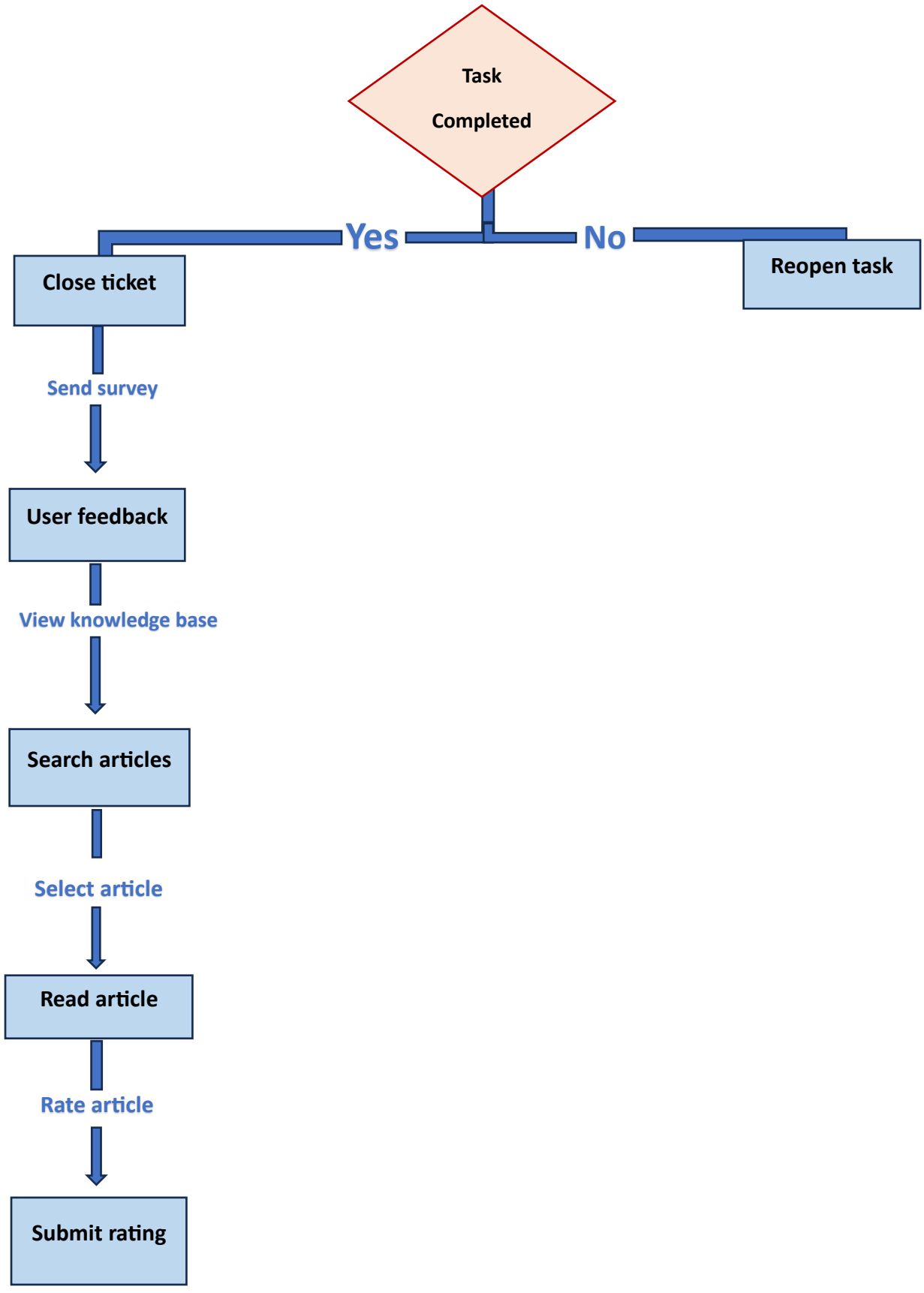
CONTEXT LEVEL DFD: -

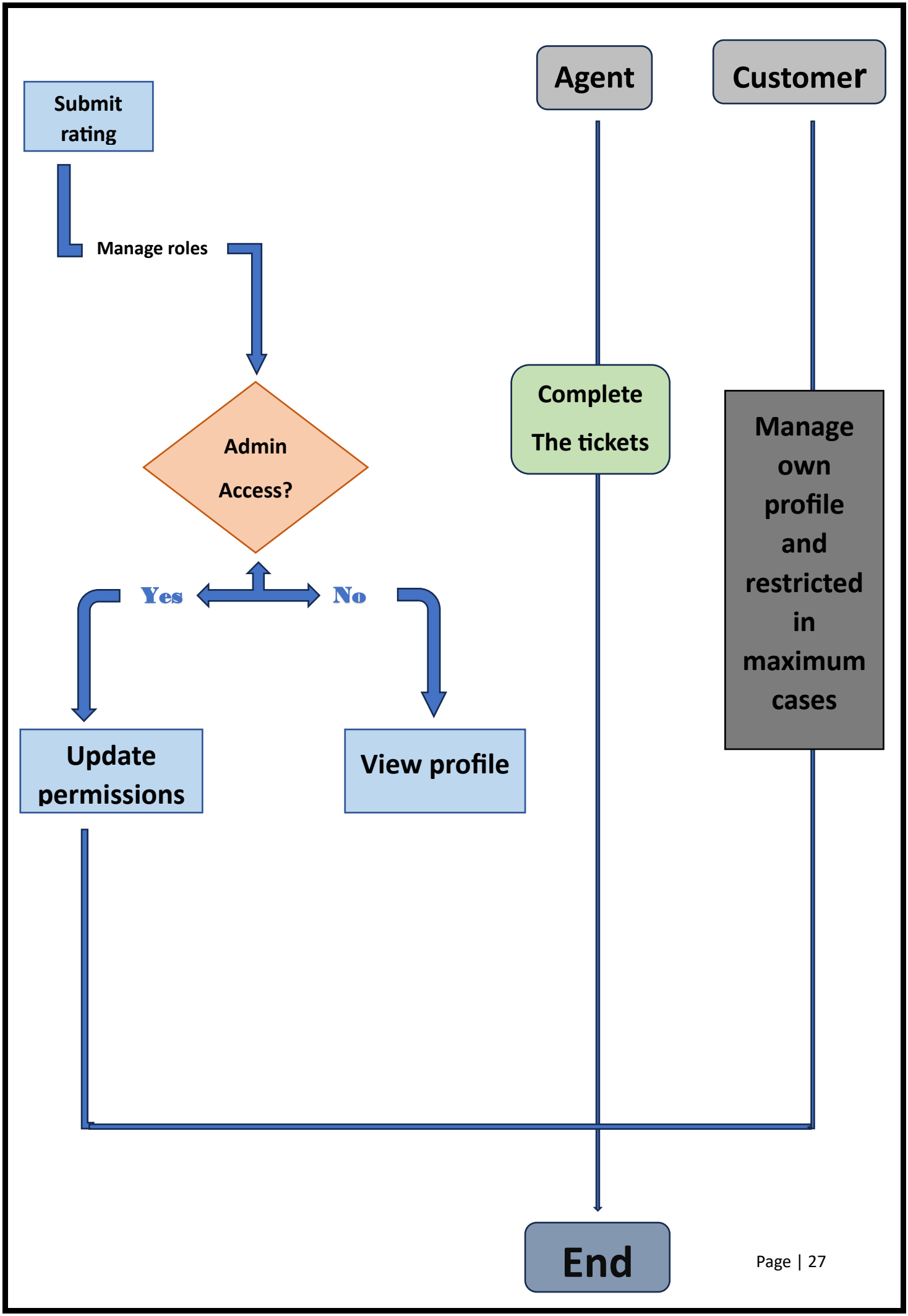



START
User logs in
User is admin
User is agent
Support
Select manage tickets
Select view ticket
Select manage own page
Manage users,
Manage roles,
View analytics,
Manage setting,
Manage billing,
Manage tickets,
View tickets,
Respond tickets,
View analytics,
Manage own profile
View tickets,
Respond tickets,
Manage own profile
Create new ticket
View ticket details
Click to view profile
Enter ticket info
View analytics
Manage own profile
And restricted in
maximum cases

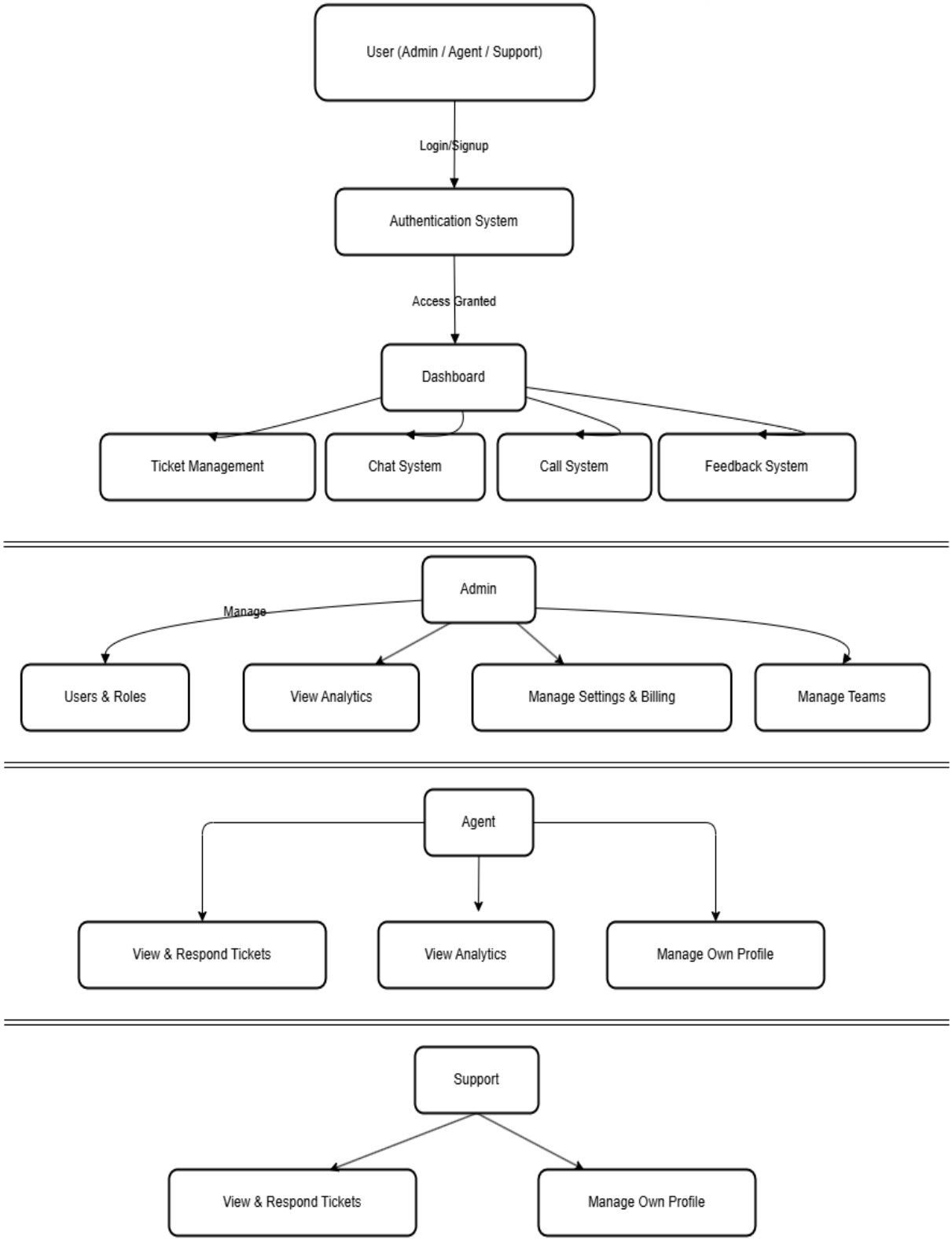

ADMIN
Agent
Create new ticket
Enter ticket info
Submit ticket
Retry
Ticket valid?
Yes
No
Create tickets
Show error
Assign task
Select task
Add details
Update task
View ticket details
View analytics
Click to see the tickets
Click to view analytic
Assign to supporter or another agent
Only view the analytics table and charts
Complete the ticket
Complete the tickets


Submit ticket
Ticket valid?
No
Yes
Create ticket
Show error
Assign task
Select task
Add details
Update task
Complete tasks


Task
Completed
Yes
No
Close ticket
Reopen task
Send survey
User feedback
View knowledge base
Search articles
Select article
Read article
Rate article
Submit rating

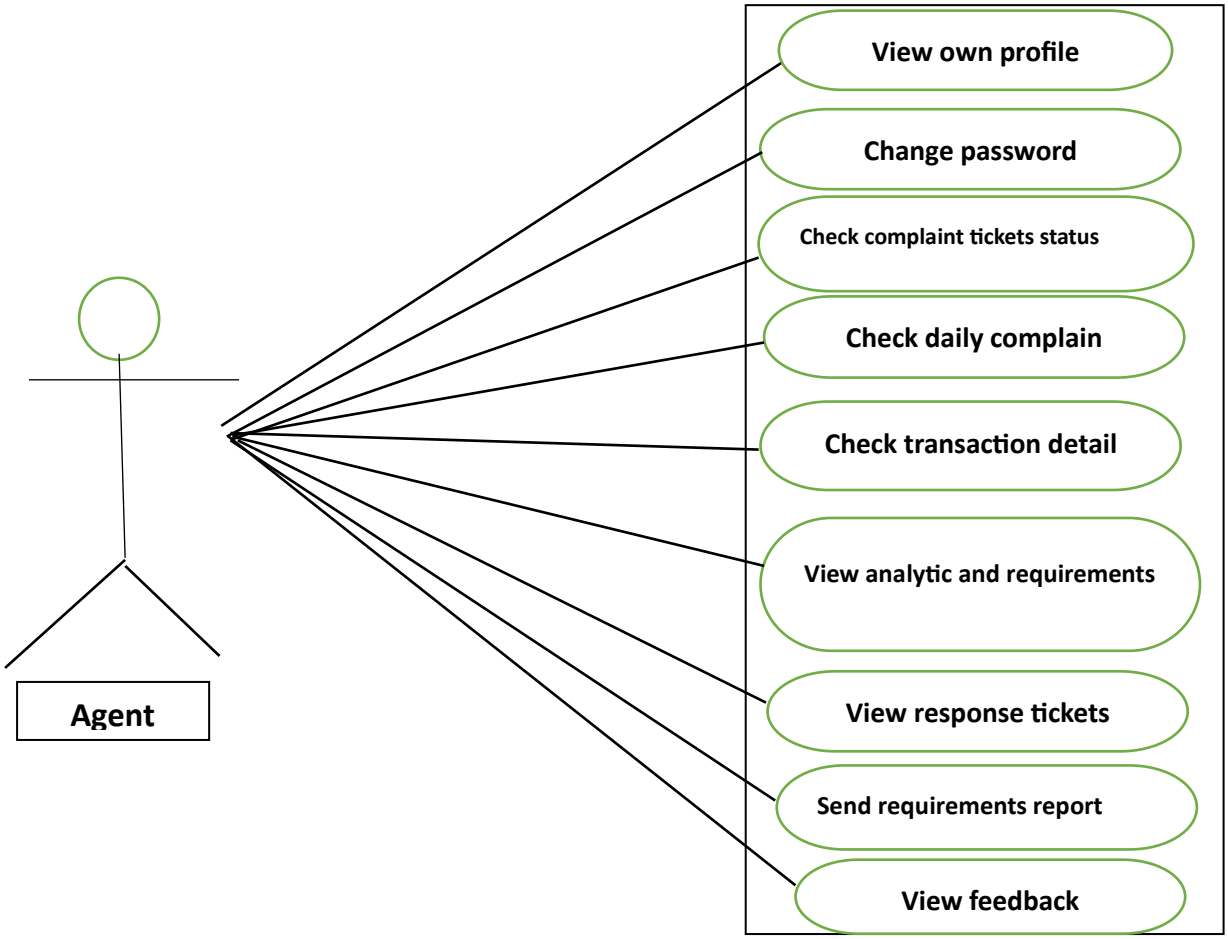

Agent
Customer
Submit rating
Manage roles
Admin Access?
Yes
No
Update permissions
View profile
Complete The tickets
Manage own profile and restricted in maximum cases
End

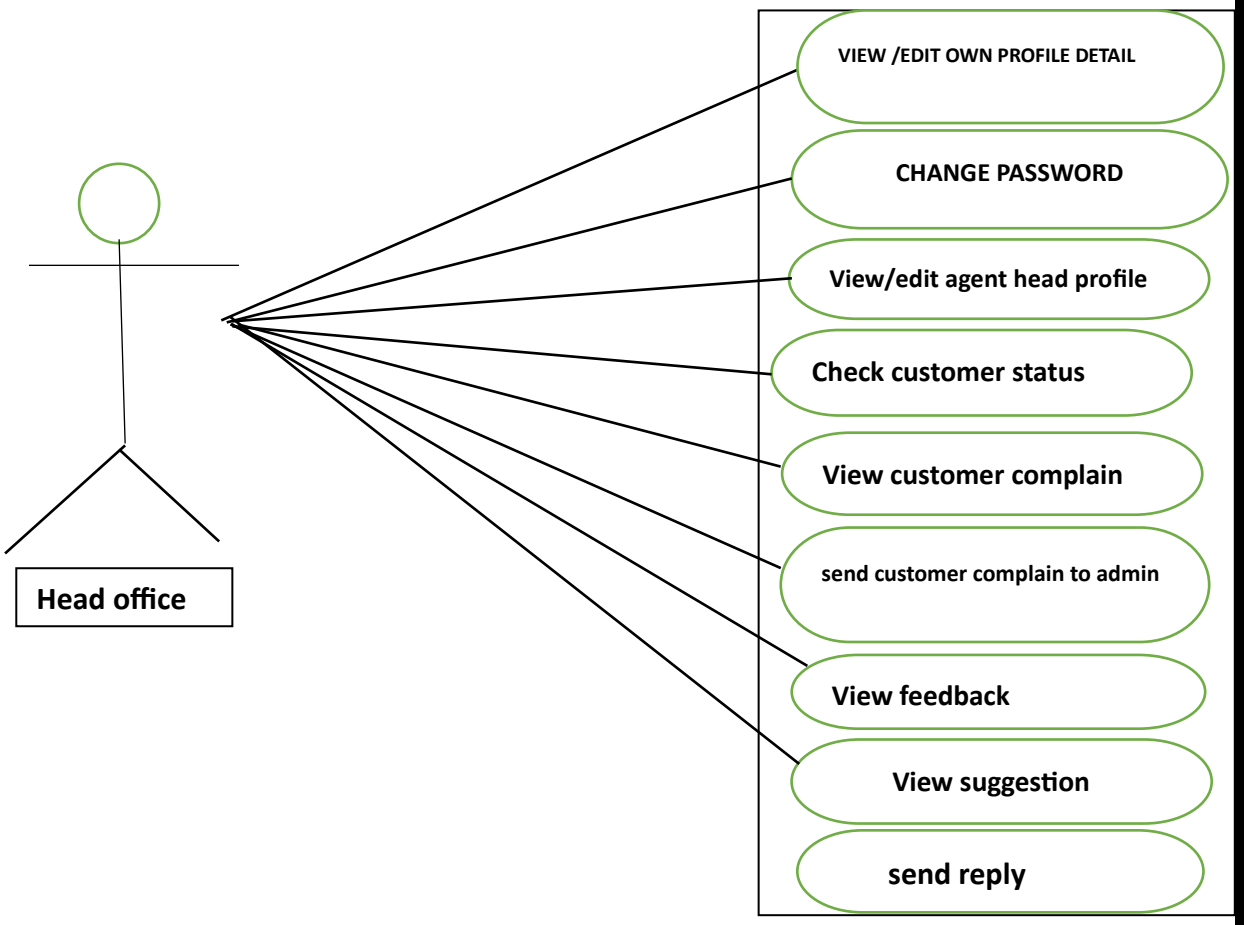
4.5. ER DIAGRAM

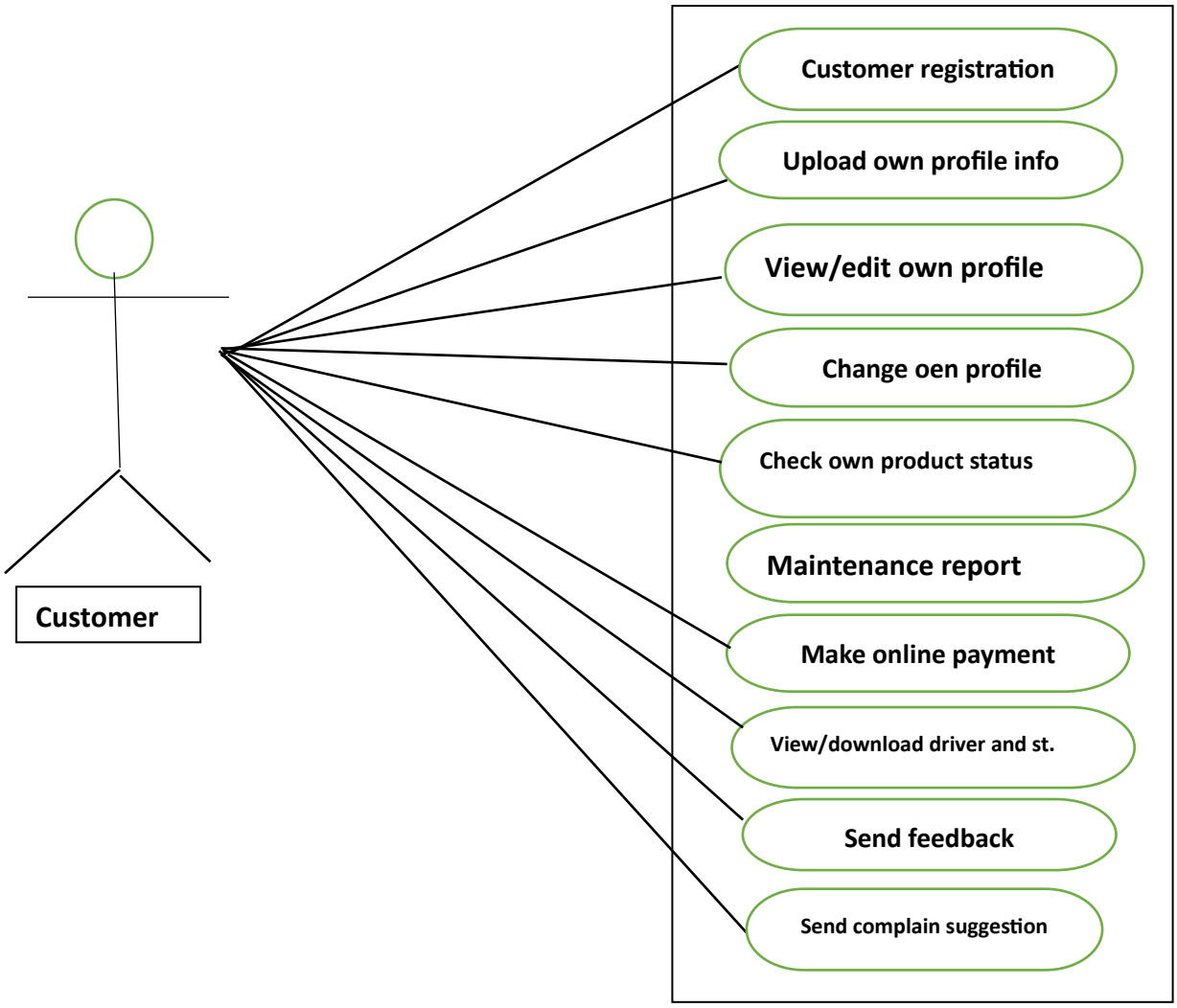


4.6. USE CASE DIAGRAM

AGENT

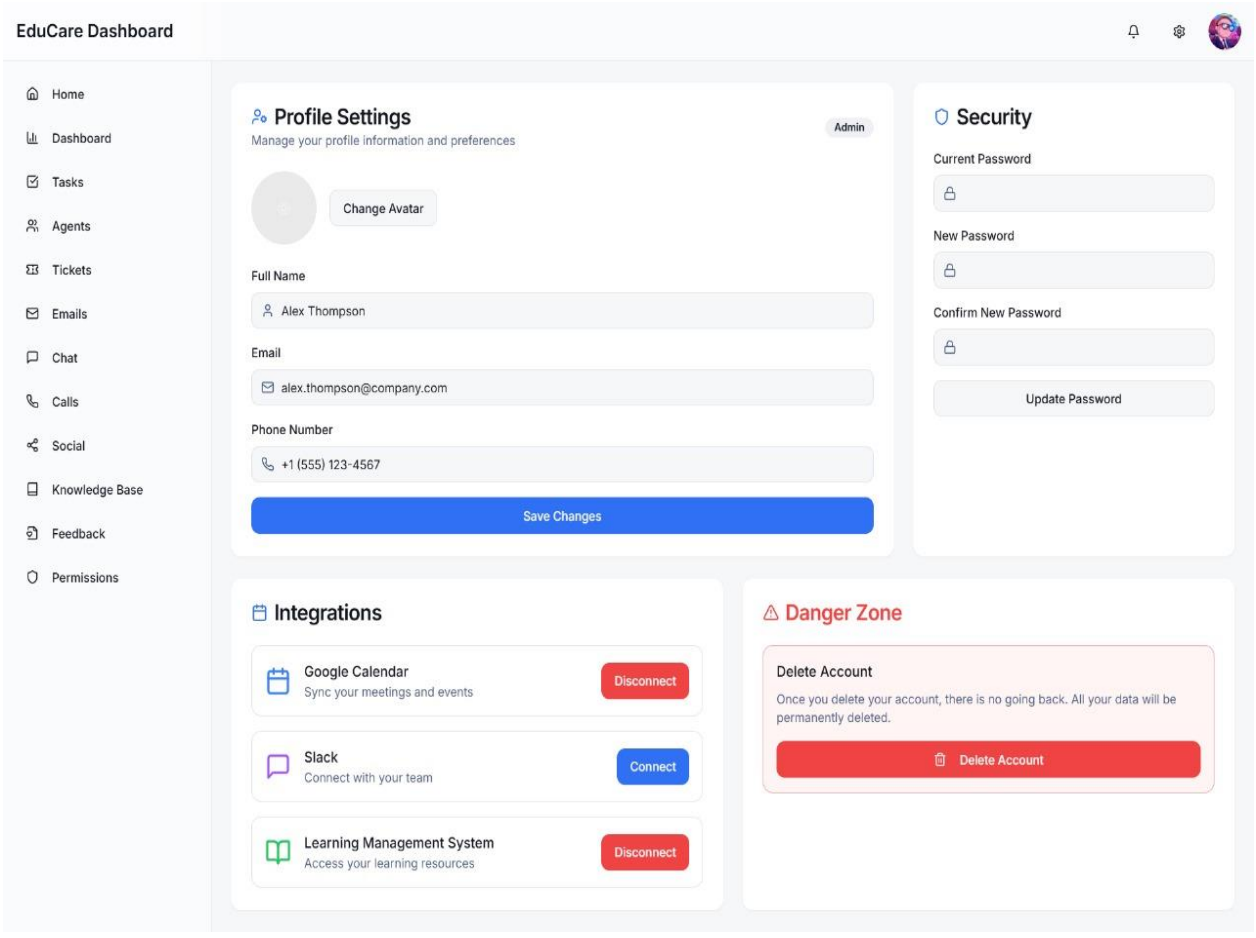



VIEW /EDIT OWN PROFILE DETAIL
CHANGE PASSWORD
View/edit agent head profile
Check customer status
View customer complain
send customer complain to admin
View feedback
View suggestion
send reply
Head office

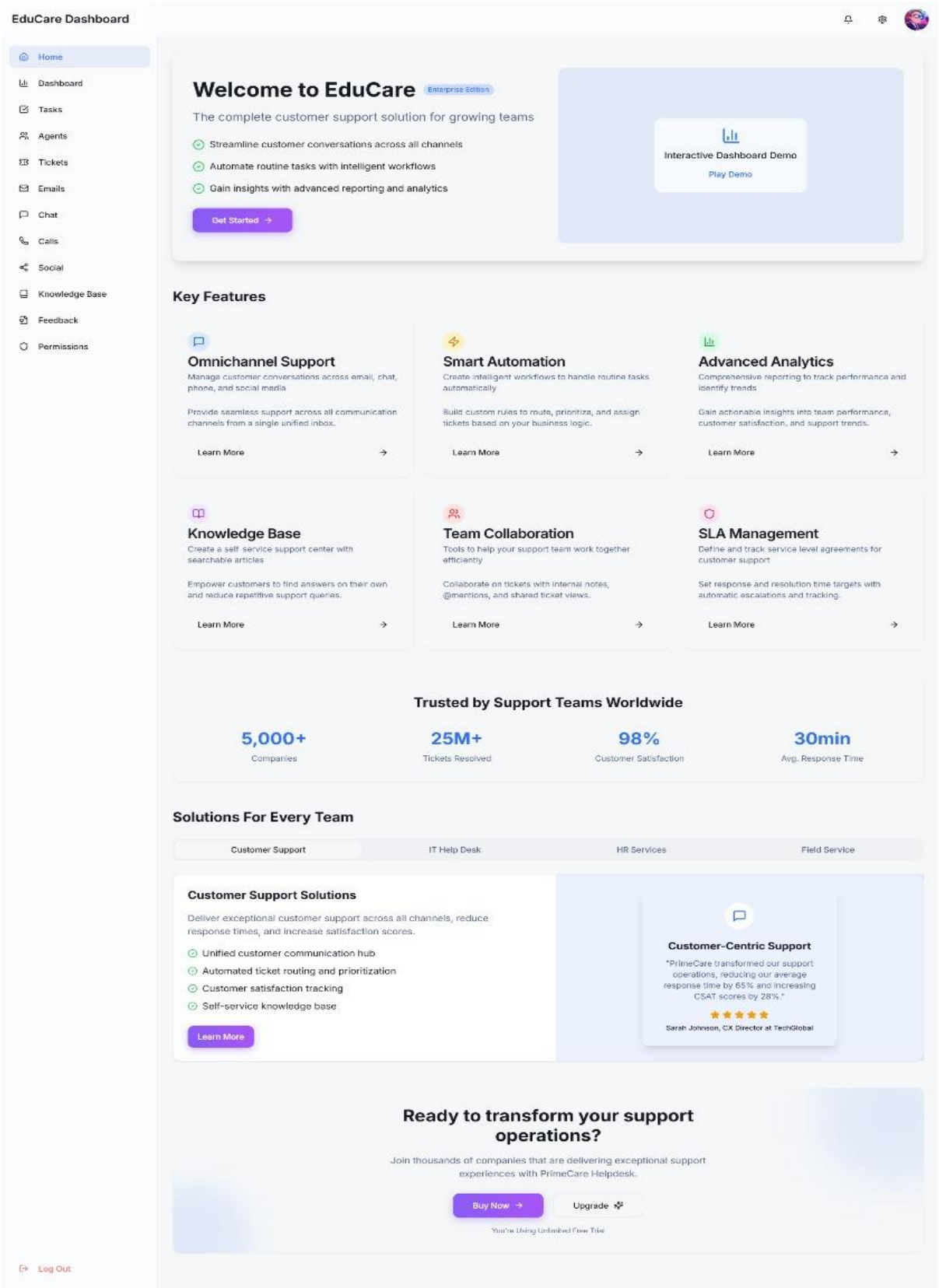

Customer registration
Upload own profile info
View/edit own profile
Change oen profile
Check own product status
Maintenance report
Make online payment
View/download driver and st.
Send feedback
Send complain suggestion
Customer

5. SNAPSHOT

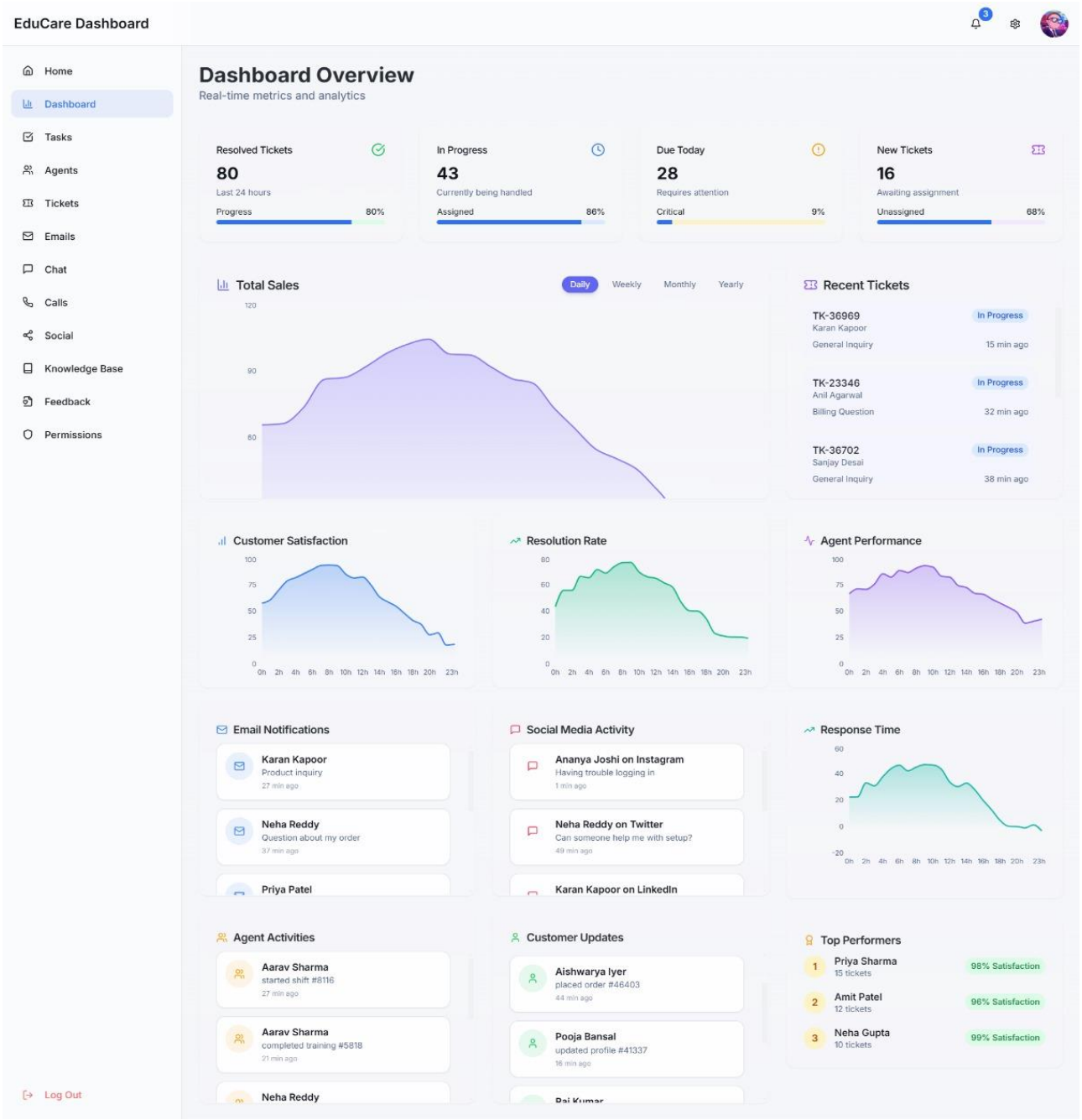
PROFILE PAGE



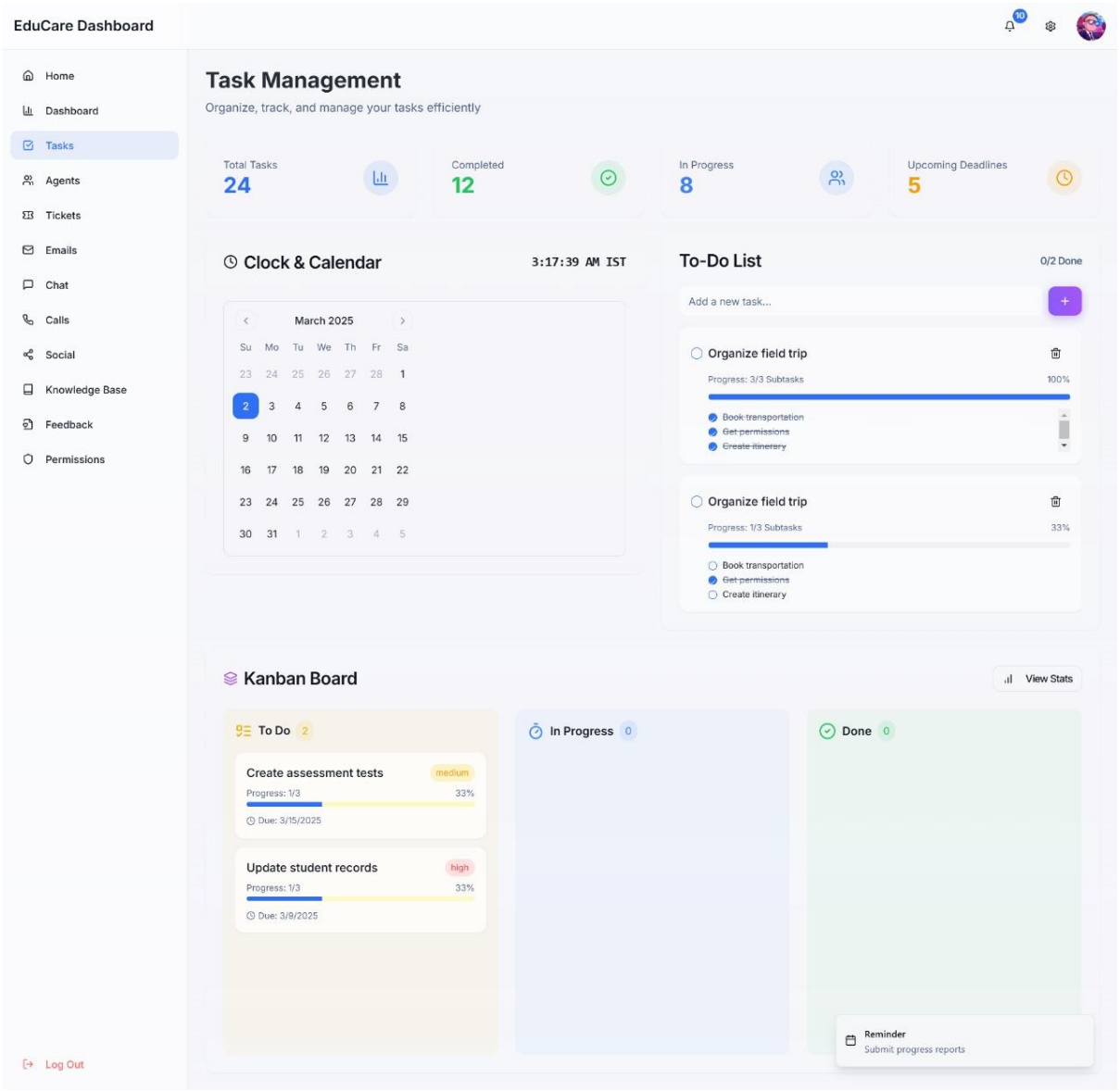
HOME PAGE


EduCare Dashboard
Home
Dashboard
Tasks
Agents
Tickets
Emails
Chat
Calls
Social
Knowledge Base
Feedback
Permissions
Log Out
Welcome to EduCare
Enterprise Edition
The complete customer support solution for growing teams
Streamline customer conversations across all channels
Automate routine tasks with intelligent workflows
Gain insights with advanced reporting and analytics
Get Started
Interactive Dashboard Demo
Play Demo
Key Features
Omnichannel Support
Manage customer conversations across email, chat, phone, and social media
Provide seamless support across all communication channels from a single unified inbox.
Learn More
Smart Automation
Create intelligent workflows to handle routine tasks automatically
Build custom rules to route, prioritize, and assign tickets based on your business logic.
Learn More
Advanced Analytics
Comprehensive reporting to track performance and identify trends
Gain actionable insights into team performance, customer satisfaction, and support trends.
Learn More
Knowledge Base
Create a self-service support center with searchable articles
Empower customers to find answers on their own and reduce repetitive support queries.
Learn More
Team Collaboration
Tools to help your support team work together efficiently
Collaborate on tickets with internal notes, @mentions, and shared ticket views.
Learn More
SLA Management
Define and track service level agreements for customer support
Set response and resolution time targets with automatic escalations and tracking.
Learn More
Trusted by Support Teams Worldwide
5,000+
Companies
25M+
Tickets Resolved
98%
Customer Satisfaction
30min
Avg. Response Time
Solutions For Every Team
Customer Support
IT Help Desk
HR Services
Field Service
Customer Support Solutions
Deliver exceptional customer support across all channels, reduce response times, and increase satisfaction scores.
Unified customer communication hub
Automated ticket routing and prioritization
Customer satisfaction tracking
Self-service knowledge base
Learn More
Customer-Centric Support
"PrimeCare transformed our support operations, reducing our average response time by 65% and increasing CSAT scores by 28%."
Sarah Johnson, CX Director at TechGlobal
Ready to transform your support operations?
Join thousands of companies that are delivering exceptional support experiences with PrimeCare Helpdesk.
Buy Now
Upgrade

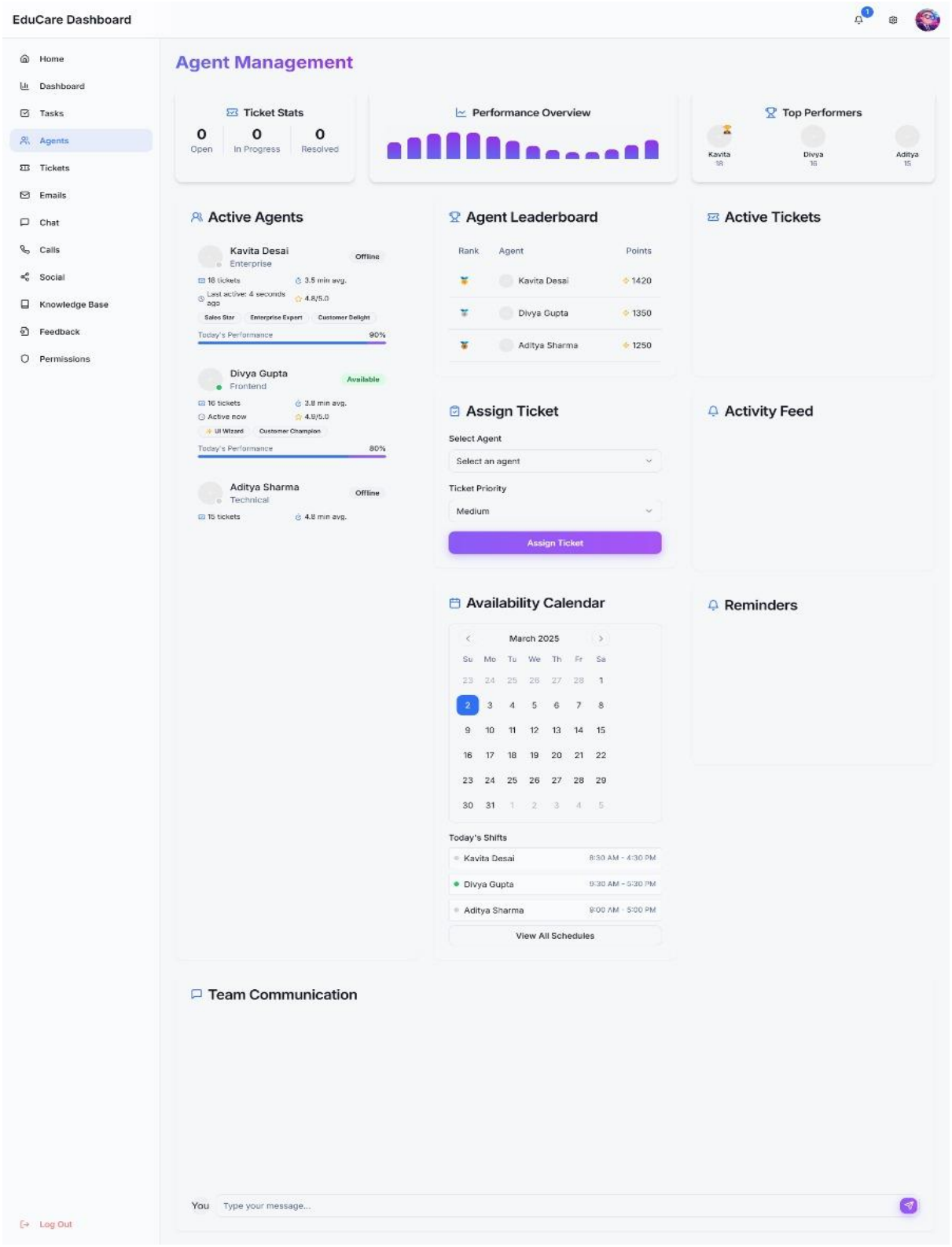
DASHBOARD PAGE



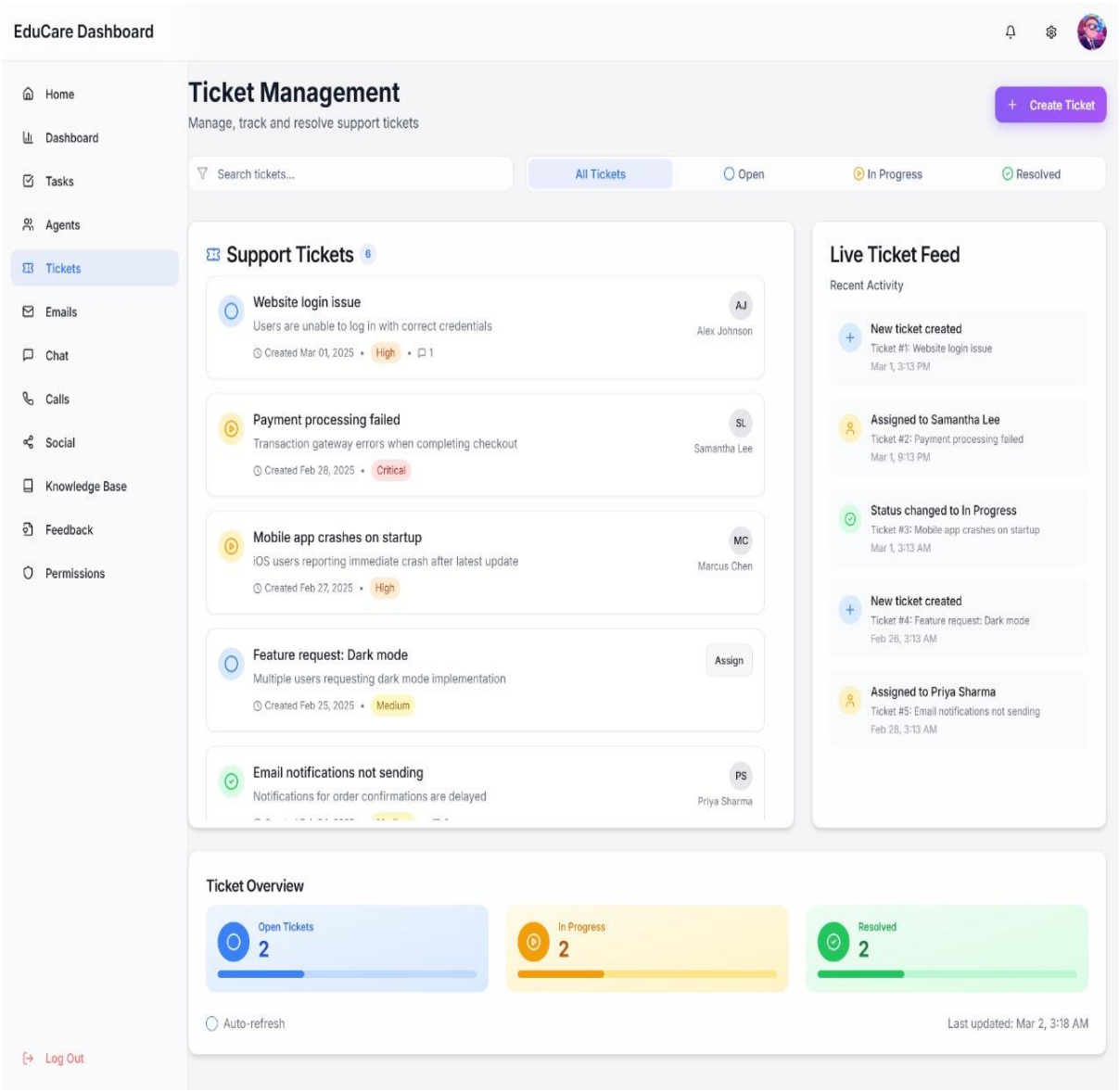
TASKS PAGE


EduCare Dashboard
Home
Dashboard
Tasks
Agents
Tickets
Emails
Chat
Calls
Social
Knowledge Base
Feedback
Permissions
Log Out
Task Management
Organize, track, and manage your tasks efficiently
Total Tasks
24
Completed
12
In Progress
8
Upcoming Deadlines
5
Clock & Calendar
3:17:39 AM IST
March 2025
To-Do List
0/2 Done
Add a new task...
Organize field trip
Progress: 3/3 Subtasks
100%
Book transportation
Get permissions
Create itinerary
Organize field trip
Progress: 1/3 Subtasks
33%
Book transportation
Get permissions
Create itinerary
Kanban Board
View Stats
To Do
2
In Progress
0
Done
0
Create assessment tests
medium
Progress: 1/3
33%
Due: 3/15/2025
Update student records
high
Progress: 1/3
33%
Due: 3/9/2025
Reminder
Submit progress reports

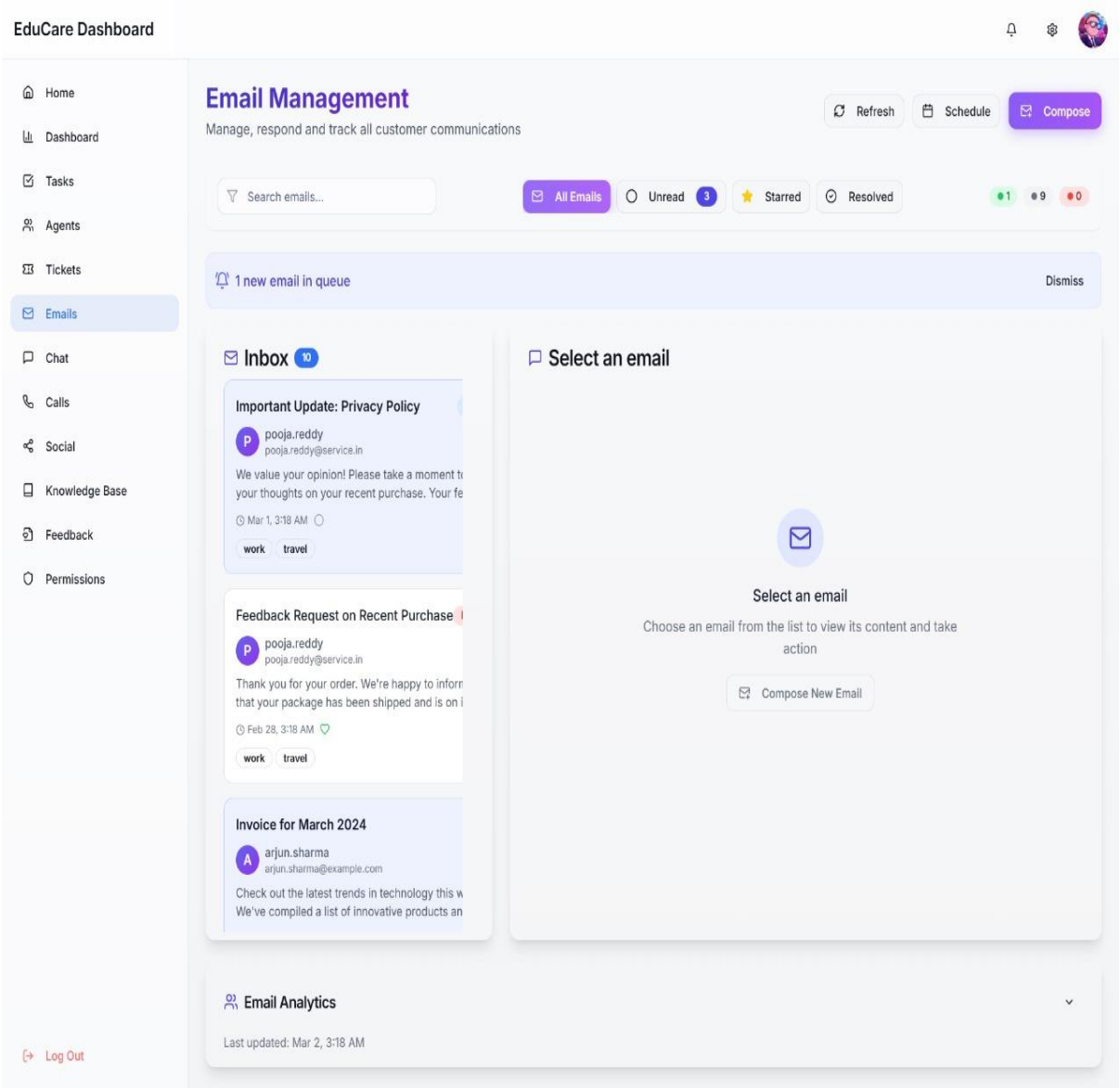
AGENTS PAGE


EduCare Dashboard
Home
Dashboard
Tasks
Agents
Tickets
Emails
Chat
Calls
Social
Knowledge Base
Feedback
Permissions
Log Out
Agent Management
Ticket Stats
0 Open
0 In Progress
0 Resolved
Performance Overview
Top Performers
Kavita
Divya
Aditya
Active Agents
Kavita Desai
Enterprise
Offline
Divya Gupta
Frontend
Available
Aditya Sharma
Technical
Offline
Agent Leaderboard
Rank
Agent
Points
Kavita Desai 1420
Divya Gupta 1350
Aditya Sharma 1250
Active Tickets
Assign Ticket
Select Agent
Select an agent
Ticket Priority
Medium
Assign Ticket
Activity Feed
Availability Calendar
March 2025
Today's Shifts
Kavita Desai
Divya Gupta
Aditya Sharma
View All Schedules
Reminders
Team Communication
You
Type your message...

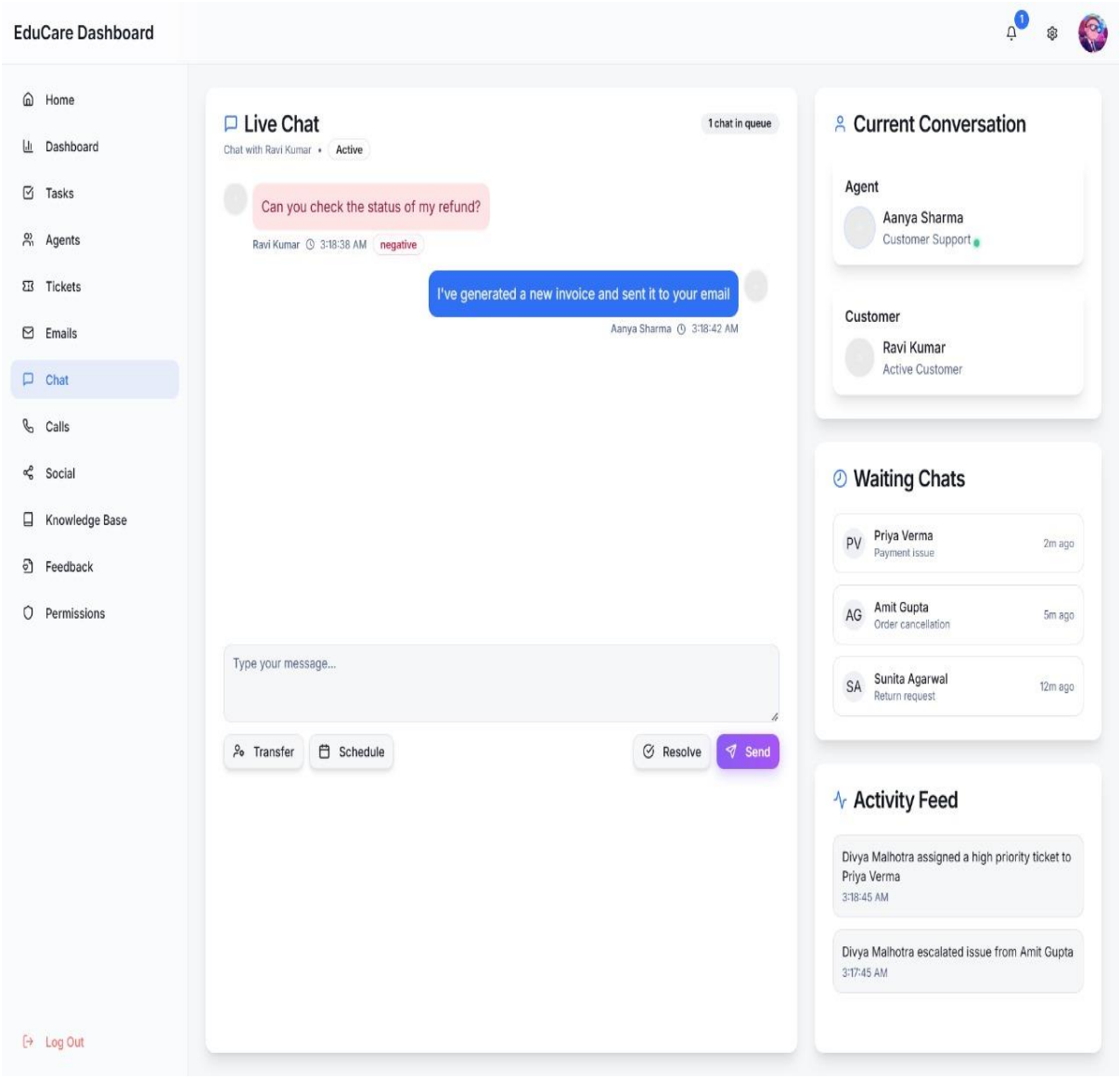
TICKETS PAGE



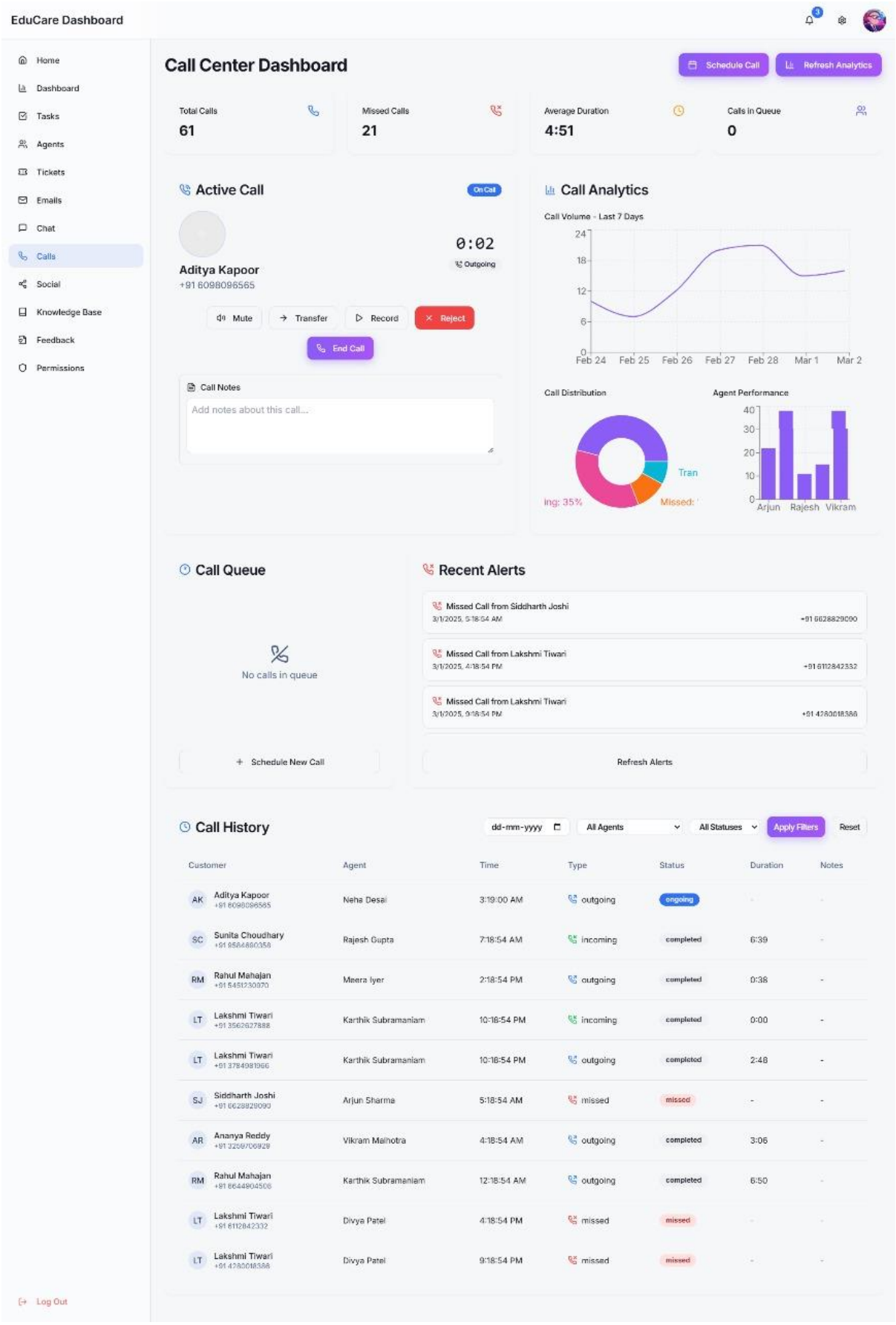
EMAILS PAGE


EduCare Dashboard
Home
Dashboard
Tasks
Agents
Tickets
Emails
Chat
Calls
Social
Knowledge Base
Feedback
Permissions
Log Out
Email Management
Manage, respond and track all customer communications
Refresh
Schedule
Compose
Search emails...
All Emails
Unread 3
Starred
Resolved
1 new email in queue
Dismiss
Inbox 10
Important Update: Privacy Policy
pooja.reddy
pooja.reddy@service.in
We value your opinion! Please take a moment t
your thoughts on your recent purchase. Your fe
Mar 1, 3:18 AM
work
travel
Feedback Request on Recent Purchase
pooja.reddy
pooja.reddy@service.in
Thank you for your order. We're happy to inforn
that your package has been shipped and is on i
Feb 28, 3:18 AM
work
travel
Invoice for March 2024
arjun.sharma
arjun.sharma@example.com
Check out the latest trends in technology this w
We've compiled a list of innovative products an
Select an email
Select an email
Choose an email from the list to view its content and take action
Compose New Email
Email Analytics
Last updated: Mar 2, 3:18 AM

CHAT PAGE


EduCare Dashboard
Home
Dashboard
Tasks
Agents
Tickets
Emails
Chat
Calls
Social
Knowledge Base
Feedback
Permissions
Log Out
Live Chat
1 chat in queue
Chat with Ravi Kumar • Active
Can you check the status of my refund?
Ravi Kumar 3:18:38 AM negative
I've generated a new invoice and sent it to your email
Aanya Sharma 3:18:42 AM
Type your message...
Transfer
Schedule
Resolve
Send
Current Conversation
Agent
Aanya Sharma
Customer Support
Customer
Ravi Kumar
Active Customer
Waiting Chats
PV Priya Verma
Payment issue
2m ago
AG Amit Gupta
Order cancellation
5m ago
SA Sunita Agarwal
Return request
12m ago
Activity Feed
Divya Malhotra assigned a high priority ticket to Priya Verma
3:18:45 AM
Divya Malhotra escalated issue from Amit Gupta
3:17:45 AM

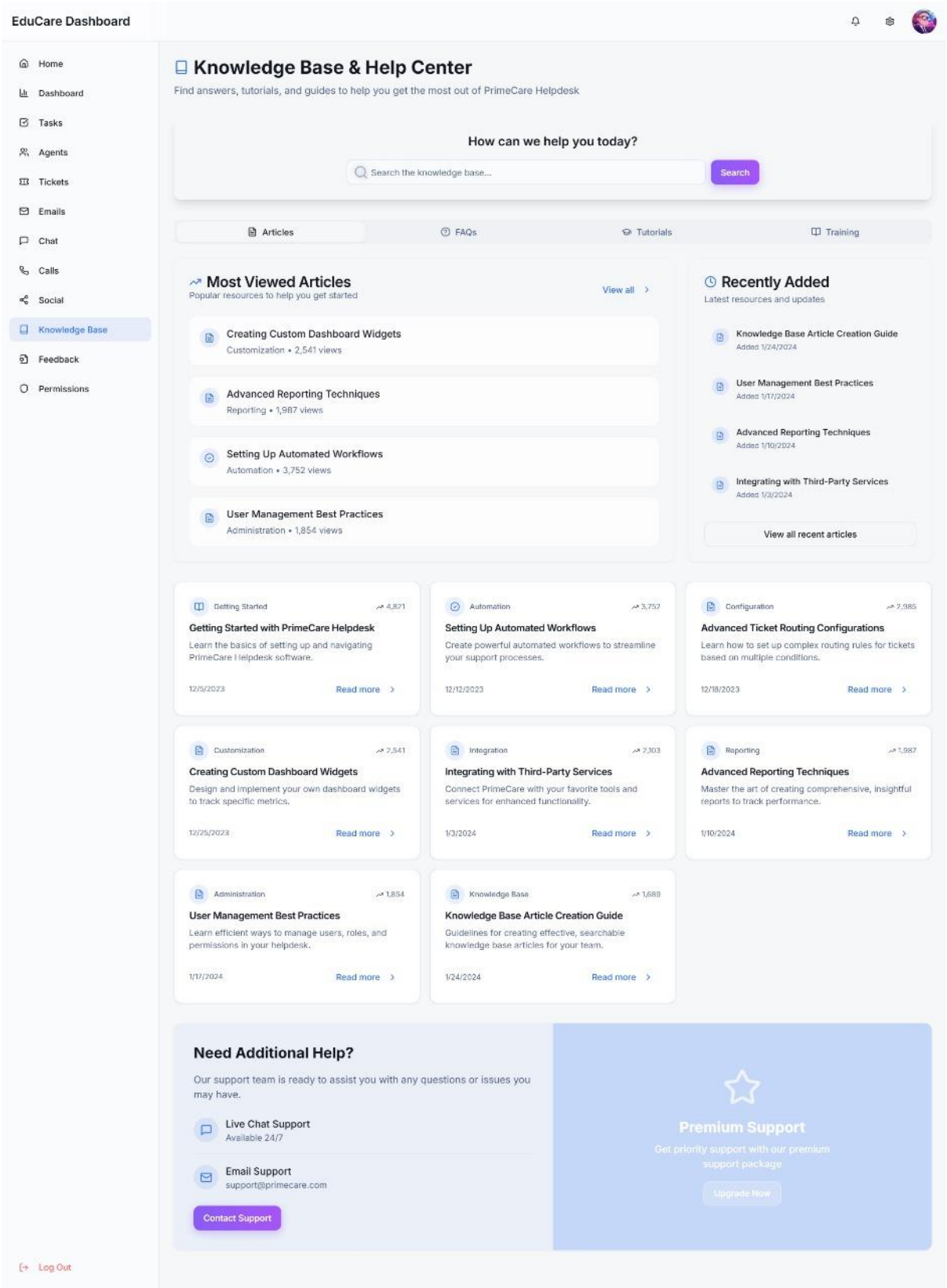
CALLS PAGE


EduCare Dashboard
Home
Dashboard
Tasks
Agents
Tickets
Emails
Chat
Calls
Social
Knowledge Base
Feedback
Permissions
Log Out
Call Center Dashboard
Schedule Call
Refresh Analytics
Total Calls
61
Missed Calls
21
Average Duration
4:51
Calls in Queue
0
Active Call
On Call
0:02
Outgoing
Aditya Kapoor
+91 8098096565
Mute
Transfer
Record
Reject
End Call
Call Notes
Add notes about this call....
Call Analytics
Call Volume - Last 7 Days
24
18
12
6
0
Feb 24
Feb 25
Feb 26
Feb 27
Feb 28
Mar 1
Mar 2
Call Distribution
Agent Performance
40
30
20
10
0
Arjun
Rajesh
Vikram
Call Queue
No calls in queue
Schedule New Call
Recent Alerts
Missed Call from Siddharth Joshi
3/1/2025, 5:18:54 AM
+91 6628829090
Missed Call from Lakshmi Tiwari
3/1/2025, 4:18:54 PM
+91 6112842332
Missed Call from Lakshmi Tiwari
3/1/2025, 9:18:54 PM
Refresh Alerts
Call History
dd-mm-yyyy
All Agents
All Statuses
Apply Filters
Reset
Customer
Agent
Time
Type
Status
Duration
Notes
Aditya Kapoor
Neha Desai
3:19:00 AM
outgoing
ongoing
Sunita Choudhary
Rajesh Gupta
7:18:54 AM
incoming
completed
6:39
Rahul Mahajan
Meera Iyer
2:18:54 PM
outgoing
completed
0:38
Lakshmi Tiwari
Karthik Subramaniam
10:18:54 PM
incoming
completed
0:00
Lakshmi Tiwari
Karthik Subramaniam
10:18:54 PM
outgoing
completed
2:48
Siddharth Joshi
Arjun Sharma
5:18:54 AM
missed
missed
Ananya Reddy
Vikram Malhotra
4:18:54 AM
outgoing
completed
3:06
Rahul Mahajan
Karthik Subramaniam
12:18:54 AM
outgoing
completed
6:50
Lakshmi Tiwari
Divya Patel
4:18:54 PM
missed
missed
Lakshmi Tiwari
Divya Patel
9:18:54 PM
missed
missed

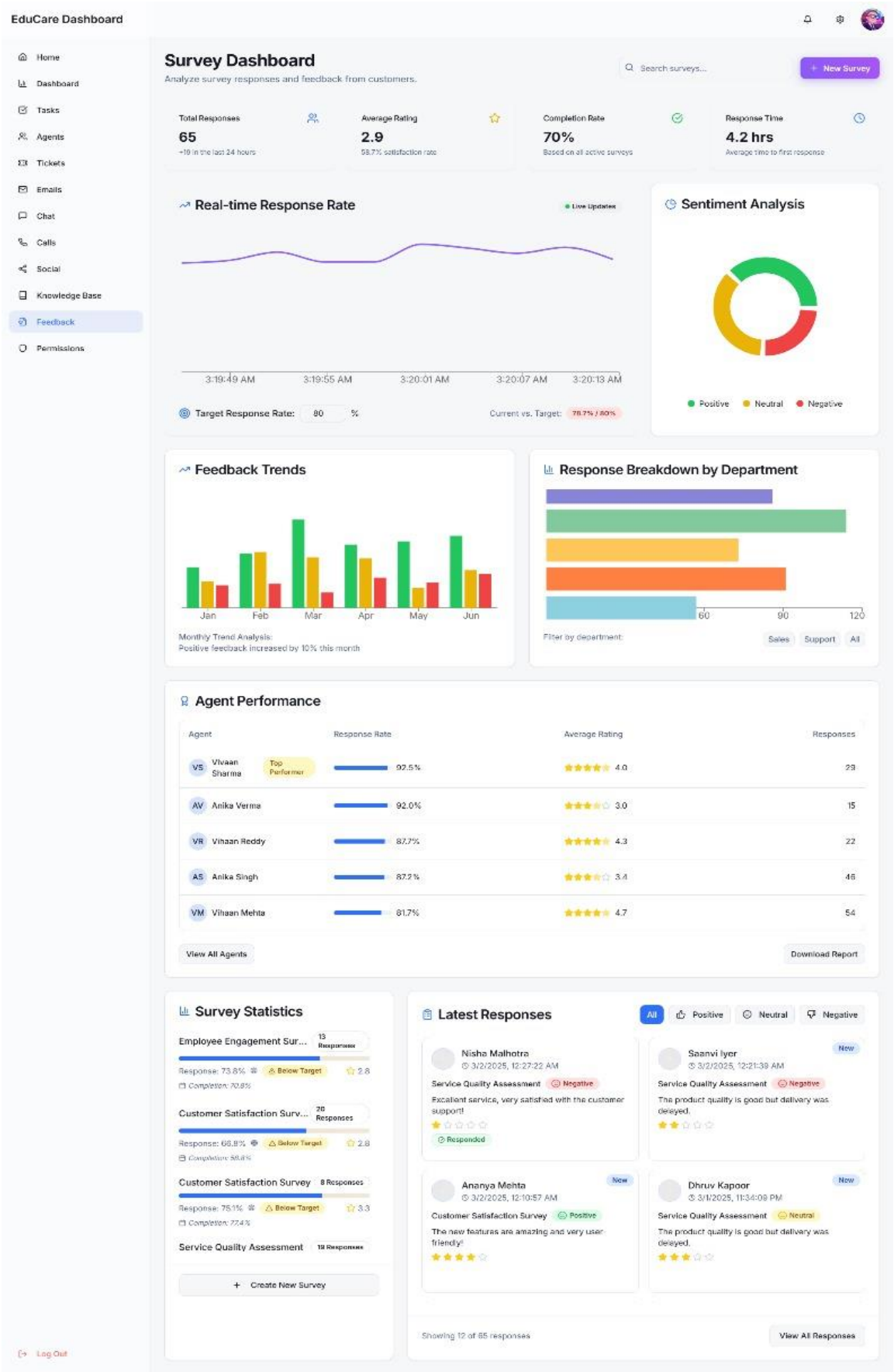
SOCIAL PAGE


EduCare Dashboard
Home
Dashboard
Tasks
Agents
Tickets
Emails
Chat
Calls
Social
Knowledge Base
Feedback
Permissions
Log Out
Social Media Analytics
Real-time insights and engagement metrics
Overview
Social Feed
Campaigns
Engagement Trends
Daily interaction rates across all platforms
80
60
40
20
0
4
5
6
7
8
9
10
Engagement Goals
Overall
73.0% / 80%
Twitter
70.2% / 75%
Facebook
72.0% / 70%
Instagram
79.0% / 85%
Sentiment Analysis
Distribution of feedback sentiment across platforms
Positive
Neutral
Negative
Platform Distribution
Engagement breakdown by social media platform
36
27
18
9
0
Twitter
Facebook
Instagram

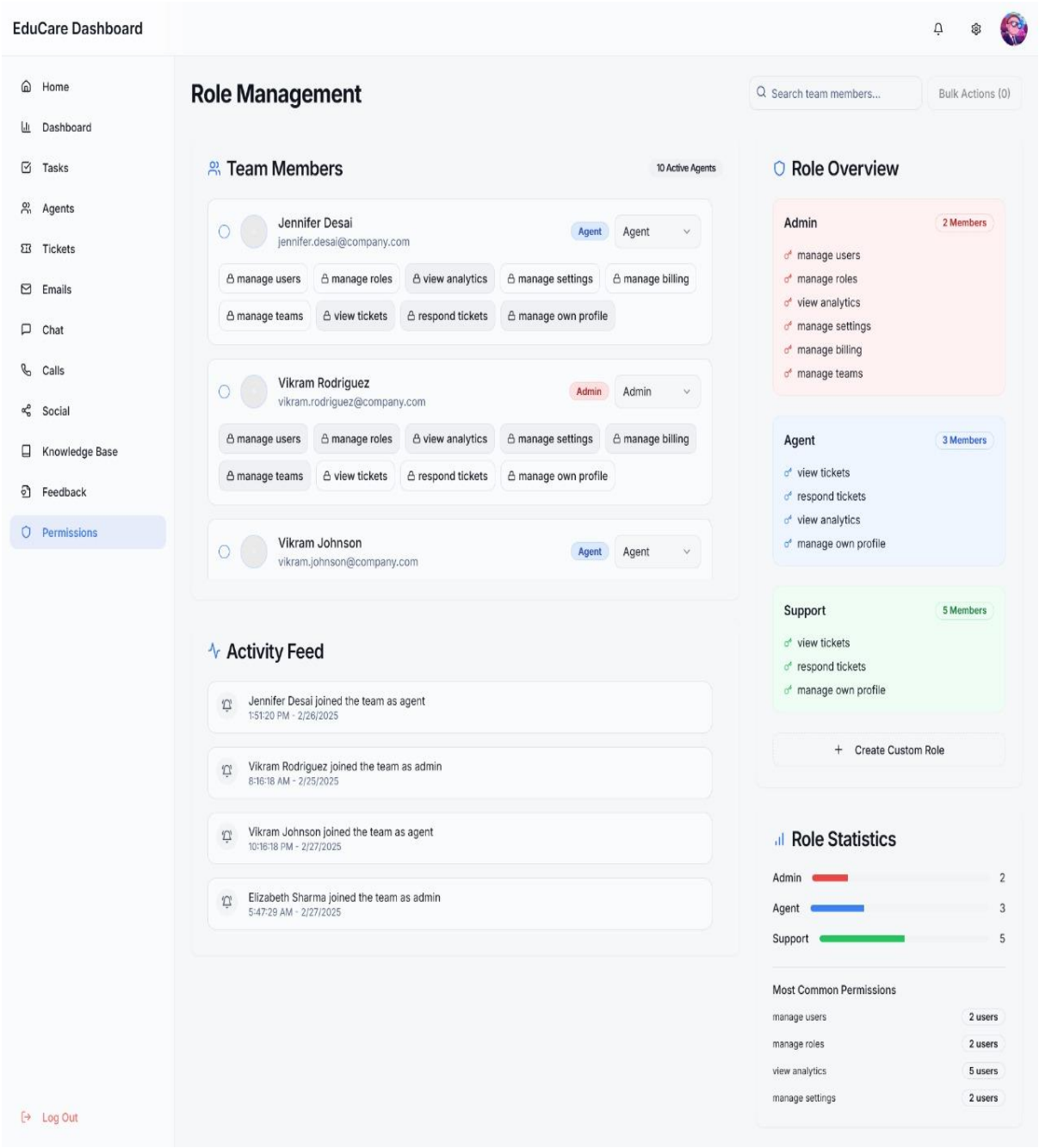
KNOWLEDGE PAGE



FEEDBACK PAGE



PERMISSIONS PAGE



6. TESTING: -

Test techniques and testing strategies used with test case design & test report

Testing is a set of activities that can be planned in advance and conducted systematically. A number of testing strategies have been proposed in literature all provide the software developer with a template for testing and have the following generic characteristics.

- Testing begins at the component level and works outward towards integration of the entire computer-based system.
- Testing is conducted by the developer of the software and for large projects an independent test group may be used.
- Testing and debugging are different activities but debugging must be accommodated in any testing strategy.

STRATEGETIC ISSUES

The following are the strategic issues that must be addressed if a successful software testing strategy is to be implemented.

- Specify product requirements in a quantifiable manner long before testing commences.
- State testing objectives explicitly.
- Understand the needs of the users and develop a profile for each category of users.
- Build robust software that incorporates certain techniques to enable it to test itself.
- Use effective formal technical reviews as a filter prior to testing.
- Conduct formal technical reviews to access the test strategy and test case themselves.
- Develop a continuous improvement approach for the testing process.

UNIT TESTING: -

Unit testing has been applied to test the project. Unit testing focuses on the smallest unit of software design-the software component or module. Using component level design important control paths are tested to uncover errors within the boundary of the module. Unit test is white box oriented and the step can be conducted in parallel for multiple components.

UNIT TEST CONSIDERATION: -

The module interface is tested to ensure that information flows into and out of the program unit under test. The local data structure is examined to ensure that data stored temporarily maintained is integrity during all steps of algorithm execution. Boundary conditions are tested to ensure that the module operates properly at boundaries established to limit or restrict processing. All independent paths through the control center are exercised and finally all error handling paths are tested. Tests of data flow across a module interface are required before any other test is done. Local data structure is examined and their impact should be determined. Some of the most common errors in computation are as follows.

- Misunderstood or incorrect arithmetic precedence
- Incorrect initialization
- Incorrect symbolic representation of an expression

Test cases should uncover errors such as

1. Comparison of different data types
2. Incorrect logical operator or precedence

3. Expectation of equality when precision error makes equality unlikely
4. Improperly modified loop variable.

UNIT TEST PROCEDUES: -

After source code has been developed, reviewed and verified for correspondence to component level design, unit testing is done. A review of design information provides information for establishing test cases. Each test case must be coupled with asset of expected results.

Because a component is not a stand-alone program, drivers or stubs may be developed. Drivers may be considering as a main program that accepts test data and passes it to the component. Stub uses the subordinate's module interface to do minimal data manipulation. Drivers and stubs represent overheads, that software must be written but that is not delivered with the final overhead. If stubs and drivers are kept simple overheads is relatively low.

SECURITY MECHANISM: -

We propose to use window XP operating system, which has multi-user environment and provide data security facility for users. Also, we shall be using SQL-server package as the back-end tool, which has its own security mechanism. Finally, we shall have provision for authenticating user identity through proper verification to increase data security.

Security cases:

In our project only authorized users can open the project,

A login screen is implemented at the opening of the project so that unauthorized users can't open the software and so that data will be secured from outside user.

TEST CASE DESIGN:

System testing is very much essential before actual implementation of the system. All kinds of errors and in-compatibility must be improved before it is ready for user acceptance testing. If all part of system is correct then the system objectives will be successfully achieved, otherwise not. Inadequate testing and non-testing lead to the errors that may not appear until months later. This creates 2 types of problem:

1. The time lag between the cause and the appearance of the problem.

2. The effect of the system errors of files and records within the system.

A small system error can considerably explode into much larger problems. Effective testing earlier in the process translates directly into long-term cost saving from reduced number of errors. Another reason of testing is its utility as a user-oriented vehicle before implementation. First test of the system is to see whether it produces the correct outputs. No other test can be more crucial. Some of the tests, which can have been performed, are given below:

Volume Testing:

In this test we create as many records as would normally be produced to verify the hardware and the software that will function properly. The user is easily asked to provide test data for volume testing. In the system we develop a huge number of records are being tested and the test output shows that the system can hold an amount of data required by the firm.

Stress Testing:

The stress testing is to provide that does not malfunction under peak loads. Unlike volume testing, where time is not a factor, we subject the system to a high volume of data

over short time period. This simulates an on-line environment where high volume of activities occurs spurts.

Recovery Testing:

A forced system failure is induced to test backup recovery procedure for file integration. Inaccurate data are entered to see how the system response in terms of error deduction and protection related to file integrity as a test demonstrate that data program is secure from unauthorized access.

USABILITY DOCUMENTATION AND PROCEDURE:

The usability tests verify the friendly nature of the system. It also tests where an unknown user can handle the system freely or not. A crucial phase of the system life cycle is the successful implementation of the new system design. Implementation simply means covering the new system into operation. These involve creating computer compatible files, training operating staff and installing hardware, terminal and telecommunication network before the system is setup and running critical factor in conversion is not disrupting the functioning of the organization.

Testing of Individual Program:

The individual programs are completed during the program development stage itself. Each program was tested in some test data at the time of coding necessary changes in order to make sure that the programs working properly.

Creating test Data:

Though some test data during individual program development was not sufficient for testing the system as a whole. During the time testing all types of checking has been done depending upon situation.

System testing:

After successfully completion of the individual forms the whole system is run through a series of test to ensure the working of the system as a whole. The effects of testing the entire program are to verify that the program is working properly and according user's specification that were made during the period of system studies.

User training:

User training is an important to be taken into account before implementation.

Testing and implementation

Testing techniques:

White box testing

- Condition testing
- Data flow testing

Black box testing

- Equivalence partitioning
- Boundary value analysis

Testing strategies

- Unit Testing
- Integration testing
- Validation testing

Testing intends the developed to discards preconceived notions of correctness of the software developed and overcome the conflict of interest that occurs when errors are uncovered. Testing is done in order to find the errors that are present in the system. The error occurs when the output of the software does not match with the expected output. In order to detect the error and correct then different testing techniques and testing strategies have been used.

TESTING TECHNIQUES

The common testing techniques are

1. WHITE BOX TESTING
2. BLACK BOX TESTING

WHITE BOX TESTING

It is predicated upon the close examination of procedural detail. Logical paths through the software are tested by providing test cases that exercise specific set of condition or loops It guarantees that

- All independent paths in the module have been executed at least once
- Execution of all logical decision on their both true and false side
- Execution of loops to their boundary values
- Exercise internal data structure to ensure their validity

CONDITIONAL TESTING: Conditional testing is done in order to exercises all logical condition

DATAFLOW TESTING: -The data flow testing method select test paths of a program according to the location of definition and use of variables in the program.

LOOPTESTING: - Loop testing is a white box testing technique that focuses on the validity of the loop construction.

BLACK BOX TESTING: -

Black box testing is done in order to the detect the interface errors. It guarantees that the inputs are properly accepted and outputs are correctly produced and the integrity of the external information is maintained. It

Finds the following common errors

1. interface errors
2. errors in data structure
3. Initialization and termination errors. The difference black box techniques are:

EQIVALANCE PARTIONING: It is the testing method that divides the input domain of the program into classes of data from which different test cases are designed.

BOUNDARY VALUE ANALUSIS: A greater number of errors tend to occur at the boundary of the input domain. Therefore, boundary values analysis is required to derive the test that exercises boundary values.

TESTING STRATEGIES: - Various testing strategies have been proposed but only few of them have been used in this project. They are

UNIT TESTING: It focuses verification effort on the smallest unit of the software design. Important control ate tested to uncover error with in the boundary of the module.

INTEGRATION TESTING: Integration testing is a systematic approach for constructing the program structure while at the same time conducting test to uncover errors associated with interfacing. There are two types of integration they are

TOP-DOWN INTEGRATION

BOTTOM-UP INTEGRATION

VALIDATION TESTING: After the culmination of integration testing, software is completely assembled as package, interfacing errors are un-covered and corrected then the validation testing begins.

SYSTEM TESTING: -

The systems are integrated to eventually from the entire system. After the system is put together system testing is performed to see of the entire requirements are met and if the system performs as specified by the requirement. For example, one module expected by another sub module. If field size of both sub modules is different then occurred at the time of system testing.

SPECIAL SYSTEM TESTING:

There are other tests are in a special category. There are:

- **Peak Load Testing:** It determines whether the system will able to handle the volume of activity that occurs when the system is at the peak of the processing demand. For example, terminals are activated at the same time
- **Storage Testing:** It determines the capacity of the system to store the transaction data on disk or other files.
- **Performance Time Testing:** It determines the length of the time required to process the transaction data by the system. For example, response time for enquiry when system is fully loaded with operating data.
- **Recovery Testing:** It determines the ability of the user to recover data or restart the system after failure.
- **Procedure Testing:** It determines the clarity of the documentation on operation and uses of system by having users do exactly what manual request.

7. FUTURE SCOPE

With the increasing demand for quality education and personalized learning experiences, educational institutions are focusing on enhancing their student services. A Customer Care Management System (CCMS) in the education sector plays a crucial role in improving student engagement, addressing concerns, and streamlining communication between students, parents, teachers, and administrators. The future of education CCMS looks promising, with advancements in artificial intelligence (AI), data analytics, automation, and cloud computing shaping its development.

This explores the future scope of Education Customer Care Management Systems, focusing on emerging trends, technologies, challenges, and opportunities.

1. **Evolution of Education Customer Care Management Systems**

Education Customer Care Management Systems have evolved from basic ticketing and query-resolution tools to AI-driven, multi-channel, cloud-based, and personalized systems. Initially, support was limited to phone calls and emails, but modern CCMS now incorporate:

- **Chatbots & AI-powered assistants**
- **Omnichannel support (email, chat, social media, WhatsApp, etc.)**
- **Automated ticketing & case management**
- **Personalized student support systems**
- **Data-driven insights for decision-making**

As institutions embrace digital transformation, CCMS is expected to evolve further, integrating with Learning Management Systems (LMS), Student Information Systems (SIS), and CRM tools to provide a seamless experience.

2. Key Trends Shaping the Future of Education CCMS

Artificial Intelligence & Machine Learning Integration:

AI and ML will revolutionize CCMS by enabling real-time support, predictive analytics, and personalized assistance. Future applications include:

- AI Chatbots & Virtual Assistants – 24/7 automated support for student queries on admissions, courses, fees, exams, etc.
- Sentiment Analysis – Understanding student concerns by analyzing feedback, emails, and social media comments.
- Predictive Analytics – Identifying potential dropouts and academic risks based on behavioral data.

Omnichannel Support & Self-Service Portals:

- Social media (Facebook, Twitter, LinkedIn, Instagram, WhatsApp, etc.)
- Mobile apps & self-service portals
- Voice assistants like Alexa & Google Assistant

- Self-service portals will allow students to track queries, access FAQs, submit complaints, and chat with virtual agents, reducing response time and workload on human agents.

Cloud-Based & SaaS Solutions

- Anywhere, Anytime Access – Remote access for students, faculty, and support staff.
- Seamless Integration – Easy integration with LMS, SIS, and CRM platforms.
- Cost Efficiency – Reduced infrastructure costs and maintenance.
- Software-as-a-Service (SaaS) models will dominate, allowing educational institutions to subscribe to CCMS without heavy investment in IT infrastructure.

Personalization & Data-Driven Insights

- Student behavior & interaction history
- Academic performance trends
- Past queries & complaints
- Personalized recommendations for courses, scholarships, extracurricular activities, and career guidance will enhance student satisfaction and retention.
- Blockchain for Data Security & Transparency

Automation & Robotic Process Automation (RPA)

- Automation will eliminate manual tasks in education customer care, such as:
- Automated ticket routing & escalation
- Auto-generated responses for common queries
- AI-powered feedback collection & analysis
- This will reduce human workload and improve efficiency.

3. Benefits of Future Education CCMS

Improved Student Engagement & Satisfaction

- Quick resolution of queries enhances student experience.
- 24/7 support ensures accessibility.
- Personalized communication builds trust.

Enhanced Efficiency & Productivity

- Automation reduces workload on human agents.
- AI-driven insights improve decision-making.
- Cloud-based solutions enable seamless collaboration.

Cost Reduction & Resource Optimization

- SaaS-based CCMS eliminates IT infrastructure costs.
- Automation reduces staffing expenses.
- Predictive analytics prevent costly student dropouts.

Better Institutional Reputation & Competitiveness

- A modern CCMS helps institutions stand out.
- Positive student experiences lead to better reviews and recommendations.
- Data-driven decision-making enhances institutional strategy.

4. Challenges & Considerations

Data Privacy & Security

- Protection of student records and interactions is critical.
- Compliance with data privacy laws (GDPR, FERPA, etc.) is necessary.

Adoption Resistance

- Faculty and staff may resist new technologies.
- Proper training and change management strategies are needed.

Cost & Implementation Complexities

- Initial investment in advanced CCMS solutions can be high.
- Integration with existing systems may require customization.

5. Future Opportunities & Innovations

AI-Powered Career Counseling

- CCMS will integrate AI-driven career guidance systems that:
- Recommend career paths based on skills and interests.
- Connect students with job and internship opportunities.

Integration with Virtual & Augmented Reality

- Future CCMS will leverage VR and AR for:
- Virtual campus tours.
- Interactive learning support.

Voice-Based Support Systems

- AI-driven voice assistants will enable:
- Hands-free query resolution.
- Multilingual support for global students.

The future of Education Customer Care Management Systems is dynamic and technology-driven, focusing on AI, automation, personalization, and cloud-based solutions. Institutions that adopt modern, data-driven CCMS will see improved student engagement, enhanced efficiency, and higher retention rates. However, addressing challenges such as data security, cost, and adoption barriers will be crucial for seamless implementation.

With continuous innovation, Education CCMS will become an integral part of the digital education ecosystem, ensuring superior student experiences and institutional success in the years ahead.

8. GLOSSARY

8.1. WEBSITE: -

A website is a collection of Web pages, images, videos or other digital assets that is hosted on one or more web servers, usually accessible via the Internet.

A Web page is a document, typically written in (X) HTML; hat is almost always accessible via HTTP, a protocol that transfers information from the Web server to display in the user's Web browser. All publicly accessible websites are seen collectively as constituting the "World Wide Web".

The pages of a website can usually be accessed from a common root URL called the homepage, and usually reside on the same physical server. The URLs of the pages organize them into a hierarchy, although the hyperlinks between them control how the reader perceives the overall structure and how the traffic flows between the different parts of the site.

Some websites require a subscription to access some or all of their content. Examples of subscription sites include many business sites, parts of many news sites, academic journal sites, gaming sites, message boards, Web-based e-mail, services, social networking websites, and sites providing real-time stock market data. Because they require authentication to view the content, they are technically an Intranet site

History

The World Wide Web was created in 1990 by CERN engineer, Tim Berners-Lee. On 30 April 1993, CERN announced that the World Wide Web would be free to anyone.

Before the introduction of HTML and HTTP other protocols such as file transfer protocol and the gopher protocol were used to retrieve individual files from a server. These protocols offer a simple directory structure which the user navigates and chooses files to download. Documents were most often presented as plain text files without formatting or were encoded in word processor formats.

Overviews

Organized by function a website may be

- ✓ A personal website
- ✓ A commercial website
- ✓ A government website
- ✓ A non-profit organization website

It could be the work of an individual, a business or other organization, and is typically dedicated to some particular topic or purpose. Any website can contain a hyperlink to any other website, so the distinction between individual sites, as perceived by the user, may sometimes be blurred.

Websites are written in, or dynamically converted to, HTML (Hyper Text Markup Language) and are accessed using a software interface classified as an over agent. Weh pages can be viewed or otherwise accessed from a range of computer-based and Internet-enabled devices of various sizes, including desktop computers, laptops, PDAs and cell phones.

A website is hosted on a computer system known as a web server, also called an HTTP server, and these terms can also refer to the software that runs on these systems and that retrieves and delivers the Web pages in response to requests from the website users. Apache

is the most commonly used Web server software (according to Netcraft statistics) and Microsoft's Internet Information Server (IIS) is also commonly used.

Static Website

A Static Website is one that has web pages stored on the server in the same form as the user will view them. It is primarily coded in HTML (Hyper-text Markup Language) A static website is also called a Classic website, a 5-page website or a Brochure website because it simply presents pre-defined information to the user. It may include information about a company and its products and services via text, photos, Flash animation, audio-video and interactive menus and navigation:

This type of website usually displays the same information to all visitors; thus, the information is static. Similar to handing out a printed brochure to customers or clients, static website will generally provide consistent, standard information for an extended period of time. Although the website owner may make updates periodically, it is a manual process to edit the text, photo and other content and may require basic website design skills and software

Dynamic website

A Dynamics Website in one that does not have web pages stored on the server in the same form as the user will view them, based, the web page content changes automatically and/or frequently based on certain criteria. It generally collates information on the hop each time a page is requested.

A website can be dynamic in one of two ways. The first is that the web page code is constructed dynamically, piece by piece. The second is that the web page content displayed varies based on certain criteria. The criteria may be pre-defined rules or may be based on variable user input.

The main purpose behind a dynamic site is that it is much simpler to maintain a few web pages plus a database than it is to build and update hundreds or thousands of individual web pages and links. In one way, a data-driven website is similar to a static site because the information that is presented on the site is still limited to what the website owner has allowed to be stored in the database (data entered by the owner and/or input by users and approved by the owner). The advantage is that there is usually a lot more information stored in a database and made available to users.

A dynamic website also describes its construction or how it is built, and more specifically refers to the code used to create a single web page. A Dynamic Web Page is generated on the fly by piecing together certain blocks of code, procedures or routines. A dynamically-generated web page would call various bits of information from a database and put them together in a pre-defined format to present the reader with a coherent page. It interacts with users in a variety of ways including by reading cookies recognizing users' previous history, session variables, server-side variables etc., or by using direct interaction (form elements, mouseovers, etc.). A site can display the current state of a dialogue between users, monitor a changing situation, or provide information in some way personalized to the requirements of the individual user.

Some countries, for example the UK. have introduced legislation regarding web accessibility.

Software systems

Turning a website into an income source is a common practice for web-developers and website owners. There are several methods for creating a website business which fall into two broad categories, as defined below.

1. Content based sites

Some websites derive revenue by selling advertising space on the site (see contextual ads)

2. Product or service-based sites

Some websites derive revenue by offering products or services. In the case of e-commerce websites, the products or services may be purchased at the website itself, by entering credit card or other payment information into a payment form on the site. While most business websites serve as a shop window for existing brick and mortar businesses, it is increasingly the case that some websites are businesses in their own right, that is, the products they offer are only available for purchase on the web.

Websites occasionally derive income from a combination of these two practices. For example, a website such as an online auctions website may charge the users of its auction service to list an auction, but also display third-party advertisements on the site, from which it derives further income.

Types of websites

There are many varieties of Web sites, each specializing in a particular type of content or use, and they may be arbitrarily classified in any number of ways.

- ✓ **Affiliate:** enabled portal that renders not only its custom CMS but also syndicated content from other content providers for an agreed fee. There are usually three relationship tiers. Affiliate Agencies (e.g., Commission Junction), Advertisers (e.g. eBay) and consumer (e.g. Yahoo).
- ✓ **Archive site:** used to preserve valuable electronic content threatened with extinction. Two examples are: Internet Archive, which since 1996 has preserved billions of old (and new) Web pages; and Google Groups, which in early 2005 was archiving over 845,000,000 messages posted to Usenet news/discussion groups.
- ✓ **Blog (or web log) site:** sites generally used to post online diaries which may include discussion forums (e.g., blogger, Xanga).
- ✓ **Content site:** sites whose business is the creation and distribution of original content (e.g. Slate, About.com).
- ✓ **Corporate website:** used to provide background information about a business, organization, or service.
- ✓ **Commerce site (or ecommerce site):** for purchasing goods, such as Amazon.com, CSN Stores, and Overstock.com.

Community site: a site where persons with similar interests communicate with each other, usually by chat or message boards, such as MySpace,

- ✓ **Community site:** a site where persons with similar interests communicate with each other, usually by chat or message boards, such as Myspace.
- ✓ **City Site:** A site that shows information about a certain city or town and events that takes place in that town. Usually created by the city council or other "movers and shakers".
- ✓ **Database site:** a site whose main use is the search and display of a specific database's content such as the Internet Movie Database or the Political Graveyard.
- ✓ **Development site:** a site whose purpose is to provide information and resources related to software development, Web design and the like.
- ✓ **Directory site:** a site that contains varied contents which are divided into categories and subcategories, such as Yahoo! directory, Google directory and Open Directory project
- ✓ **Directory site:** a site that contains varied contents which are divided into categories and subcategories, such as Yahoo! directory, Google directory and Open Directory Project.
- ✓ **Download site:** strictly used for downloading electronic content, such as software, game demos or computer wallpaper.
- ✓ **Employment site:** allows employers to post job requirements for a position or positions and prospective employees to fill an application.
- ✓ **Fan site:** A web site created and maintained by fans of and for a particular celebrity, as opposed to a web site created, maintained, and controlled by a celebrity through their own paid webmaster. May also be known as a Shrine in the case of certain subjects, such as anime, and mange characters.
- ✓ **Game site:** a site that is itself a game or "playground" where many people come to play (e.g. MSN Games and Pogo.com).
- ✓ **Gambling site:** A site in which you can do non-sports related gambling
- ✓ Geodomain refers to domain names that are the same as those of geographic entities, such as cities and countries. For example, Richmond.com is the geodomain for Richmond, Virginia.
- ✓ **Gambling site:** A site in which you can do non-sports related gambling Geodomain refers to domain names that are the same as those of geographic entities, such as cities and countries. For example, Richmond.com is the geodomain for Richmond, Virginia.
- ✓ **Gripe site**: a site devoted to the critique of a person, place, corporation, government, or institution.
- ✓ **Humor site:** satirizes, parodies or otherwise exists solely to amuse.
- ✓ **Information site:** contains content that is intended to inform visitors, but not necessarily for commercial purposes, such as: RateMyProfessors.com, Free Internet Lexicon and Encyclopedia. Most government, educational and non-profit institutions have an informational site.
- ✓ **Mirror (computing) site:** A complete reproduction of a website.
- ✓ **News site:** similar to an information site, but dedicated to dispensing news and commentary.
- ✓ **Personal homepage:** run by an individual or a small group (such as a family) that contains information or any content that the individual wishes to include. These are usually uploaded using a web hosting service such as Geocities.

- ✓ **Personal homepage:** run by an individual or a small group (such as a family) that contains information or any content that the individual wishes to include. These are usually uploaded using a web hosting service such as GeoCities.
- ✓ **Political site:** A site on which people may voice political views.
- ✓ **Rating site:** A site on which people can praise or disparage what is featured.
- ✓ **Review site:** A site on which people can post reviews for products or services.
- ✓ **Video sharing:** A site that enables user to upload videos, such as YouTube and Google Video.
- ✓ **Shock site:** includes images or other material that is intended to be offensive to most viewers (e.g. rotten.com).
- ✓ **Warez:** a site designed to host and let users download copyrighted materials illegally.
- ✓ **Web portal:** a site that provides a starting point or a gateway to other resources on the Internet or an intranet.
- ✓ **Wiki site:** a site which users collaboratively edit (such as Wikipedia and Wikihow Wiki how).

8.2. HTML

HTML, an initialize of Hypertext Markup Language, is the predominant markup language for Web pages. It provides a means to describe the structure of text-based information in a document by denoting certain text as links, headings, paragraphs, lists, and so on and to supplement that text with interactive forms, embedded images, and other objects. HTML is written in the form of tags, surrounded by angle brackets. HTML can also describe, to some degree, the appearance and semantics of a document, and can include embedded scripting language code (such as JavaScript) which can affect the behaviour of Web browsers and other HTML processors.

Origins

In 1980, physicist Tim Berners-Lee, who was an independent contractor at CERN, proposed and prototyped ENQUIRE, a system for CERN researchers to use and share documents. In 1989, Berners-Lee and CERN data systems engineer Robert Cailliau each submitted separate proposals for an Internet-based hypertext system providing similar functionality. The following year, they collaborated on a joint proposal, the World Wide Web (W3) project, which was accepted by CERN.

First specifications

The first publicly available description of HTML was a document called HTML Tags, first mentioned on the Internet by Berners-Lee in late 1991. It describes 22 elements comprising the initial, relatively simple design of HTML. Thirteen of these elements still exist in HTML 4.

Berners-Lee considered HTML to be, at the time, an application of SGML, but it was not formally defined as such until the mid-1993 publication, by the IETF, of the first proposal for an HTML specification: Berners-Lee and Dan Connolly's "Hypertext Markup Language (HTML)" Internet-Draft, which included an SGML. Document Type Definition to define the grammar. The draft expired after six months, but was notable for its acknowledgment of the NCSA Mosaic browser's custom tag for embedding in-line images, reflecting the IETF's

philosophy of basing standards on successful prototypes. Similarly, Dave Raggett's competing Internet-Draft, "HTML+ (Hypertext Markup Format)", from late 1993, suggested standardizing already-implemented features like tables and fill-out forms.

After the HTML and HTML+ drafts expired in early 1994, the IETF created an HTML Working Group, which in 1995 completed "HTML 2.0", the first HTML specification intended to be treated as a standard against which future implementations should be based. Published as Request for Comments 1866, HTML 2.0 included ideas from the HTML and HTML+ drafts. There was no "HTML 1.0"; the 2.0 designation was intended to distinguish the new edition from previous drafts.

Further development under the auspices of the IETF was stalled by competing interests. Since 1996, the HTML specifications have been maintained, with input from commercial software vendors, by the World Wide Web Consortium (W3C). However, in 2000, HTML also became an international standard (ISO/IEC 15445:2000). The last HTML specification published by the W3C is the HTML 4.01 Recommendation, published in late 1999. Its issues and errors were last acknowledged by errata published in 2001.

HTML markup

HTML markup consists of several key components, including elements (and their attributes), character-based data types, and character references and entity references. Another important component is the document type declaration.

The Hello world program, a common computer program employed for comparing programming languages, scripting languages, and markup languages is made of 8 lines of code in HTML, albeit line breaks are optional: heal

<!DOCTYPE html>

<title>Hello HTML<id>

<head>

<body>

<p>Hello World!<p>

<body>

</html>

Elements

Elements are the basic structure for HTML markup. Elements have two basic properties: attributes and content. Each attribute and each element's content has certain restrictions that must be followed for an HTML document to be considered valid. An element usually has a start tag (e.g. <element-name>) and an end tag (e.g. </element-name>). The element's attributes are contained in the start tag and content is located between the tags (e.g. <element-name attribute="value">Content</element-name>). Some elements, such as
, do not have any content and must not have a closing tag. Listed below are several types of markup elements used in HTML.

Structural markup describes the purpose of text. For example, <h2>Golf</h2> establishes "Golf" as a second-level heading, which would be rendered in a browser in a manner similar to the "HTML markup" title at the start of this section. Structural markup does not denote any specific rendering, but most Web browsers have standardized on how elements should be formatted. Text may be further styled with

Data types

HTML defines several data types for element content, such as script data and stylesheet data, and a plethora of types for attribute values, including IDs, names, URIs, numbers, units of length, languages, media descriptors, colors, character encodings, dates and times, and so on. All of these data types are specializations of character data.

The Document Type Declaration

HTML documents are required to start with a Document Type Declaration (informally, a "doctype"). In browsers, the function of the doctype is selecting the rendering mode-particularly to avoid the quirks mode.

The original purpose of the doctype is to enable validation based on Document Type Definition (DTD) with SGML tools. The DTD to which the DOCTYPE refers contains machine-readable grammar specifying the permitted and prohibited content for a document conforming to such a DTD. Browsers do not read the DTD, however. HTML5 validation is not DTD-based, so in HTML5 the doctype does not refer to a DTD.

8.3. CASCADING STYLE SHEETS

Cascading Style Sheets (CSS) is a style sheet language used to describe the presentation of a document written in a markup language. It's most common application is to style web pages written in HTML, and XHTML, but the language can be applied to any kind of XML document, including SVG and XUL.

CSS can be used locally by the readers of web pages to define colors, fonts, layout, and other aspects of document presentation. It is designed primarily to enable the separation of document content (written in HTML or a similar markup language) from document presentation (written in CSS). This separation can improve content accessibility, provide more flexibility and control in the specification of presentation characteristics, and reduce complexity and repetition in the structural content (such as by allowing for table less web design). CSS can also allow the same markup page to be presented in different styles for different rendering methods, such as on-screen, in print, by voice (when read out by a speech-based browser or screen reader) and on Braille-based, tactile devices. CSS specifies a priority scheme to determine which style rules apply if more than one rule matches against a particular element. In this so-called cascade, priorities or weights are calculated and assigned to rules, so that the results are predictable.

The CSS specifications are maintained by the World Wide Web Consortium (W3C). Internet media type (MIME type) text/css is registered for use with CSS by RFC 2318 (March 1998).

Use of CSS

Prior to CSS, nearly all of the presentational attributes of HTML documents were contained within the HTML markup; all font colors, background styles, element alignments, borders and sizes had to be explicitly described, often repeatedly, within the HTML, CSS allows authors to move much of that information to a separate stylesheet resulting in considerably simpler HTML markup.

Headings (hl elements), sub-headings (h2), sub-sub-headings (h3), etc., are defined structurally using HTML. In print and on the screen, choice of font, size, color and emphasis for these elements is presentational.

Prior to CSS, document authors who wanted to assign such typographic characteristics to, say, all h2 headings had to use the HTML font and other presentational elements for each occurrence of that heading type. The additional presentational markup in the HTML made documents more complex, and generally more difficult to maintain. In CSS, presentation is separated from structure. In print, CSS can define color, font, text alignment, size, borders, spacing, layout and many other typographic characteristics. It can do so independently for on-screen and printed views. CSS also defines non-visual styles such as the speed and emphasis with which text is read out by aural text readers. The W3C now considers the advantages of CSS for defining all aspects of the presentation of HTML pages to be superior to other methods. It has therefore deprecated the use of all the original presentational HTML markup.

History

Style sheets have existed in one form or another since the beginnings of SGML in the 1970s. Cascading Style Sheets were developed as a means for creating a consistent approach to providing style information for web documents. As HTML grew, it came to encompass a wider variety of stylistic capabilities to meet the demands of web developers. This evolution gave the designer more control over site appearance but at the cost of HTML becoming more complex to write and maintain. Variations in web browser implementations made consistent site appearance difficult, and users had less control over how web content was displayed.

To improve the capabilities of web presentation, nine different style sheet languages were proposed to the W3C's www-style mailing list. Of the nine proposals, two were chosen as the foundation for what became CSS: Cascading HTML Style Sheets (CHSS) and Stream-based Style Sheet Proposal (SSP). First, Håkon Wium Lie (now the CTO of Opera Software) proposed Cascading HTML Style Sheets (CHSS) in October 1994, a language which has some resemblance to today's CSS. Bert Bos was working on a browser called Argo which used its own style sheet language, Stream-based Style Sheet Proposal (SSP). Lie and Bos worked together to develop the CSS standard (the 'H' was removed from the name because these style sheets could be applied to other markup languages besides HTML).

Unlike existing style languages like DSSSL and FOSI, CSS allowed a document's style to be influenced by multiple style sheets. One style sheet could inherit or "cascade" from another, permitting a mixture of stylistic preferences controlled equally by the site designer and user.

Håkon's proposal was presented at the "Mosaic and the Web" conference in Chicago, Illinois in 1994, and again with Bert Bos in 1995. Around this time, the World Wide Web Consortium was being established; the W3C took an interest in the development of CSS, and it organized a workshop toward that end chaired by Steven Pemberton. This resulted in W3C adding work on CSS to the deliverables of the HTML editorial review board

ERB). Håkon and Bert were the primary technical staff on this aspect of the project, with additional members, including Thomas Reardon Microsoft participating as well. By the end of 1996, CSS was ready to become official, and the CSS level 1 Recommendation was published in December.

Development of HTML, CSS, and the DOM had all been taking place in one group, the HTML Editorial Review Board (ERB). Early in 1997, the ERB was split into three working groups: HTML Working group, chaired by Dan Connolly of W3C; DOM Working group, chaired by Lauren Wood of Soft Quad; and CSS Working group, chaired by Chris Lilley of W3C.

The CSS Working Group began tackling issues that had not been addressed with CSS level 1, resulting in the creation of CSS level 2 on November 4, 1997. It was published as a W3C Recommendation on May 12, 1998. CSS level 3, which was started in 1998, is still under development as of 2008. In 2005 the CSS Working Groups decided to enforce the requirements for standards more strictly. This meant that already published standards like CSS 2.1, CSS 3 Selectors and CSS 3 Text were pulled back from Candidate Recommendation to Working Draft level.

Advantages

By combining CSS with the functionality of a Content Management System, a considerable amount of flexibility can be programmed into content submission forms. This allows a contributor, who may not be familiar or able to understand or edit CSS or HTML code to select the layout of an article or other page they are submitting on-the-fly, in the same form. For instance, a contributor, editor or author of an article or page might be able to select the number of columns and whether or not the page or article will carry an image. This information is then passed to the Content Management System, and the program logic will evaluate the information and determine, based on a certain number of combinations, how to apply classes and IDs to the HTML elements, therefore styling and positioning them according to the pre-defined CSS for that particular layout type. When working with large-scale, complex sites, with many contributors such as news and informational sites, this advantage weighs heavily on the feasibility and maintenance of the project. When CSS is used effectively, in terms of inheritance and "cascading," a global stylesheet can be used to affect and style elements site-wide. If the situation arises that the styling of the elements should need to be changed or adjusted, these changes can be made easily, simply by editing a few rules in the global stylesheet. Before CSS, this sort of maintenance was more difficult, expensive and time-consuming.

8.4. TAILWIND CSS

Tailwind CSS is a modern utility-first CSS framework that provides a set of pre-defined utility classes to help developers design responsive and customizable user interfaces efficiently. Unlike traditional CSS frameworks like Bootstrap, which rely on pre-built components, Tailwind enables developers to style elements directly in HTML using utility classes, eliminating the need for writing custom CSS.

One of the main advantages of Tailwind is its flexibility and scalability. It allows developers to create unique designs without being restricted by predefined styles. By using a structured system of spacing, colors, typography, and layout utilities, Tailwind ensures consistency across a project while maintaining ease of customization.

Another key aspect of Tailwind is its performance optimization. With features like Just-In-Time (JIT) compilation, it generates only the necessary CSS, keeping the final output minimal and efficient. This reduces unused styles and improves website loading speed.

Tailwind CSS is widely used in modern web development due to its ease of integration with frameworks like React, Vue, Angular, and Next.js. It is particularly popular in teams and projects that require rapid prototyping, as it allows for quick styling changes without modifying multiple CSS files. Overall, Tailwind CSS simplifies the web development workflow by providing an intuitive, class-based approach to styling while ensuring responsiveness, maintainability, and high performance.

Key features of tailwind CSS are:

- Utility-first approach
- Highly customizable
- Responsive design with mobile-first approach
- Dark mode support
- JIT (just-in-time) compiler

8.5. REACT JAVA SCRIPT

React.js is an open-source JavaScript library developed by Facebook in 2013 to simplify the process of building interactive and efficient user interfaces. It is primarily used for developing single-page applications (SPAs) and mobile applications (via React Native). Unlike traditional JavaScript frameworks that manipulate the real DOM directly, React uses a Virtual DOM, which enhances performance by updating only the parts of the UI that have changed, rather than reloading the entire page.

React.js (or simply React) is a popular JavaScript library used for building dynamic and interactive user interfaces, particularly for single-page applications. Developed by Facebook, it allows developers to create reusable UI components and manage the state efficiently using a virtual DOM, which enhances performance by updating only the necessary parts of the webpage. React follows a declarative programming approach, making it easier to design complex UIs by focusing on how the UI should look in different states. It also supports JSX, a syntax extension that allows writing HTML-like code within JavaScript. With features like component-based architecture, hooks for managing state and side effects, and strong

community support, React has become one of the most widely used front-end frameworks in modern web development.

Features

1. Component-Based Architecture

React follows a modular approach, where the UI is broken down into small, reusable components. Each component has its own logic and rendering, making it easy to maintain and scale applications.

2. JSX (JavaScript XML)

React uses JSX, a syntax extension that allows developers to write HTML-like code directly within JavaScript. This makes code more readable and easier to write.

3. Virtual DOM

Instead of updating the entire DOM whenever there is a change, React creates a Virtual DOM, compares it with the previous state, and updates only the necessary elements. This results in faster performance and better user experience.

4. State Management

React provides a state mechanism to manage dynamic data and re-render components efficiently when state changes. React also supports external state management libraries like Redux, Context API, and Recoil for handling complex application states.

5. Hooks (Functional Programming)

With the introduction of React Hooks in version 16.8, developers can manage state and side effects in functional components without relying on class components. Hooks like useState, useEffect, and useContext simplify component logic and improve readability.

6. React Router

React provides React Router, a library that enables seamless navigation between different pages in SPAs without refreshing the page.

7. Server-Side Rendering (SSR) & Static Site Generation (SSG)

React supports Next.js, a framework that enables server-side rendering (SSR) and static site generation (SSG) to improve SEO and performance.

Advantages:

- Fast Performance due to Virtual DOM.
- Reusable Components lead to faster development and easy maintenance.
- Strong Community Support with numerous third-party libraries and tools.
- SEO-Friendly when used with SSR frameworks like Next.js.
- Easy Integration with other libraries like Redux, GraphQL, and Material-UI.

8.6. TYPESCRIPT

TypeScript is a strongly typed programming language that builds on JavaScript by adding static types. Developed by Microsoft, it is designed to enhance JavaScript's scalability and maintainability, making it ideal for large codebases. TypeScript introduces features like type annotations, interfaces, generics, and Enums, which help developers catch errors at compile time rather than runtime. It also supports modern JavaScript features and compiles down to plain JavaScript, ensuring compatibility with existing projects. With its object-oriented capabilities and strong tooling support, TypeScript is widely used in web development, especially in frameworks like Angular, React, and Node.js applications.

TypeScript is a statically typed, object-oriented programming language that extends JavaScript with powerful features like type safety, interfaces, generics, and modern object-oriented programming principles. Developed and maintained by Microsoft, TypeScript was first released in 2012 to address JavaScript's limitations in large-scale applications. Since then, it has gained immense popularity among developers, especially those working on enterprise-level applications, because of its ability to catch errors early, improve code organization, and enhance developer productivity.

At its core, TypeScript is a superset of JavaScript, meaning all valid JavaScript code is also valid TypeScript code. The primary advantage of TypeScript lies in its optional static typing, which allows developers to define types for variables, function parameters, and return values, reducing runtime errors and improving maintainability.

Features

TypeScript is a superset of JavaScript that adds static typing and additional features to improve code quality and maintainability. Here are some key features of TypeScript:

- Static Typing- Allows developers to define types for variables, function parameters, and return values. Helps catch type-related errors at compile time.
- Type Inference- TypeScript can automatically infer types even if they are not explicitly declared.
- Interfaces- Enables defining contracts for objects, ensuring consistency and reducing bugs.
- Classes and Object-Oriented Programming (OOP)- Supports classes, inheritance, interfaces, access modifiers (public, private, protected), and more.
- Generics- Allows creating reusable and type-safe components or functions.
- Enums- Provides a way to define a set of named constants, making code more readable.
- Modules and Namespaces- Supports modular programming with import and export. Helps in organizing and maintaining large codebases.

8.7. Microsoft Visual Studio

Microsoft Visual Studio is an Integrated Development Environment (IDE) from Microsoft. It can be used to develop console and Graphical user interface applications. along with Windows Forms applications, web sites, web applications, and web services in both native code together with managed code for all platforms supported by Microsoft Windows, Windows Mobile, Windows CE, .NET Framework, .NET Compact Framework and .

Visual Studio includes a code editor supporting IntelliSense as well as code refactoring. The integrated debugger works both as a source-level debugger and a machine-level debugger. Other built-in tools include a forms designer for building GUI applications, web designer, class designer, and database schema designer. It allows plug-ins to be added that enhance the functionality at almost every level - including adding support for source control systems (like Subversion and Visual SourceSafe) to adding new toolsets like editors and visual designers for domain-specific languages or toolsets for other aspects of the software development lifecycle (like the Team Foundation Server client: Team Explorer).

Visual Studio supports languages by means of language services, which allow any programming language to be supported (to varying degrees) by the code editor and debugger, provided a language-specific service has been authored. Built-in languages include C/C++ (via Visual C++), VB.NET (via Visual Basic .NET), and C# (via Visual C#). Support for other languages such as Chrome, F#, Python, and Ruby among others has been made available via language services which are to be installed separately. It also supports XML/XSLT, HTML/XHTML, JavaScript and CSS. Language-specific versions of Visual Studio also exist which provide more limited language services to the user. These individual packages are called Microsoft Visual Basic, Visual J#, Visual C#, and Visual C++.

Currently, Visual Studio 2008 and 2005 Professional Editions, along with language-specific versions (Visual Basic, C++, C#, J#) of Visual Studio 2005 are available to students as downloads free of charge via Microsoft's Dream Spark program. Visual Studio 2010 is currently in development.

8.8. C# Sharp (programming language)

C# (pronounced C Sharp) is a multi-paradigm programming language that encompasses functional, imperative, generic, object-oriented (class-based), and component-oriented programming disciplines. It was developed by Microsoft as part of the .NET initiative and later approved as a standard by ECMA (ECMA-334) and ISO (ISO/IEC 23270). C# is one of the 44 programming languages supported by the .NET Framework's Common Language Runtime.

C# is intended to be a simple, modern, general-purpose, object-oriented programming language. Anders Hejlsberg, the designer of Delphi, leads the team which is developing C#. It has an object-oriented syntax based on C++ and is heavily influenced by other programming languages such as Delphi and Java, It was initially named Cool, which stood for "C like Object

Oriented Language". However, in July 2000, when Microsoft made the project public, the name of the programming language was given as C#. The most recent version of the language is 3.0 which was released in conjunction with the .NET Framework 3.5 in 2007. The next proposed version, 4.0, is in development.

Design goals

The ECMA standard lists these design goals for C#:

- C# is intended to be a simple, modern, general-purpose, object-oriented programming language.
- Because software robustness, durability and programmer productivity are important, the language should include strong type checking, array bounds checking. detection of attempts to use uninitialized variables, source code portability, and automatic garbage collection.
- The language is intended for use in developing software components that can take advantage of distributed environments.
- Programmer portability is very important, especially for those programmers already familiar with C and C++,
- Support for internationalization is very important.
- C# is intended to be suitable for writing applications for both hosted and embedded systems, ranging from the very large that use sophisticated operating systems, down to the very small having dedicated functions.
- Although C# applications are intended to be economical with regard to memory and processing power requirements, the language is not intended to compete directly on performance and size with C or assembly language.

History

In 1996, Sun Microsystems released the Java programming language with Microsoft soon purchasing a license to implement it in their operating system. Java was originally meant to be a platform independent language, but Microsoft, in their implementation, broke their license agreement and made a few changes that would essentially inhibit Java's platform-independent capabilities. Sun filed a lawsuit and Microsoft settled, deciding to create their own version of a partially compiled, partially interpreted object-oriented programming language with syntax closely related to that of C++.

During the development of NET, the class libraries were originally written in a language/compiler called Simple Managed C (SMC). In January 1999, Anders Hejlsberg formed a team to build a new language at the time called Cool, which stood for "C like Object Oriented Language". Microsoft had considered keeping the name "Cool" as the final name of the language, but chose not to do so for trademark reasons. By the time the NET project was publicly announced at the July 2000 Professional Developers Conference, the language had been renamed C#, and the class libraries and ASP.NET runtime had been ported to C#.

C#'s principal designer and lead architect at Microsoft is Anders Hejlsberg, who was previously involved with the design of Visual J++, Borland Delphi, and Turbo Pascal. In

interviews and technical papers, he has stated that flaws in most major programming languages (e.g. C++, Java, Delphi, and Smalltalk) drove the fundamentals of the Common Language Runtime (CLR), which, in turn, drove the design of the C# programming language itself. Some argue that C# shares roots in other languages.

8.9. ASP.Net

Server Application Development

Server-side applications in the managed world are implemented through runtime hosts. Unmanaged applications host the common language runtime, which allows your custom managed code to control the behavior of the server. This model provides you with all the features of the common language runtime and class library while gaining the performance and scalability of the host server.

The following illustration shows a basic network schema with managed code running in different server environments. Servers such as IIS and SQL Server can perform standard operations while your application logic executes through the managed code.

Server-side managed code

ASP.NET is the hosting environment that enables developers to use the .NET Framework to target Web-based applications. However, ASP.NET is more than just a runtime host; it is a complete architecture for developing Web sites and Internet-distributed objects using managed code. Both Web Forms and XML Web services use IIS and ASP.NET as the publishing mechanism for applications, and both have a collection of supporting classes in the .NET Framework.

XML Web services, an important evolution in Web-based technology, are distributed, server-side application components similar to common Web sites. However, unlike Web-based applications, XML Web services components have no UI and are not targeted for browsers such as Internet Explorer and Netscape Navigator. Instead, XML Web services consist of reusable software components designed to be consumed by other applications, such as traditional client applications, Web-based applications, or even other XML. Web services. As a result, XML Web services technology is rapidly moving application development and deployment into the highly distributed environment of the Internet.

If you have used earlier versions of ASP technology, you will immediately notice the improvements that ASP.NET and Web Forms offers. For example, you can develop Web Forms pages in any language that supports the NET Framework. In addition, your code no longer needs to share the same file with your HTTP text (although it can continue to do so if you prefer). Web Forms pages execute in native machine language because, like any other managed application, they take full advantage of the runtime. In contrast, unmanaged ASP pages are always scripted and interpreted. ASP.NET pages are faster, more functional, and easier to develop than unmanaged ASP pages because they interact with the runtime like any managed application.

The NET Framework also provides a collection of classes and tools to aid in development and consumption of XML. Web services applications. XML Web services are built on standards such as SOAP (a remote procedure-call protocol), XML (an extensible data format),

and WSDL (the Web Services Description Language). The .NET Framework is built on these standards to promote interoperability with non-Microsoft solutions.

Finally, like Web Forms pages in the managed environment, your XML Web service will run with the speed of native machine language using the scalable communication of IIS.

Active Server Pages.NET

ASP.NET is a programming framework built on the common language runtime that can be used on a server to build powerful Web applications. ASP.NET offers several important advantages over previous Web development models:

- **Enhanced Performance**. ASP.NET is compiled common language runtime code running on the server. Unlike its interpreted predecessors, ASP.NET can take advantage of early binding, just-in-time compilation, native optimization, and caching services right out of the box. This amounts to dramatically better performance before you ever write a line of code.
- **World-Class Tool Support**. The ASP.NET framework is complemented by a rich toolbox and designer in the Visual Studio integrated development environment. WYSIWYG editing, drag-and-drop server controls, and automatic deployment are just a few of the features this powerful tool provides.
- **Power and Flexibility**. Because ASP.NET is based on the common language runtime, the power and flexibility of that entire platform is available to Web application developers. The .NET Framework class library, Messaging, and Data Access solutions are all seamlessly accessible from the Web. ASP.NET is also language-independent, so you can choose the language that best applies to your application or partition your application across many languages. Further, common language runtime interoperability guarantees that your existing investment in COM-based development is preserved when migrating to ASP.NET.
- **Simplicity**. ASP.NET makes it easy to perform common tasks, from simple form submission and client authentication to deployment and site configuration. For example, the ASP.NET page framework allows you to build user interfaces that cleanly separate application logic from presentation code and to handle events in a simple, Visual Basic - like forms processing model. Additionally, the common language runtime simplifies development, with managed code services such as automatic reference counting and garbage collection.
- **Manageability**. ASP.NET employs a text-based, hierarchical configuration system, which simplifies applying settings to your server environment and Web applications. Because configuration information is stored as plain text, new settings may be applied without the aid of local administration tools. This "zero local administration" philosophy extends to deploying ASP.NET Framework applications as well. An ASP.NET Framework application is deployed to a server simply by copying the necessary files to the server. No server restart is required, even to deploy or replace running compiled code.
- **Scalability and Availability**. ASP.NET has been designed with scalability in mind, with features specifically tailored to improve performance in clustered and multiprocessor environments. Further, processes are closely monitored and managed by the ASP.NET

runtime, so that if one misbehaves (leaks, deadlocks), a new process can be created in its place, which helps keep your application constantly available to handle requests.

- **Customizability and Extensibility**. ASP.NET delivers a well-factored architecture that allows developers to "plug-in" their code at the appropriate level. In fact, it is possible to extend or replace any subcomponent of the ASP.NET runtime with your own custom-written component. Implementing custom authentication or state services has never been easier. Security. With built in Windows authentication and per-application configuration, you can be assured that your applications are secure.

8.10. Google fire base (2024)

DATABASE

Firebase offers two powerful NoSQL databases:

Cloud Firestore: A scalable and flexible NoSQL database with real-time synchronization, offline capabilities, and automatic scaling. In 2024, Firestore now supports enhanced indexing, complex queries, and lower latency for high-performance applications.

Realtime Database: Designed for real-time applications like chat apps and live updates, Firebase's Realtime Database has seen improvements in query performance, security, and multi-region support.

Both databases now offer better cost optimization tools, helping developers reduce read/write costs through caching and indexing strategies.

Google firebase 2024

Google Firebase remains one of the most powerful platforms for building and managing web and mobile applications in 2024. It offers a suite of cloud-based services, including real-time databases, authentication, cloud storage, analytics, and machine learning capabilities. Firebase helps developers create scalable, serverless applications with seamless backend integration, reducing the need for managing complex infrastructure.

Firebase in 2024 is more powerful, scalable, and AI-driven than ever before. With enhanced security, improved database performance, faster hosting, and better AI integration, Firebase remains a top choice for developers building modern applications. Whether you're a startup looking for a cost-effective backend or an enterprise needing scalable infrastructure, Firebase continues to provide the tools needed to build and grow digital products efficiently.

Google Firebase has evolved significantly in 2024, solidifying its position as a comprehensive platform for building and managing web and mobile applications. It offers a suite of cloud-based services that help developers create scalable, serverless applications without worrying about backend infrastructure. With a strong focus on real-time data, security, and AI-driven enhancements, Firebase remains a go-to solution for startups and enterprise applications alike.

Google Firebase in 2024 remains a powerful platform for building and managing web and mobile applications. It provides backend services such as authentication, real-time databases, cloud storage, hosting, and machine learning capabilities.

FEATURES OF GOOGLE FIREBASE (2024)

1. Authentication and Security Enhancements

Firebase Authentication continues to provide seamless login solutions, supporting email/password authentication, phone numbers, and third-party logins (Google, Facebook, Apple, etc.). In 2024, Firebase has integrated advanced AI-driven fraud detection, multi-factor authentication (MFA) improvements, and more robust identity protection features. These enhancements help developers secure their applications from automated attacks, credential stuffing, and bot-driven abuse.

2. Firebase Hosting: Faster and More Secure

Firebase Hosting continues to provide a fast, secure, and reliable hosting solution for web applications. With automatic SSL certification, global CDN integration, and streamlined deployment via Firebase CLI, developers can deploy their applications quickly. In 2024, Firebase Hosting includes improved edge caching, IPv6 support, and AI-powered performance monitoring to ensure better load times and uptime reliability.

3. Cloud Functions: Serverless Backend with AI Enhancements

Cloud Functions for Firebase remains a crucial feature for serverless computing. It allows developers to execute backend code in response to events triggered by Firebase services or HTTP requests. The latest updates bring: Better execution speed and lower cold start times, AI-driven optimizations to auto-scale based on demand, More integrations with Google Cloud AI services, allowing developers to add features like natural language processing and image recognition effortlessly.

4. Firebase Machine Learning: AI-Powered Features

Firebase ML has grown significantly, offering AI-powered capabilities that help developers integrate features like text recognition, face detection, and sentiment analysis. In 2024, Firebase ML provides improved AutoML support, allowing developers to train and deploy custom models with minimal coding.

5. Firebase Analytics and Performance Monitoring

Firebase Analytics continues to be an essential tool for understanding user behaviour and application performance. The 2024 updates include: Real-time event tracking with enhanced user segmentation, AI-driven anomaly detection to identify unusual patterns in user behaviour, Better integration with Google Ads and BigQuery for deeper data insights.

6. Integration with Google Cloud Services

Firebase is now more tightly integrated with Google Cloud, making it easier for developers to scale their applications while leveraging enterprise-level tools. Features like BigQuery integration, AI-powered data processing, and Kubernetes support help teams build more sophisticated applications.

8.11. ADO.NET

ADO.NET is a set of computer software components that can be used by programmers to access data and data services. It is a part of the base class library that is included with the Microsoft .NET Framework. It is commonly used by programmers to access and modify data stored in relational database systems, though it can also be used to access data in non-relational sources. ADO.NET is sometimes considered an evolution of ActiveX Data Objects (ADO) technology, but was changed so extensively that it can be considered an entirely new product

Architecture

These classes provide access to a data source, such as a Microsoft SQL Server or Oracle database and OLEDB data provider. Each data source has its own set of provider objects, but they each have a common set of utility classes:

- Connection: Provides a connection used to communicate with the data source. Also acts as an abstract factory for command objects.
- Command: Used to perform some action on the data source, such as reading, updating, or deleting relational data.
- Parameter: Describes a single parameter to a command. A common example is a parameter to a stored procedure.
- DataAdapter: A bridge used to transfer data between a data source and a Dataset object (see below).
- DataReader: Used to efficiently process a large list of results one record at a time. It allows records to be accessed in a read-only, forward-only mode, i.e., records have to be accessed in sequential order; they can neither be randomly accessed nor can a record which has been processed previously be accessed again.

9. CONCLUSION

An education customer care management system (CRM) proves to be a pivotal tool for educational institutions, significantly enhancing student engagement, streamlining admissions processes, and optimizing communication with all stakeholders by centralizing data, automating tasks, and facilitating personalized interactions, ultimately leading to improved student retention, better decision-making, and a more positive overall educational experience for everyone involved.

The conclusion for customer care management systems in education emphasizes the importance of strong customer relationships, especially during times of economic change. It highlights how speech automation can help maintain and grow these relationships while managing costs.

- **Customer relationships are crucial:**
- In periods of economic change, customer relationships act as a barometer of an organization's health.
- **Speech automation is beneficial:**
- It can be used to maintain and grow customer relationships while managing spending.
- **Good customer service is vital:**
- Satisfied customers are more likely to return, contributing to healthy profits and a good reputation.
- **Continuous improvement is necessary:**
- Organizations should use market research and data to understand customer needs and preferences, ensuring they receive good service consistently.
- **Customer satisfaction leads to return business:**
- If customers consistently receive good service, they are more likely to return.

10. BIBLIOGRAPHY

ASP.NET

- ASP.NET (Black Book).
- Professional ASP.NET (Wrox Publication).
- C# Vijaymukhi.
- ASP.NET Complete Reference.
- Software engineering Concepts By Roger S. Presman
- UML IN A NUTSHELL by Alhir
- Fundamentals of Software Engineering by Rajib Mall
- Google fire base 2024

ASP.NET Tutorials and Code examples

Websites

1.http://www.topclickmedia.co.uk/static-html

2.www.w3schools.com/aspnet/default.asp

3.www.Asp.net-tutorials.com

4.www.tutorialspoint.com/asp.net/index.htm

5.www.expertrating.com/courseware/dotnetcourse/dotnet-tutorial.asp

6.www.csharp-station.com/Tutorial.aspx

7.www.ateryasoft.com

8.www.csharp.net-tutorials.com

9.www.functionx.com/csharp