Project Report

On

Task Management

(Tasker)

P.G Department of Computer Science & Application

For the partial fulfilment of the requirement for the award of

M.Sc. Computer Science

Session 2019 – 2021

Supervised By:

Dr. Anureet Kaur

Submitted By:

Varunbeer Singh

Khalsa College, Amritsar

ACKNOWLEDGEMENT

No task is single man’s effort. Cooperation and coordination of various people at various places go into the successful implementation. It is a great pleasure to have the opportunity to extend our heart that felt thanks to everybody who helped me through the successful completion of this project.

Here, I found this golden chance to acknowledge all those people who had blessed, encouraged and supported me technically and morally through all the phases of my project. I take this opportunity to express my profound sense of gratitude. I thank almighty GOD for giving me this opportunity to express gratitude to all those who helped in successful completion of this project.

I am highly obliged to **Dr. Anureet Kaur**, who helped me as possible as she could. She always had been with me whenever she could and encouraged me every time. She gave me her precious time and also gave some important logics. She encouraged me to rectify the defaults there is and timely completion of my project. I am indebted to my friends for providing me with the inspiration to do project. My special thanks to my parents who support me emotionally & encouraged me to complete my project.

Varunbeer Singh

College Roll no. 19411007

Uni. Roll no. 19110020656

Masters of Computer Science

Final Year

Khalsa College, Amritsar

DECLARATION

I swear that the project report submitted by me entitled **'Tasker'** in partial fulfilment for **Masters of Computer Science (M.Sc. CS), Khalsa College Amritsar** is an authentic record of work carried out by me in the final year of M.Sc. Degree under the guidance of **Dr. Anureet Kaur**.

The matter embodied in this project is a genuine work done by me and has not been submitted whether to this College / University or to any other University / Institute for the fulfilment of the requirement of any course of study.

Varunbeer Singh

SUPERVISORY CERTIFICATE

This is to be certifying that the project submitted to **Khalsa College Amritsar** under the title **“Tasker”** developed by **Varunbeer Singh** is a bona fide and unique piece of work carried out under my supervision and guidance. He worked hard to collect information and to develop this project. I wish best of luck for future.

Dr. Anureet Kaur

P.G. Department of Computer Science & Applications

Khalsa College Amritsar

Preface

Task management is the process of managing a task through its life cycle. It involves planning, testing, tracking, and reporting. Task management can help either individual achieve goals, or groups of individuals collaborate and share knowledge for the accomplishment of collective goals. Tasks are also differentiated by complexity, from low to high.

Effective task management requires managing all aspects of a task, including its status, priority, time, human and financial resources assignments, recurrence, dependencies, notifications and so on. These can be lumped together broadly into the basic activities of task management.

Managing multiple individuals or team tasks may be assisted by specialized software, for example workflow or project management software.

Task management may form part of project management and process management and can serve as the foundation for efficient workflow in an organization. Project managers adhering to task-oriented management have a detailed and up-to-date project schedule, and are usually good at directing team members and moving the project forward

The project "**Tasker**" is developed with the purpose of managing our tasks in efficient and automated away. It can become easy and manageable to handle task in a big company with a lot of members and workers. Admin can assign tasks and see progress of his colleagues. While the workers can keep others notified of their work so they work together in a comfortable manner.

INDEX

Task Management Website

Introduction:

Task management is as integral to the project management discipline as a ball to the game of football. Without task management software, there is no project management software.

For this reason, task management is the primary foundation of any project management application. In other words, there could not be a good project management software without a very good task management software as an integral part of it.

Task management is the tool for creating and managing tasks through the project's lifecycle. It includes gathering requirements, planning, status tracking, testing and creating final reports when tasks are completed.

Individuals use task management tools like a pen and a pad or software tools to organize and accomplish personal goals for every day's chores.

Teams rely on task management software to collaborate and achieve group goals together. Tasks could have status, start date, due date, people who are assigned to work on them, comments, tags and files which are attached to them.

More advanced task management systems support dependencies, recurrence, priority, and complexity.

The market is saturated with to-do list and task management tools in varying maturity and functionality.

The hope of this tutorial is to educate the reader about the attributes of good task management software.

We also aim to clarify terms used by project management practitioners when task management is discussed or planned.

There are three levels of task management software, each targeting different project complexities, team's experience, team's structure, and team sizes.

I have grouped these task management tools as beginners, Intermediate, and advance. For each, the required attributes and functionality is listed and explained.

Beginners Task management software

When managing task for personal use or simple projects all the functionality provided by advance task manager is not needed.

Here we have listed the minimum functionality needed to complete basic personal or simple projects.

1. Title: A task should have a title; this helps to state the reason for the task in one simple sentence.
2. Description: Task description is used when the task title is not enough and more information should be given to the person the task is assigned, to make things clear.
3. Start date: When the task starts
4. Due date: When the task ends
5. Assigned to: Individual who is responsible for completing this task
6. That is all needed for a simple task manager to provide. Most to-do lists have this level of functionality. Quite a lot could be accomplished even with is simple task management system. An example if simple project management is Asana.

Intermediate task management software

For more advanced projects when a few people work together, the task manager needs to provide more functionality and features than listed above to be effective. In addition to all features listed above, this type of task manager needs to provide the following:

1. Recurrence: In many projects a task is repeated daily, weekly or monthly. A recurring task helps to simplify this by creating the task once and using it as many times as needed.
2. File attachment: To support task description or files needed to accomplish or verify the task, the task manager should have a facility for attaching files to tasks. Incorporation of third-party file storage systems like Google Drive and Drobox is a big plus.
3. Status: The clear showing of a task's status is very important and can trigger action by the project manager or the team as a whole. A task could have one of the following states:
   - Backlogged
   - Open task
   - Assigned
   - Worked on
   - Late

- On time
- Completed
- Failed
- Forwarded

4. Tagging: A good task manager provides tagging to label task for easier search and grouping.
5. Search: A good task manager provide search by different attributes like assigned to, status and tags.
6. Percentage completed: This provides a tool for the person working on the task to tell the rest of the team what percentage of the task is completed at any given time.
7. Comments: During task's life cycle question may arise. The system should be able to provide the ability for the team members to add comments on task for effective collaboration.

Most online project managers available today provide this level of functionality. This is good enough for most projects which don't need the advanced features we will discuss in the next section.

An example of intermediate project management is Trello

Advance task management software

Complex projects, especially those involving the collaboration of multi-disciplined groups like software engineering, hardware engineering etc, need more advanced project management tools than the previous two types mentioned above.

These projects need robust applications which can manage tasks' journey from inception to completion.

In addition of features mentioned in the above two paragraphs, advanced task management applications have the following elements:

1. Dependencies: In complex projects, we could have tasks that can't start before another task is completed. Or a task can't start unless another task has already started. These requirements are called dependencies or predecessors. The task that is controlling when another task can start is called predecessor. There are four types dependencies used in project management and by far the Finish to Start is the most widely used.
   - Finish to Start- Task A can't start before task B is finished
   - Finish to Finish- Task A can't finish until task B is finished
   - Start to Start- Task A can't start unless task B has started

- Start to Finish- Task B can’t finish unless task A has started

2. Priority: Each task has a priority level from low to high. The project managers set the priority when the task is assigned.
3. Complexity: This is for grouping tasks to simple to very complex. This helps the project team pay extra attention to tasks which are harder to accomplish.
4. Bug Tracking: Advance task management systems have a facility for entering bugs during testing. Tasks could be linked to bugs related to them
5. Gantt chart: Even though Gantt chart was developed in 1950’s, it is still the best method for visualizing tasks in one graph. Advance Gantt charts let the user manipulate task data right from the Gantt.
6. Burn-down chart: The Burn-down chart is another visualization tool for comparing the project plan against the project’s actual progress. It shows if tasks are on time as planned or late. It also predicts the estimated project finish date based on project’s own up to now data.

Advance project management software is used by from simple to multidisciplinary projects which need advance task management plus tracking tools needed to run projects productively. Binfire, Wrike, and Clarizen are examples of project management software which have advanced task management tools.

Concept:

Collaborate, manage projects, and reach new productivity peaks. From high rises to the home office, the way your team works is unique—accomplish it all with Tasker.

Features:

Powering a productive team means using a powerful tool. From meetings and projects to events and goal setting, Tasker’s intuitive features give any team the ability to quickly set up and customize workflows for just about anything.

You and your team can start up a Tasker board in seconds. With the ability to view board data from many different angles, the entire team stays up-to-date in the way that suits them best:

- Use a Timeline view for project planning
- Calendar helps with time management
- Table view connects work across boards
- See board stats with Dashboard, and more!

Lists and cards are the building blocks of organizing work on a Trello board. Grow from there with task assignments, timelines, productivity metrics, calendars, and more.

Spin up a Tasker card with a click, then uncover everything it can hold. Break down bigger card tasks into steps with file attachment previews, reminders, checklists and comments—emoji reactions included! Plus, gain powerful perspective by seeing all cards by list and status at the board level.

Your team can:

- Manage deadlines
- Provide and track feedback
- Assign tasks and hand off work
- Connect work across apps

Tasker features are your portal to more organized work—where every single part of your task can be managed, tracked, and shared with teammates. Open any card to uncover an ecosystem of checklists, due dates, attachments, conversations, and more.

Butler uses natural language commands to automate just about any task in Trello:

- Automate common actions like moving lists
- Create custom buttons to build process quickly
- Surface upcoming deadlines to the team
- Schedule teammate assignments, and more!

HARDWARE AND SOFTWARE REQUIREMENTS

Hardware Requirements:

The minimum Hardware Specification required for the project is given below:

Processor	Intel Pentium or later.
Hard disk	At least 60MB
RAM	1 GB or more.
Display Color	256 Colors
Accessories	Mouse, Keyboard

Software Requirements:

The minimum Software Specification required for the project is given below:

Operating System	Windows XP… or higher.
Front End	ASP.NET using C#
Back End	Microsoft SQL Server
IDE	Visual Studio 2010
Browser	Any browser with JavaScript & CSS 3 support.

Technologies Used

Microsoft Visual Studio:

.NET FRAMEWORK : The **.NET Framework** is a software framework developed by Microsoft that runs primarily on Microsoft_Windows. It includes a large class library called Framework Class Library (FCL) and provides language interoperability (each language can use code written in other languages) across several programming languages. Programs written for .NET Framework execute in a software environment (in contrast to a hardware environment) named the Common Language Runtime (CLR). The CLR is an application virtual machine that provides services such as security, memory management, and exception handling. As such, computer code written using .NET Framework is called "managed code". FCL and CLR together constitute the .NET Framework.

Introduction to ASP.NET

ASP.NET is not just a simple upgrade or the latest version of ASP. ASP.NET combines unprecedented developer productivity with performance, reliability and deployment. ASP.NET redesigns the whole process. It’s still easy to grasp for new comers but it provides many new ways of managing projects.

1. EASY PROGRAMMING MODEL: ASP.NET makes building real world web. Applications dramatically easier.ASP.NET server controls enable an html-like style of declarative programming that let you build great pages with far less code than with classic ASP.

2. FLEXIBLE LANGUAGE OPTIONS: ASP.NET lets you leverage your current Programming language skills.ASP.NET supports more than 25 .NET languages (built in support for VB.NET, C# and JScript.NET), giving you unprecedented flexibility in your choice of language.

3. GREAT TOOL SUPPORT: You can harness the full power of ASP.NET using any text editor, even notepad. VB.NET provides integrated support for debugging and deploying ASP.NET web applications. The enterprise versions of Visual Studio.NET deliver life cycle features to help organizations plan, design, analyze, build, test and coordinate teams that develop ASP.NET web applications.

4. ENHANCED RELIABILITY: ASP.NET ensures that your application is always available to your users.

5. RICH CLASS FRAMEWORK: The .NET framework offers over 4500 classes that encapsulates rich functionality like XML, Data Access, file upload, regular expressions, image generation, performance monitoring and logging, transactions, message queuing, SMTP mail, and much more.

6. COMPILED EXECUTION: ASP.NET is much faster than classic ASP. The “just hit save” update model of ASP. However, no explicit compile detects any changes, dynamically compile step is required. ASP.NET will automatically detect any changes, dynamically compile the files if needed, and store the compiled results to reuse for subsequent requests.

7. RICH OUTPUT CACHING: ASP.NET output caching can dramatically improve the performance and scalability of your application. When output caching is enabled on a page, ASP.NET executes the page just once, and saves the result in memory in addition to sending it to the user. When another user requests the same page, ASP.NET serves the cached result from memory without re- executing the page.

8. EASY DEPLOYMENT: ASP.NET takes the pain of deploying server applications. ASP.NET dramatically simplifies installation of your application. With ASP.NET you can deploy an entire application as easily as an html page.

9. DYNAMIC UPDATE OF RUNNING APPLICATION: ASP.NET lets you update Compiled components; the developer would have to restart the web server each time he deployed an update.

10. MOBILE WEB DEVICE SUPPORT: ASP.NET Mobile controls let you easily target cell phones, PDA’s and over 80 mobile web devices. You write your application just once, and the mobile controls automatically generate WAP/WML, HTML as required by the requesting device.

C# LANGUAGE: C# is a general-purpose, multi-paradigm programming language encompassing static typing, strong typing, lexically scoped, imperative, declarative, functional, generic, object-oriented (class-based), and component-oriented programming disciplines.

C# was developed around 2000 by Microsoft as part of its .NET initiative and later approved as an international standard by Ecma (ECMA-334) in 2002 and ISO (ISO/IEC 23270) in 2003. It was designed by Anders Hejlsberg, and its development team is currently led by Mads Torgersen, being one of the programming languages designed for the Common Language Infrastructure (CLI). The most recent version is 9.0, which was released in 2020 in .NET 5.0 and included in Visual Studio 2019 version 16.8.

HTML: The HyperText Markup Language, or HTML is the standard markup language for documents designed to be displayed in a web browser. It can be assisted by technologies such as Cascading Style Sheets (CSS) and scripting languages such as JavaScript.

Web browsers receive HTML documents from a web server or from local storage and render the documents into multimedia web pages. HTML describes the structure of a web page semantically and originally included cues for the appearance of the document.

CSS: Cascading Style Sheets (CSS) is a style sheet language used for describing the presentation of a document written in a markup language such as HTML. CSS is a cornerstone technology of the World Wide Web, alongside HTML and JavaScript.

CSS is designed to enable the separation of presentation and content, including layout, colors, and fonts. This separation can improve content accessibility, provide more flexibility and control in the specification of presentation characteristics, enable multiple web pages to share formatting by specifying the relevant CSS in a separate .css file which reduces complexity and repetition in the structural content as well as enabling the .css file to be cached to improve the page load speed between the pages that share the file and its formatting.

BACK END:

SQL Database: Structured Query Language) is a domain-specific language used in programming and designed for managing data held in a relational database management system (RDBMS), or for stream processing in a relational data stream management system (RDSMS). It is particularly useful in handling structured data, i.e., data incorporating relations among entities and variables.

System Development Life Cycle

INTRODUCTION

Information system development involves various activities performed together. The stages involved during System Life Cycle are:

- Recognition of need
- Feasibility study
- Analysis
- Design
- Implementation
- Post implementation and maintenance

Recognition of need

It is the first stage of information system development cycle. This gives a clearer picture of what actually the existing system is. The preliminary investigation must define the scope of the project and the perceived problems, opportunities and directives that triggered the project.
The preliminary investigation includes the following tasks:

- List problems, opportunities and directives.
- Negotiate preliminary scope.
- Assess project worth.
- Plan the project.
- Present the project and plan.

Feasibility Study

The statement "don't try to fix it unless you understand it" apply describe the feasibility study stage of system analysis. The goal of feasibility study is to evaluate alternative system and to purpose the most feasible and desirable system for development.

It consists of the following:

- Statement of the problem
- Summary of findings and recommendations

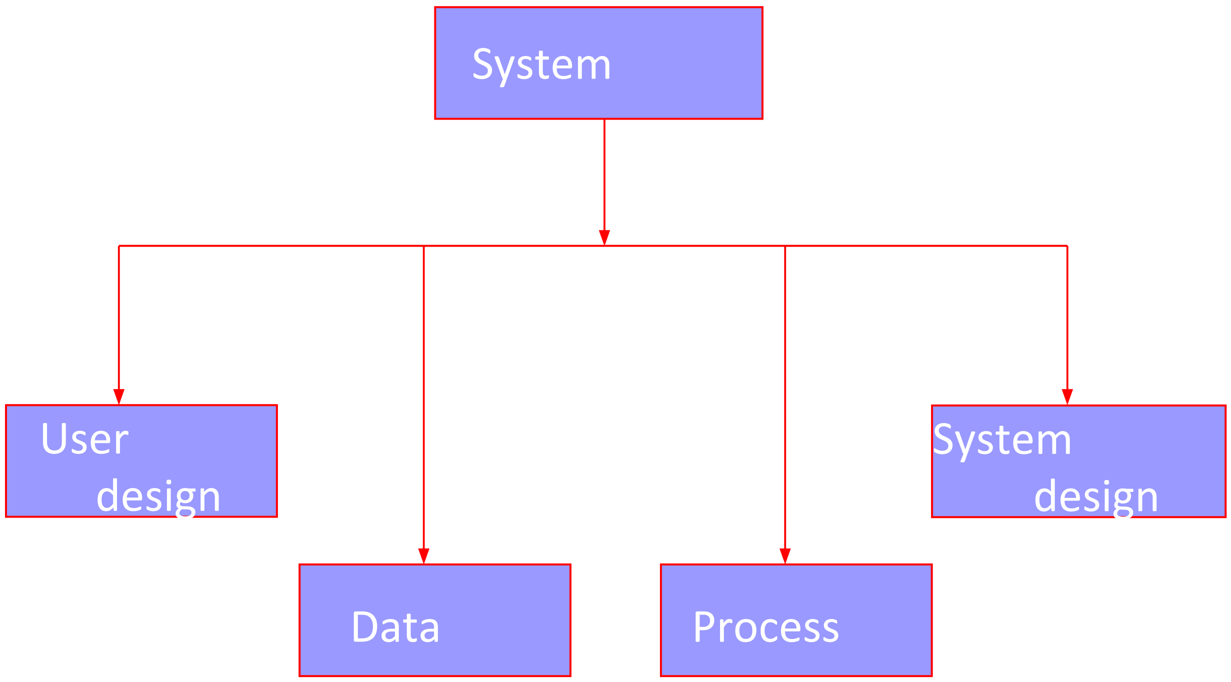
- Details of findings
- Recommendations and conclusions

There are basically five types of feasibility are addressed in the study.

- Technical feasibility
- Economic feasibility
- Motivational feasibility
- Schedule feasibility
- Operational feasibility

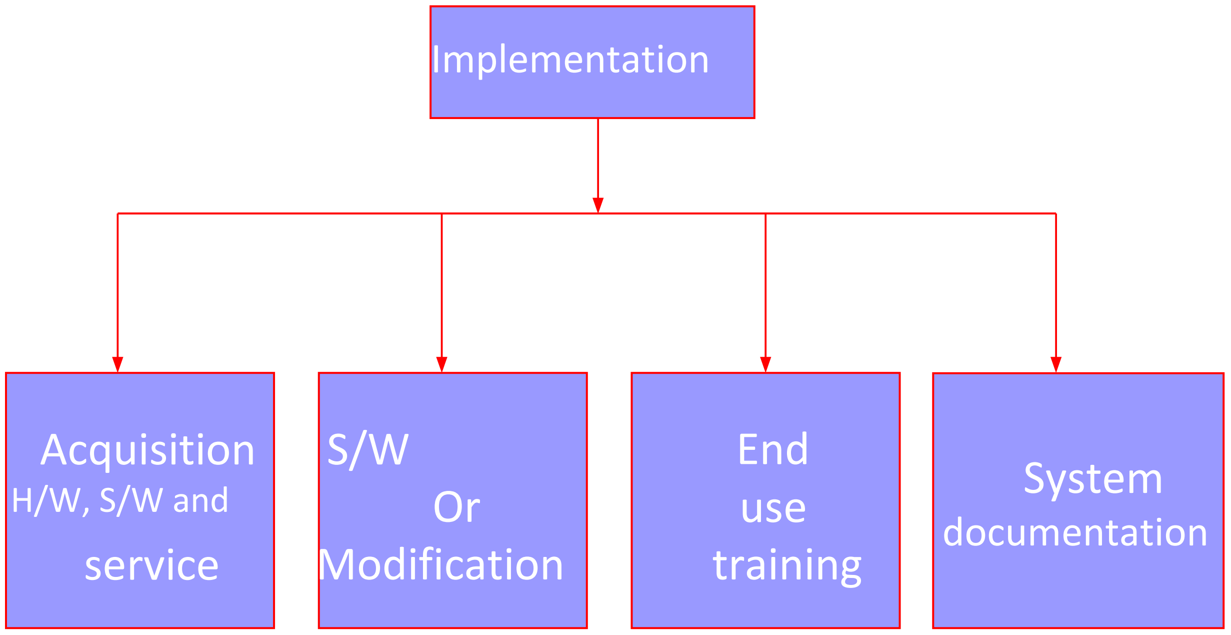
System Design

System design can be viewed as the design of user interface, data, process and system specification.



System Implementation

Implementation is the stage where theory is converted into practical. The implementation is a vital step in ensuring the success of new systems. Even a well-designed system can fail if it is not properly implemented.



Post Implementation and Maintenance

Once a system is fully implemented and being operated by end user, the maintenance function begins. Systems maintenance is the monitoring, evaluating and modifying of operational information system to make desirable or necessary improvements.

Modules and Their Functionalities

- **Master Page:** A master page is an ASP.NET file with the extension .master (with a predefined layout that can include static text, HTML elements, and server controls, Used as a template for the rest of the modules

- **Home Page:** Brief introduction of the platform “Tasker” and a starting point for a Admin and Users to register and use the website.

- **Task List Page:** Displays information about the tasks to all the members of the company. The list of these tasks is assigned be the admin himself. Here the users can see their name of the Task, Task Details, Due Date and they can see their comment if they have added any comment by themselves.

- **My Tasks:** Displays information about the user of the company and a place for the user to check out their task. The user from here can comment about the progress of their task to the admin and the admin will receive that comment. The user will see only their information here.

- **Manage Tasks:** From here the admin can see the list of the task that he/she has provided to and will see all the details as well. The Admin can manage tasks here. The Admin can delete the tasks from here as per the requirement depending upon the tasks.

- **Tasks:** With this Module the Admin of the company can provide tasks to various employees of the company. The Admin can put the correct name of the user and put task, task details and due date from here.

- **Sign Up:** Users can sign up to create their profiles and to see their tasks and comment on their progress.

- **Log In:** Users can login once they have created their profile through sign up page.

- **User Profiles:** Users can see their details in this module and they can update and save as they like.

- **Forgot Password:** If user forgets their password, they change their password from here.

- **Admin Sign Up**: The admin can create a profile to manage the users in the website.

- **Admin Login:** Admin can login his/her profile from here.

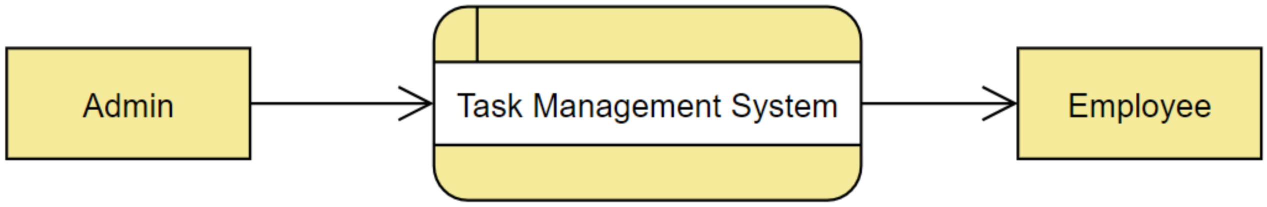
- **Grid View:** Admin can see the list of users from here and delete the user as per requirement.

Data Flow Diagram

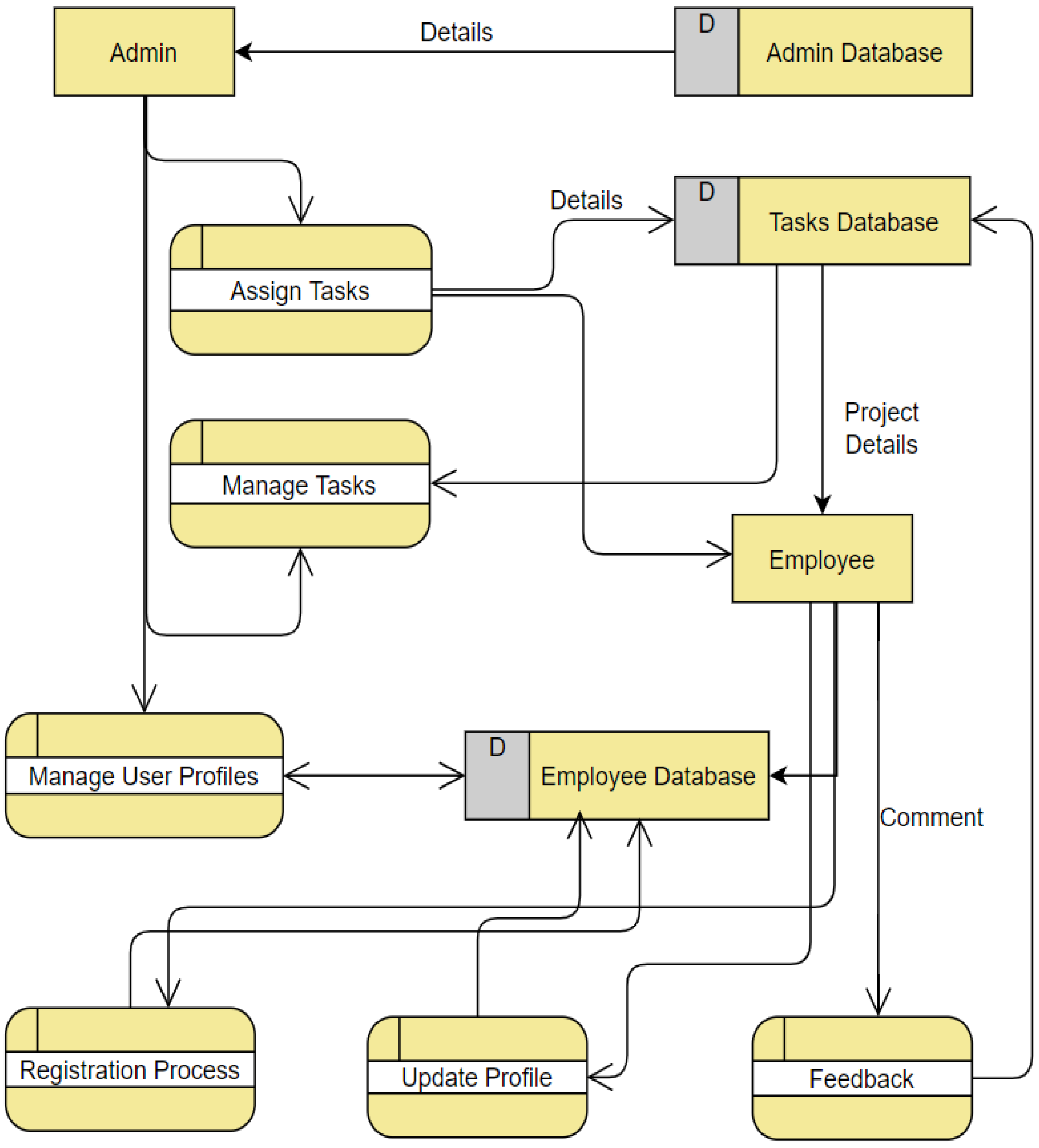
Overview

Data flow diagram (DFD) represents the flows of data between different processes in a business. It is a graphical technique that depicts information flow and the transforms that are applied as data move form input to output. It provides a simple, intuitive method for describing business processes without focusing on the details of computer systems. DFDs are attractive technique because they provide what users do rather than what computers do.

DFD 0 LEVEL



DFD 1 LEVEL

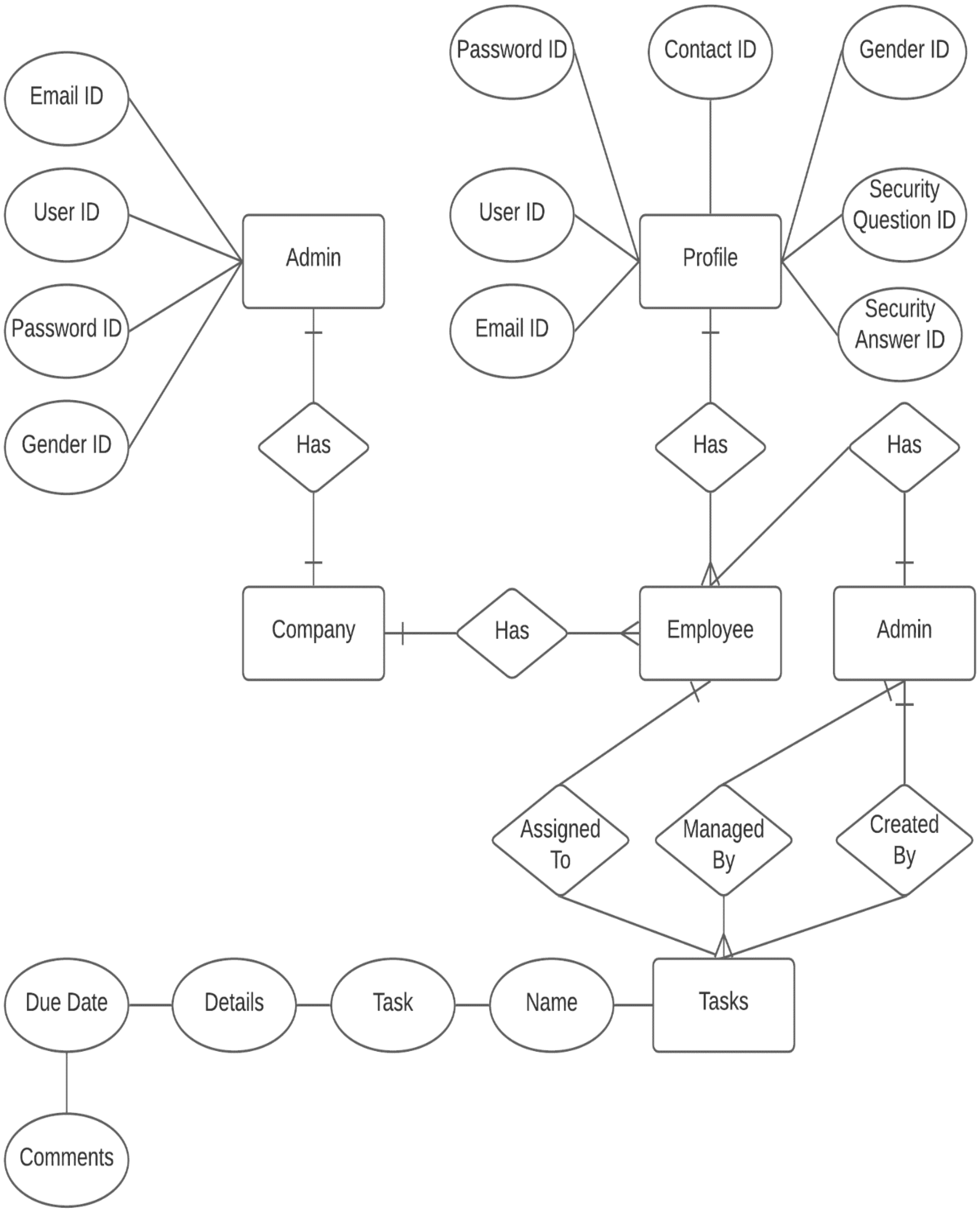

Admin
Details
D
Admin Database
Details
D
Tasks Database
Assign Tasks
Manage Tasks
Project Details
Employee
Manage User Profiles
D
Employee Database
Comment
Registration Process
Update Profile
Feedback

Entity–Relationship Model

An **Entity–relationship model (ER model)** describes the structure of a database with the help of a diagram, which is known as **Entity Relationship Diagram (ER Diagram)**. An ER model is a design or blueprint of a database that can later be implemented as a database. The main components of E-R model are: entity set and relationship set.

An ER diagram shows the relationship among entity sets. An entity set is a group of similar entities and these entities can have attributes. In terms of DBMS, an entity is a table or attribute of a table in database, so by showing relationship among tables and their attributes, ER diagram shows the complete logical structure of a database.

ER Model of Tasker

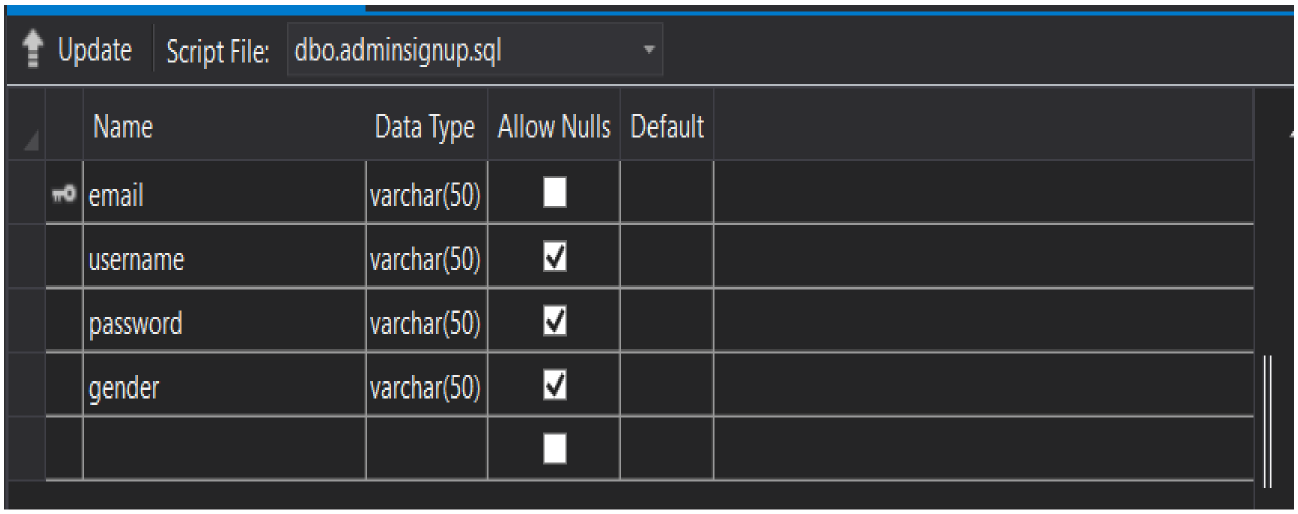


Database Design

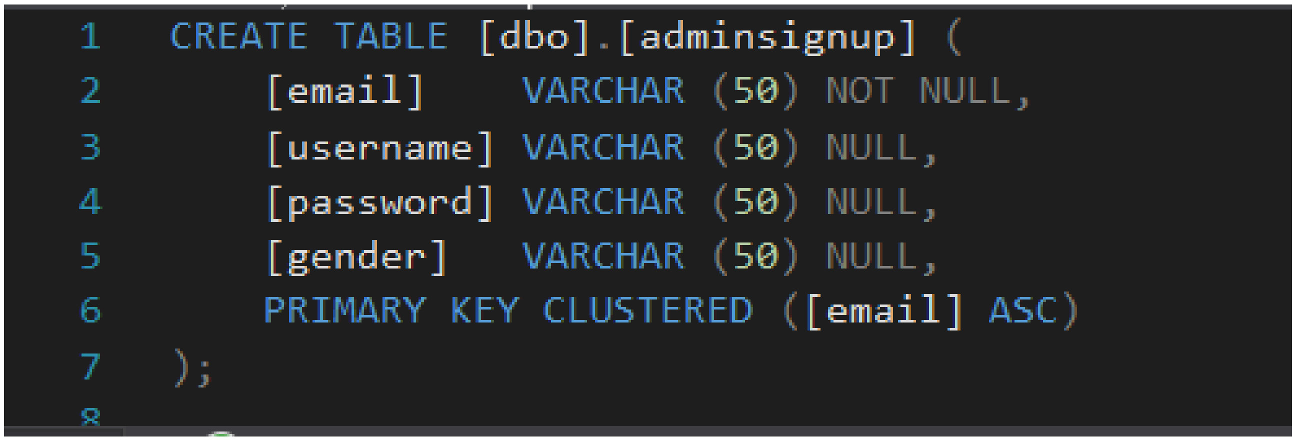
In this phase system and software design is prepared from the requirement specification which was studied in the first phase. System design helps in specifying hardware and system requirements and also helps in defining overall system architecture. This system design specification serves as input of the next phase of the model.

Admin Data:

In Admin database, data regarding admin is stored here. This includes when Admin registers or make profile using Sign Up module. In this email is the primary key. All the data is fetched through email. Admin register using information as email username password and gender and use these for log in. Admin sign up and log in are different modules than user or employee.



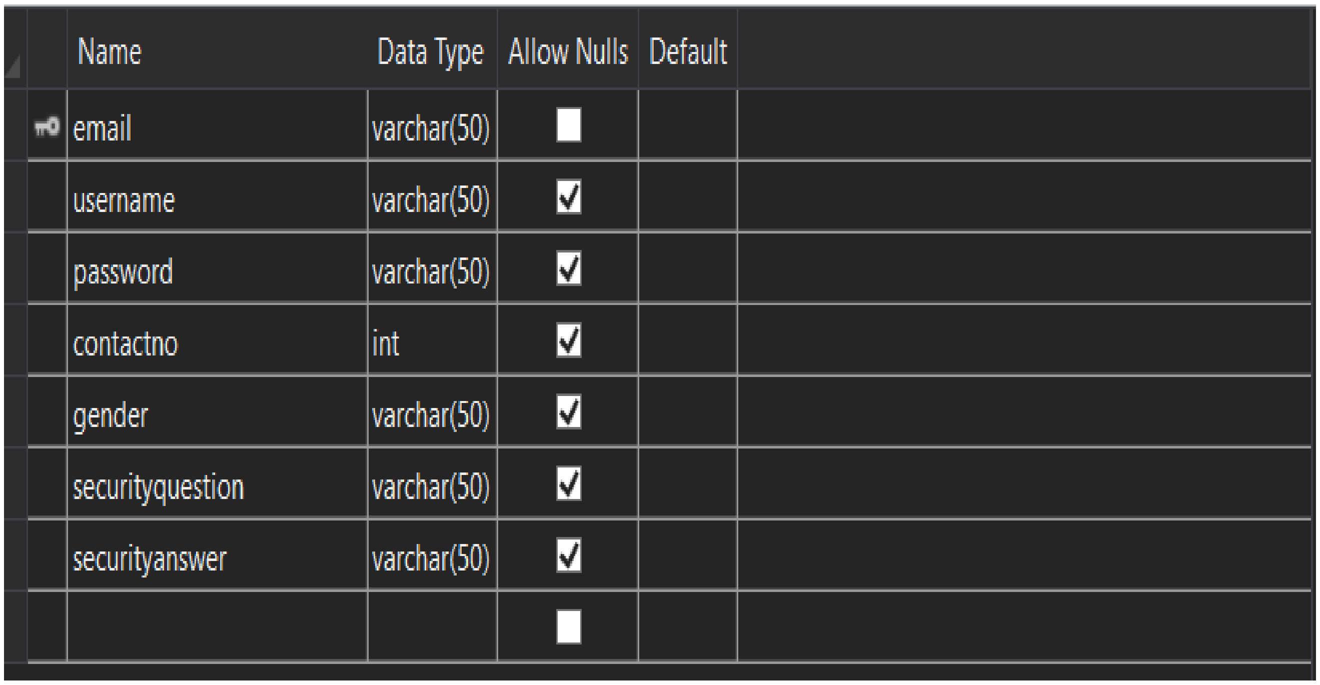
Name	Data Type	Allow Nulls	Default
email	varchar(50)	☐	
username	varchar(50)	✓	
password	varchar(50)	✓	
gender	varchar(50)	✓	
		☐	



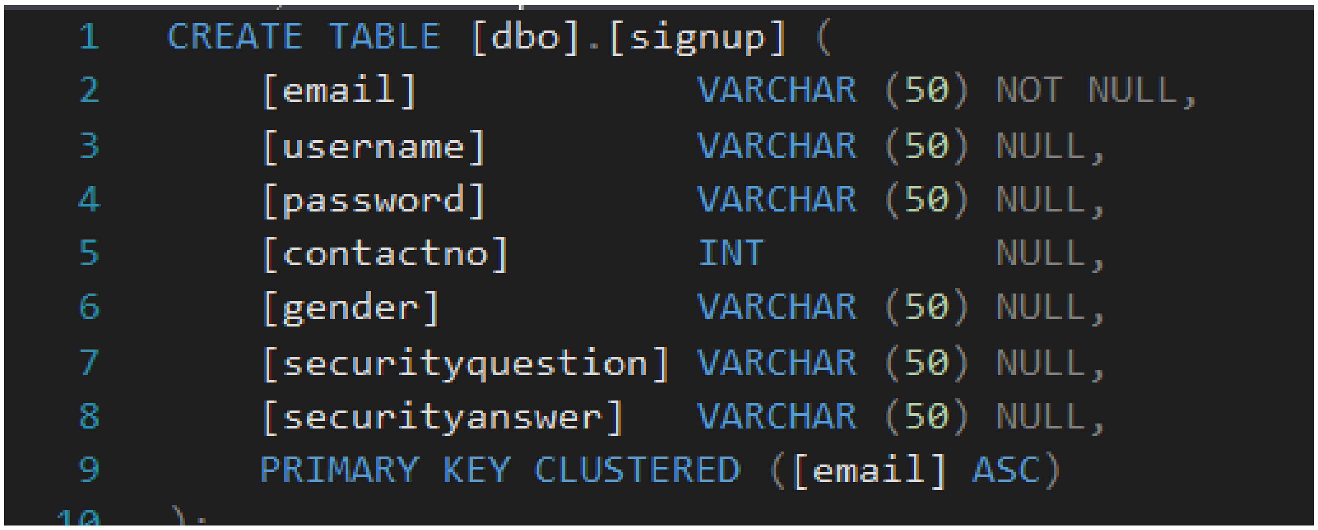
```
CREATE TABLE [dbo].[adminsignup] (
    [email]    VARCHAR (50) NOT NULL,
    [username] VARCHAR (50) NULL,
    [password] VARCHAR (50) NULL,
    [gender]   VARCHAR (50) NULL,
    PRIMARY KEY CLUSTERED ([email] ASC)
);
```

User Data:

This is user or employee database. Here information regarding user registration stores in. As same as the admin database, in this email is also the primary key and is used to fetched data if required. Using Grid View, the information was fetched through the email information of the user. This database is used for user/employee sign up and log in.



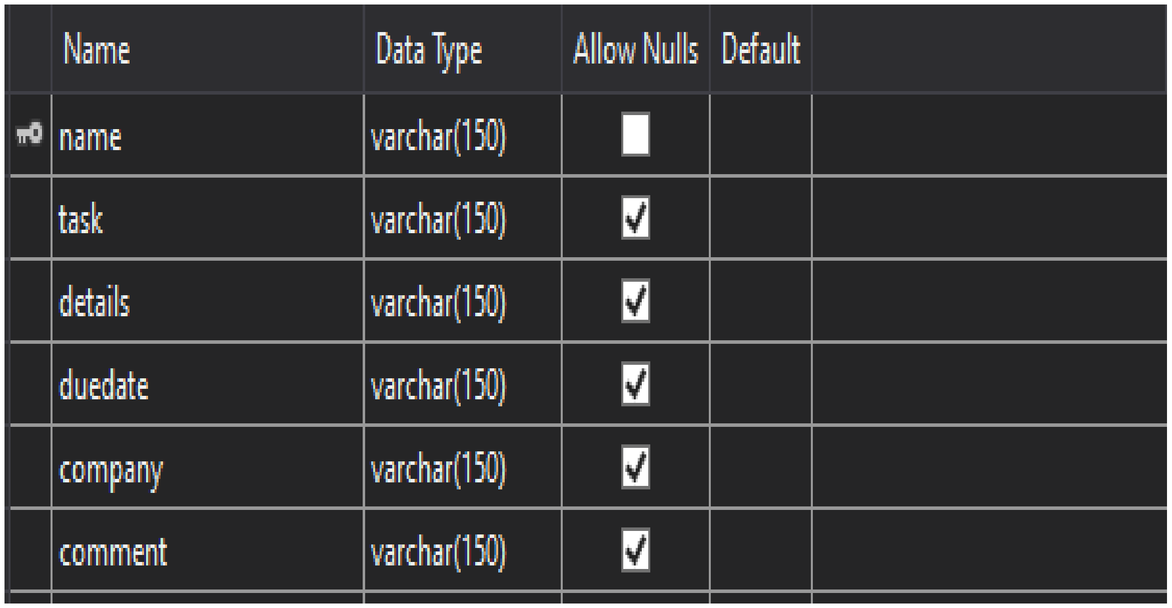
Name	Data Type	Allow Nulls	Default
email	varchar(50)	☐	
username	varchar(50)	✓	
password	varchar(50)	✓	
contactno	int	✓	
gender	varchar(50)	✓	
securityquestion	varchar(50)	✓	
securityanswer	varchar(50)	✓	
		☐	



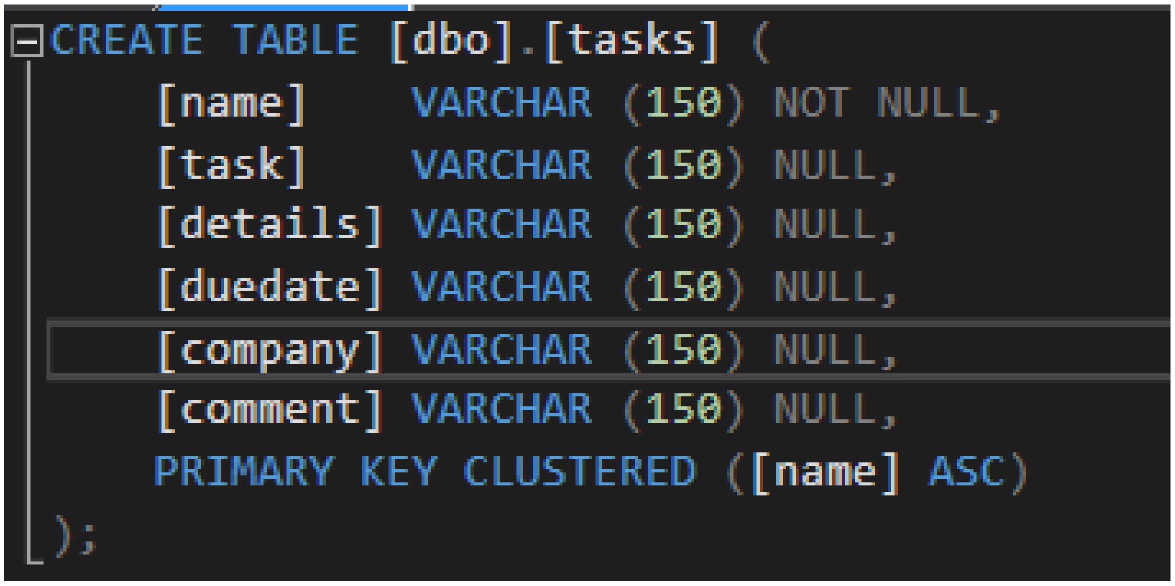
```
CREATE TABLE [dbo].[signup] (
    [email]            VARCHAR (50) NOT NULL,
    [username]         VARCHAR (50) NULL,
    [password]         VARCHAR (50) NULL,
    [contactno]        INT          NULL,
    [gender]           VARCHAR (50) NULL,
    [securityquestion] VARCHAR (50) NULL,
    [securityanswer]   VARCHAR (50) NULL,
    PRIMARY KEY CLUSTERED ([email] ASC)
);
```

Tasks Data:

In Tasks database, information regarding tasks is stored in this database. The admin assigns and manages the task and data is stored in this database. This information is again is shown to the user or employee through this database.



Name	Data Type	Allow Nulls	Default
name	varchar(150)	☐	
task	varchar(150)	✓	
details	varchar(150)	✓	
duedate	varchar(150)	✓	
company	varchar(150)	✓	
comment	varchar(150)	✓	



```
CREATE TABLE [dbo].[tasks] (
    [name]    VARCHAR (150) NOT NULL,
    [task]    VARCHAR (150) NULL,
    [details] VARCHAR (150) NULL,
    [duedate] VARCHAR (150) NULL,
    [company] VARCHAR (150) NULL,
    [comment] VARCHAR (150) NULL,
    PRIMARY KEY CLUSTERED ([name] ASC)
);
```

Code of Modules

Master Page (HTML):

```
<%@ Master Language="C#" AutoEventWireup="true" CodeFile="MasterPage.master.cs"
Inherits="MasterPage" %>
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<title></title>
<meta name="keywords" content="" />
<meta name="description" content="" />
<link
href="http://fonts.googleapis.com/css?family=Source+Sans+Pro:200,300,400,600,700,900"
rel="stylesheet" />
<link href="default.css" rel="stylesheet" type="text/css" media="all" />
<link href="fonts.css" rel="stylesheet" type="text/css" media="all" />
    <asp:ContentPlaceHolder id="head" runat="server">
    </asp:ContentPlaceHolder>
</head>
<body class="back-img">
    <form id="form1" runat="server"><div></div>
<div class="header">
                     <div class="inner-header">
                            <div class="logo-container">
                                   <h1>T<span>asker</span></h1>
                            </div>
                            <ul class="navigation">
                                   <a href="HomePage.aspx"><li>Home</li></a>
                                   <a href="AdminLogin.aspx"><li>Admin</li></a>
                                   <a><li>Company</li></a>
                                   <a><li>Members</li></a>
                                   <a href="SignUp.aspx"><li>SignUp</li></a>
                                   <a href="login.aspx"><li>Login</li></a</ul></div></div>
<asp:ContentPlaceHolder id="ContentPlaceHolder1" runat="server">
        </asp:ContentPlaceHolder>
<div id="copyright">
       <ul class="contact">
              <li><a href="#" class="icon icon-twitter"><span>Twitter</span></a></li>
              <li><a href="#" class="icon icon-facebook"><span></span></a></li>
              <li><a href="#" class="icon icon-dribbble"><span>Pinterest</span></a></li>
              <li><a href="#" class="icon icon-tumblr"><span>Google+</span></a></li>
              <li><a href="#" class="icon icon-rss"><span>Pinterest</span></a></li></ul>
       <p>© Untitled. All rights reserved. | </p>
</div> </form></body></html>
```

Sign Up Page (C#):

```
using System.Data;
using System.Data.SqlClient;
using System.Web.Configuration;
```

```
public partial class _Default : System.Web.UI.Page
{
    string strcon = null;
    int result = 0;
    protected void Button1_Click(object sender, EventArgs e){
        strcon =
WebConfigurationManager.ConnectionStrings["ConnectionString"].ConnectionString.ToString()
;
        string gen;
        SqlConnection sqlcon = new SqlConnection(strcon);
        sqlcon.Open();
        if(RadioButton1.Checked)
        {
            gen = RadioButton1.Text;
        }
        else
        {
            gen = RadioButton2.Text;
        }
        SqlCommand sqlcmd = new SqlCommand("insert into
signup(username,email,password,gender,securityquestion,securityanswer,contactno)
values('"+TextBox1.Text+"','"+TextBox3.Text+"','"+TextBox2.Text+"','" + gen +
"','"+DropDownList1.Text+ "','" + TextBox6.Text + "','"+TextBox5.Text+"')", sqlcon);
        result = sqlcmd.ExecuteNonQuery();
        if(result > 0)
        {
            Response.Write("insertion");
            Response.Redirect("login.aspx");
        }
            else
        {
            Response.Write("error");
        }
        sqlcon.Close();
    }}
```

Login (C#):

```
using System.Data.SqlClient;
using System.Configuration;
using System.Data;
public partial class Default2 : System.Web.UI.Page
{
    SqlCommand cmd = new SqlCommand();
    SqlDataAdapter sda = new SqlDataAdapter();
    DataSet ds = new DataSet();
    protected void Button1_Click(object sender, EventArgs e)
    {
        SqlConnection con = new
SqlConnection(ConfigurationManager.ConnectionStrings["ConnectionString"].ConnectionString
);
        con.Open();
        SqlCommand cmd = new SqlCommand("select * from signup where username=@username
and password=@password", con);
        cmd.Parameters.AddWithValue("@username", TextBox1.Text);
        cmd.Parameters.AddWithValue("@password", TextBox2.Text);
        SqlDataAdapter da = new SqlDataAdapter(cmd);
        DataTable dt = new DataTable();
        da.Fill(dt);
        if (dt.Rows.Count > 0)
        {}
        else
        {
            Label1.Visible = true;
            Literal1.Text = "Wrong Details";

        }
        foreach (DataRow ROW in dt.Rows)
        {
            Session["email"] = ROW["email"];
            Session["username"] = ROW["username"];
            Response.Redirect("userprofile.aspx");
        }
        Response.Write("wrong password");
        con.Close();
    }
}
```

Forgot Password (C#):

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Data.SqlClient;
using System.Web.Configuration;
public partial class _Default : System.Web.UI.Page
{
    string strcon =
WebConfigurationManager.ConnectionStrings["ConnectionString"].ConnectionString.ToString()
;
    protected void Button1_Click(object sender, EventArgs e)
    {
        int result = 0;
        SqlConnection sqlcon = new SqlConnection(strcon);
        sqlcon.Open();
        SqlCommand sqlcmd = new SqlCommand("update signup set
password='"+TextBox2.Text+"' where email='"+TextBox1.Text+"'",sqlcon);
        result = sqlcmd.ExecuteNonQuery();
        if (result > 0)
{
            Response.Write("Password is Updated");
            Response.Redirect("SignIn.aspx");
        }
        else
        {
            Response.Write("Password is not Updated.");
        }
        sqlcon.Close();
    }
}
```

User Profile Page (C#):

```
public partial class Default2 : System.Web.UI.Page
{
    string strcon = 
WebConfigurationManager.ConnectionStrings["ConnectionString"].ConnectionString.ToString()
;
    string emailid;

    protected void Page_Load(object sender, EventArgs e)
    {
        emailid = Session["email"].ToString();
        Literal1.Text = Session["username"].ToString();
        if (!Page.IsPostBack)
        {
            SqlConnection sqlcon = new SqlConnection(strcon);
            sqlcon.Open();
            SqlCommand sqlcmd = new SqlCommand("select * from signup where email='" + 
emailid + "'", sqlcon);
            sqlcmd.ExecuteNonQuery();
            DataTable table = new DataTable();
            SqlDataAdapter adt = new SqlDataAdapter(sqlcmd);
            adt.Fill(table);
            foreach(DataRow ROW in table.Rows)
            {
                TextBox1.Text = ROW["email"].ToString();
                TextBox2.Text = ROW["username"].ToString();
                TextBox3.Text = ROW["gender"].ToString();
            }
            sqlcon.Close();
        } }
    protected void Button1_Click(object sender, EventArgs e)
    {
        TextBox1.Enabled = true;
        TextBox2.Enabled = true;
        TextBox3.Enabled = true;
    }
    protected void Button2_Click(object sender, EventArgs e)
    {
        int result = 0;
        SqlConnection sqlcon = new SqlConnection(strcon);
        sqlcon.Open();
        SqlCommand sqlcmd = new SqlCommand("update signin set username ='" + 
TextBox1.Text + "',email='" + TextBox2.Text + "',gender='" + TextBox3.Text + "' where 
email='"+emailid+"'", sqlcon);
        result = sqlcmd.ExecuteNonQuery();
        if(result>0)
        {
            Response.Write("updated");
        }
        else
        {
            Response.Write("not update");
        }
        sqlcon.Close();
    }
    protected void Button3_Click(object sender, EventArgs e)
    {
        Session["email"] = null;
        Response.Redirect("login.aspx");
    }}
```

Admin Sign Up (C#):

```
public partial class Default2 : System.Web.UI.Page
{
    string strcon =
WebConfigurationManager.ConnectionStrings["ConnectionString"].ConnectionString.ToString()
;
    protected void Button1_Click(object sender, EventArgs e)
    {
        int result = 0;
        string gen;
        SqlConnection sqlcon = new SqlConnection(strcon);
        sqlcon.Open();
        if (RadioButton1.Checked)
        {
            gen = RadioButton1.Text;}
        else
        {
            gen = RadioButton2.Text;
        }
        SqlCommand sqlcmd = new SqlCommand("insert into
adminsignup(email,username,password,gender) values('" + TextBox1.Text + "','" +
TextBox2.Text + "','" + TextBox3.Text + "','" + gen + "')", sqlcon);
        result = sqlcmd.ExecuteNonQuery();
        if (result > 0)
        {
            Response.Write("inserted");
            Response.Redirect("AdminLogin.aspx");
        }

        else
        {
            Response.Write("not inserted");
        }
        sqlcon.Close();}}


public partial class _Default : System.Web.UI.Page
{
    string strcon =
WebConfigurationManager.ConnectionStrings["ConnectionString"].ConnectionString.ToString()
;
    protected void Page_Load(object sender, EventArgs e)
    {

    }

    protected void Button1_Click(object sender, EventArgs e)
    {
        SqlConnection sqlcon = new SqlConnection(strcon);
        sqlcon.Open();
        SqlCommand sqlcmd = new SqlCommand("select * from adminsignup where email='" +
TextBox1.Text + "' and password='" + TextBox2.Text + "'", sqlcon);
```

```
        sqlcmd.ExecuteNonQuery();
        DataTable table = new DataTable();
        SqlDataAdapter adt = new SqlDataAdapter(sqlcmd);
        adt.Fill(table);
        foreach (DataRow ROW in table.Rows)
        {
            Session["email"] = ROW["email"];//add this line to login page
            Response.Redirect("GridView.aspx");
        }
        Response.Write("wrong details");
        sqlcon.Close();

    }
}
```

Grid View (HTML):

```
<%@ Page Title="" Language="C#" MasterPageFile="~/MasterPage.master"
AutoEventWireup="true" CodeFile="GridView.aspx.cs" Inherits="_Default" %>

<asp:Content ID="Content1" ContentPlaceHolderID="head" Runat="Server">
</asp:Content>
<asp:Content ID="Content2" ContentPlaceHolderID="ContentPlaceHolder1" Runat="Server">
    <div class="back">
    <asp:GridView ID="GridView1" runat="server" AutoGenerateColumns="False"
DataKeyNames="email">
        <Columns>
            <asp:TemplateField HeaderText="email" SortExpression="email">
                <EditItemTemplate>
                    <asp:Label ID="Label1" runat="server" Text='<%# Eval("email")
%>'></asp:Label>
                </EditItemTemplate>
                <ItemTemplate>
                    <asp:Label ID="Label1" runat="server" Text='<%# Bind("email")
%>'></asp:Label>
                </ItemTemplate>
            </asp:TemplateField>
            <asp:BoundField DataField="username" HeaderText="username"
SortExpression="username" />
            <asp:BoundField DataField="password" HeaderText="password"
SortExpression="password" />
            <asp:BoundField DataField="contactno" HeaderText="contactno"
SortExpression="contactno" />
            <asp:BoundField DataField="gender" HeaderText="gender"
SortExpression="gender" />
            <asp:BoundField DataField="securityquestion" HeaderText="securityquestion"
SortExpression="securityquestion" />
            <asp:BoundField DataField="securityanswer" HeaderText="securityanswer"
SortExpression="securityanswer" />
            <asp:TemplateField HeaderText="Select">
                <EditItemTemplate>
                    <asp:CheckBox ID="CheckBox1" runat="server" />
                </EditItemTemplate>
                <ItemTemplate>
                    <asp:CheckBox ID="CheckBox1" runat="server" />
                </ItemTemplate>
            </asp:TemplateField>
        </Columns>
    </asp:GridView>
    <br />
    <br />
    <asp:Button ID="Button1" CssClass="button-custom" runat="server" Text="Delete"
OnClick="Button1_Click" />
    </div>
</asp:Content>
```

Grid View (C#):

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Data;
using System.Data.SqlClient;
using System.Web.Configuration;


public partial class _Default : System.Web.UI.Page
{
    string strcon =
WebConfigurationManager.ConnectionStrings["ConnectionString"].ConnectionString.ToString()
;
    protected void Page_Load(object sender, EventArgs e)
    {
        if (!Page.IsPostBack)
        {
            refreshdata();
        }

    }
    public void refreshdata()
    {

        SqlConnection sqlcon = new SqlConnection(strcon);
        SqlCommand cmd = new SqlCommand("select * from signup", sqlcon);
        SqlDataAdapter sda = new SqlDataAdapter(cmd);
        DataTable dt = new DataTable();
        sda.Fill(dt);
        GridView1.DataSource = dt;
        GridView1.DataBind();

    }


    protected void Button1_Click(object sender, EventArgs e)
    {
        SqlConnection sqlcon = new SqlConnection(strcon);
        foreach (GridViewRow gvrow in GridView1.Rows)
        {

            CheckBox chck = gvrow.FindControl("CheckBox1") as CheckBox;
            if (chck.Checked)
            {
                var Label = gvrow.FindControl("Label1") as Label;

                SqlCommand cmd = new SqlCommand("delete from signup where email=@email", 
sqlcon);
                cmd.Parameters.AddWithValue("email", Label.Text);
                sqlcon.Open();
                int id = cmd.ExecuteNonQuery();
                sqlcon.Close();

            }}}}
```

My Task (C#):

```
public partial class _Default : System.Web.UI.Page
{
    string strcon = 
WebConfigurationManager.ConnectionStrings["ConnectionString"].ConnectionString.ToString()
;
    string name;
    string nameid;
    protected void Page_Load(object sender, EventArgs e)
    {
        name = Session["username"].ToString();
        Label3.Text = Session["username"].ToString();
        if (!Page.IsPostBack)
        {
            SqlConnection sqlcon = new SqlConnection(strcon);
            sqlcon.Open();
            SqlCommand sqlcmd = new SqlCommand("select * from tasks where name='" + name 
+ "'", sqlcon);
            sqlcmd.ExecuteNonQuery();
            DataTable table = new DataTable();
            SqlDataAdapter adt = new SqlDataAdapter(sqlcmd);
            adt.Fill(table);
            foreach (DataRow ROW in table.Rows)
            {

                    Label2.Text = ROW["task"].ToString();
                    Label1.Text = ROW["details"].ToString();
                    Label4.Text = ROW["duedate"].ToString();

            }
            sqlcon.Close();
        }}
    protected void Button1_Click(object sender, EventArgs e)
    {
        int result = 0;
        SqlConnection sqlcon = new SqlConnection(strcon);
        sqlcon.Open();
        SqlCommand sqlcmd = new SqlCommand("update tasks set comment ='" + TextBox1.Text 
+ "' where name='" + name + "'", sqlcon);
        result = sqlcmd.ExecuteNonQuery();
        if (result > 0)
        {
            Response.Write("updated");
        }
        else
        {
            Response.Write("not update");
        }
        sqlcon.Close();
    }}
```

Manage Tasks(C#):

```
using System;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Data;
using System.Data.SqlClient;
using System.Web.Configuration;

public partial class _Default : System.Web.UI.Page
{
    string strcon = 
WebConfigurationManager.ConnectionStrings["ConnectionString"].ConnectionString.ToString()
;
    protected void Page_Load(object sender, EventArgs e)
    {
        if (!Page.IsPostBack)
        {
            refreshdata();
        }}
    public void refreshdata()
    {

        SqlConnection sqlcon = new SqlConnection(strcon);
        SqlCommand cmd = new SqlCommand("select * from tasks", sqlcon);
        SqlDataAdapter sda = new SqlDataAdapter(cmd);
        DataTable dt = new DataTable();
        sda.Fill(dt);
        GridView1.DataSource = dt;
        GridView1.DataBind();
    }
    protected void Button1_Click(object sender, EventArgs e)
    {
        SqlConnection sqlcon = new SqlConnection(strcon);
        foreach (GridViewRow gvrow in GridView1.Rows)
        {

            CheckBox chck = gvrow.FindControl("CheckBox1") as CheckBox;
            if (chck.Checked)
            {
                var Label = gvrow.FindControl("Label1") as Label;

                SqlCommand cmd = new SqlCommand("delete from tasks where name=@name", 
sqlcon);
                cmd.Parameters.AddWithValue("name", Label.Text);
                sqlcon.Open();
                int id = cmd.ExecuteNonQuery();
                sqlcon.Close();
            }}}}
```

Tasks List(HTML):

```
<%@ Page Title="" Language="C#" MasterPageFile="~/MasterPage.master"
AutoEventWireup="true" CodeFile="Task List.aspx.cs" Inherits="_Default" %>

<asp:Content ID="Content1" ContentPlaceHolderID="head" Runat="Server">
</asp:Content>
<asp:Content ID="Content2" ContentPlaceHolderID="ContentPlaceHolder1" Runat="Server">
    <div class="back">
    <asp:ListView ID="ListView1" runat="server" DataKeyNames="name"
DataSourceID="SqlDataSource1">
        <AlternatingItemTemplate>
            <li style="background-color: #FFFFFF;color: #284775;">name:
                <asp:Label ID="nameLabel" runat="server" Text='<%# Eval("name") %>' />
                <br />
                task:
                <asp:Label ID="taskLabel" runat="server" Text='<%# Eval("task") %>' />
                <br />
                details:
                <asp:Label ID="detailsLabel" runat="server" Text='<%# Eval("details") %>'
/>
                <br />
                duedate:
                <asp:Label ID="duedateLabel" runat="server" Text='<%# Eval("duedate") %>'
/>
                <br />
                company:
                <asp:Label ID="companyLabel" runat="server" Text='<%# Eval("company") %>'
/>
                <br />
                comment:
                <asp:Label ID="commentLabel" runat="server" Text='<%# Eval("comment") %>'
/>
                <br />
            </li>
        </AlternatingItemTemplate>
        <EditItemTemplate>
            <li style="background-color: #999999;">name:
                <asp:Label ID="nameLabel1" runat="server" Text='<%# Eval("name") %>' />
                <br />
                task:
                <asp:TextBox ID="taskTextBox" runat="server" Text='<%# Bind("task") %>'
/>
                <br />
                details:
                <asp:TextBox ID="detailsTextBox" runat="server" Text='<%# Bind("details")
%>' />
                <br />
                duedate:
                <asp:TextBox ID="duedateTextBox" runat="server" Text='<%# Bind("duedate")
%>' />
                <br />
                company:
                <asp:TextBox ID="companyTextBox" runat="server" Text='<%# Bind("company")
%>' />
                <br />
                comment:
```

```
                <asp:TextBox ID="commentTextBox" runat="server" Text='<%# Bind("comment")
%>' />
                <br />
                <asp:Button ID="UpdateButton" runat="server" CommandName="Update"
Text="Update" />
                <asp:Button ID="CancelButton" runat="server" CommandName="Cancel"
Text="Cancel" />
            </li>
        </EditItemTemplate>
        <EmptyDataTemplate>
            No data was returned.
        </EmptyDataTemplate>
        <InsertItemTemplate>
            <li style="">name:
                <asp:TextBox ID="nameTextBox" runat="server" Text='<%# Bind("name") %>'
/>
                <br />
                task:
                <asp:TextBox ID="taskTextBox" runat="server" Text='<%# Bind("task") %>'
/>
                <br />
                details:
                <asp:TextBox ID="detailsTextBox" runat="server" Text='<%# Bind("details")
%>' />
                <br />
                duedate:
                <asp:TextBox ID="duedateTextBox" runat="server" Text='<%# Bind("duedate")
%>' />
                <br />
                company:
                <asp:TextBox ID="companyTextBox" runat="server" Text='<%# Bind("company")
%>' />
                <br />
                comment:
                <asp:TextBox ID="commentTextBox" runat="server" Text='<%# Bind("comment")
%>' />
                <br />
                <asp:Button ID="InsertButton" runat="server" CommandName="Insert"
Text="Insert" />
                <asp:Button ID="CancelButton" runat="server" CommandName="Cancel"
Text="Clear" />
            </li>
        </InsertItemTemplate>
        <ItemSeparatorTemplate>
            <br />
        </ItemSeparatorTemplate>
        <ItemTemplate>
            <li style="background-color: #E0FFFF;color: #333333;">name:
                <asp:Label ID="nameLabel" runat="server" Text='<%# Eval("name") %>' />
                <br />
                task:
                <asp:Label ID="taskLabel" runat="server" Text='<%# Eval("task") %>' />
                <br />
                details:
                <asp:Label ID="detailsLabel" runat="server" Text='<%# Eval("details") %>'
/>
                <br />
                duedate:
```

```
                <asp:Label ID="duedateLabel" runat="server" Text='<%# Eval("duedate") %>'
/>
                <br />
                company:
                <asp:Label ID="companyLabel" runat="server" Text='<%# Eval("company") %>'
/>
                <br />
                comment:
                <asp:Label ID="commentLabel" runat="server" Text='<%# Eval("comment") %>'
/>
                <br />
            </li>
        </ItemTemplate>
        <LayoutTemplate>
            <ul id="itemPlaceholderContainer" runat="server" style="font-family: Verdana,
Arial, Helvetica, sans-serif;">
                <li runat="server" id="itemPlaceholder" />
            </ul>
            <div style="text-align: center;background-color: #5D7B9D; font-family:
Verdana, Arial, Helvetica, sans-serif;color: #FFFFFF;">
            </div>
        </LayoutTemplate>
        <SelectedItemTemplate>
            <li style="background-color: #E2DED6;font-weight: bold;color: #333333;">name:
             <asp:Label ID="nameLabel" runat="server" Text='<%# Eval("name") %>' />
             task:
             <asp:Label ID="taskLabel" runat="server" Text='<%# Eval("task") %>' />
                details:
             <asp:Label ID="detailsLabel" runat="server" Text='<%# Eval("details") %>' />
                duedate:
             <asp:Label ID="duedateLabel" runat="server" Text='<%# Eval("duedate") %>' />
                company:
             <asp:Label ID="companyLabel" runat="server" Text='<%# Eval("company") %>' />
                comment:
             <asp:Label ID="commentLabel" runat="server" Text='<%# Eval("comment") %>' />
            </li>
        </SelectedItemTemplate>
    </asp:ListView>
    <br />
    <asp:SqlDataSource ID="SqlDataSource1" runat="server" ConnectionString="<%$
ConnectionStrings:ConnectionString %>" SelectCommand="SELECT * FROM
[tasks]"></asp:SqlDataSource>
    <br />
    </div>
</asp:Content>
```

Task(C#):

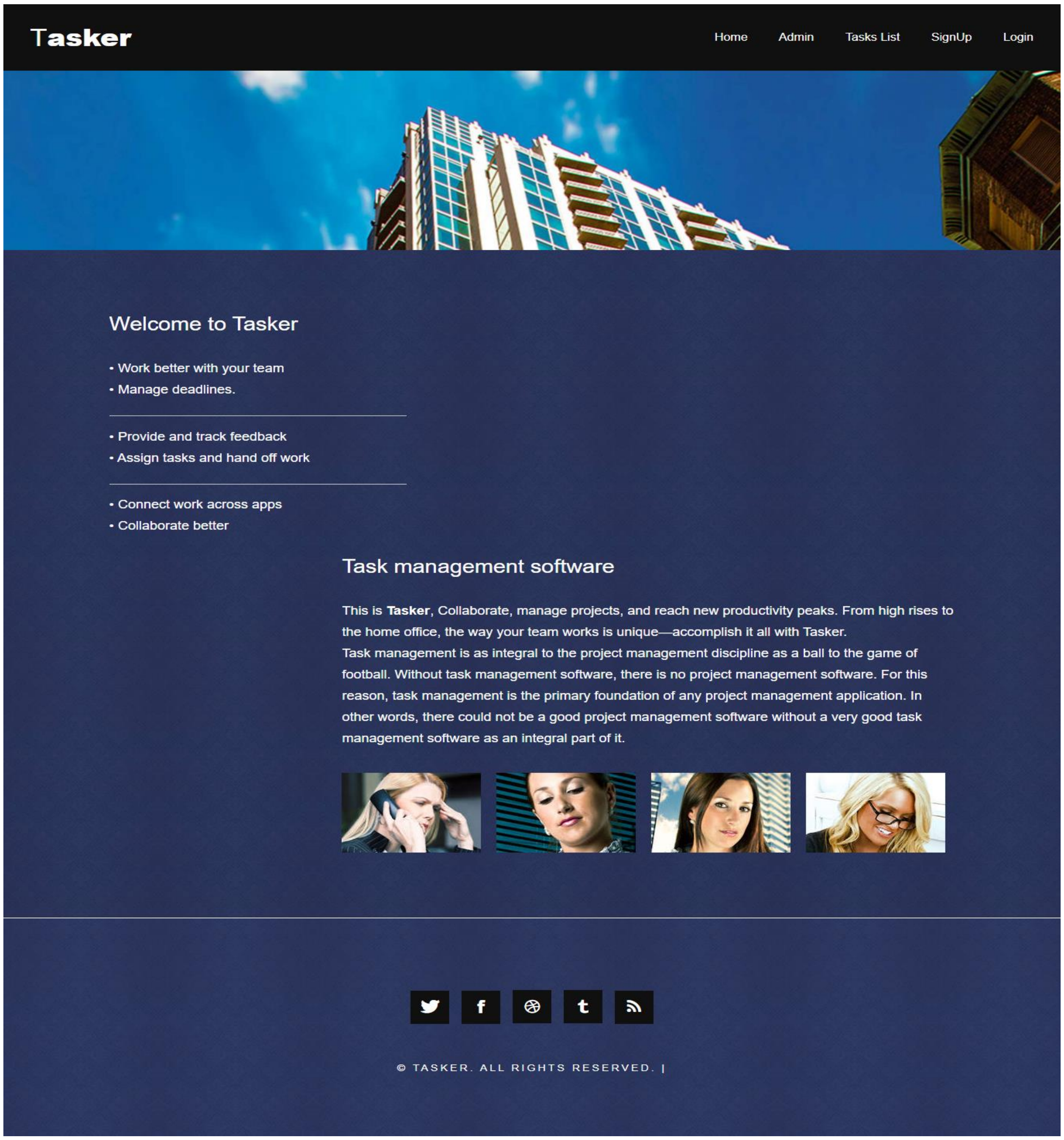
```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Data;
using System.Data.SqlClient;
using System.Web.Configuration;
```

```
public partial class _Default : System.Web.UI.Page
{
    string strcon = null;
    int result = 0;
    protected void Page_Load(object sender, EventArgs e)
    {

    }

    protected void Button1_Click(object sender, EventArgs e)
    {
        strcon =
WebConfigurationManager.ConnectionStrings["ConnectionString"].ConnectionString.ToString()
;
        string gen;
        SqlConnection sqlcon = new SqlConnection(strcon);
        sqlcon.Open();
        SqlCommand sqlcmd = new SqlCommand("insert into
tasks(name,company,task,details,duedate) values('" + TextBox1.Text + "','" +
TextBox5.Text + "','" + TextBox2.Text + "','" + TextBox3.Text + "','" + TextBox4.Text +
"')", sqlcon);
        result = sqlcmd.ExecuteNonQuery();
        if (result > 0)
        {
            Response.Write("insertion");
            Response.Redirect("Task List.aspx");
        }
        else
        {
            Response.Write("error");
        }
        sqlcon.Close();
    }
}
```

Screenshots of Tasker

Home Page:

The homepage of Tasker shows information regarding the function of the website. This is the first page that the visitor will encounter.



Welcome to Tasker

• Work better with your team
• Manage deadlines.

• Provide and track feedback
• Assign tasks and hand off work

• Connect work across apps
• Collaborate better

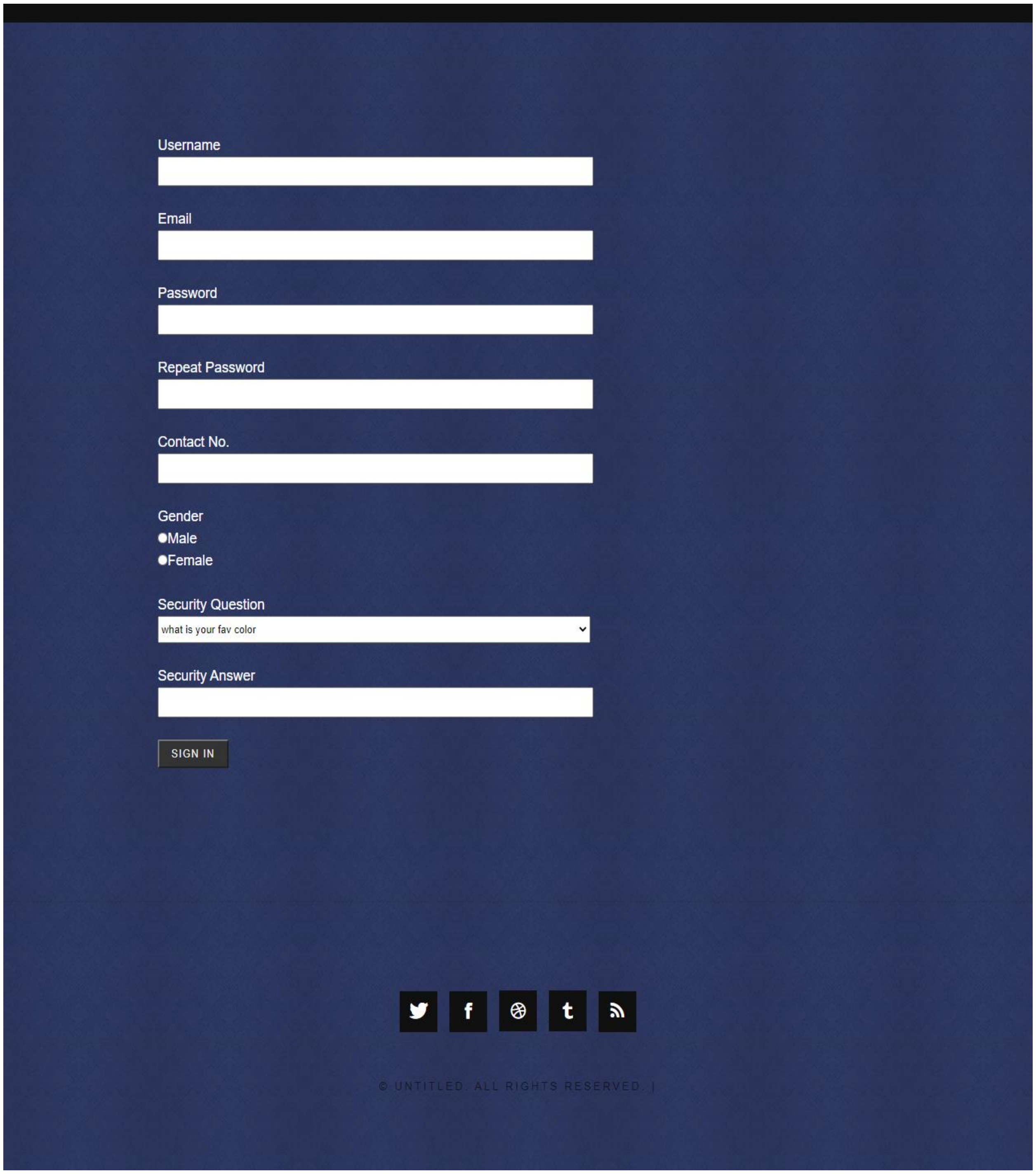
Task management software

This is **Tasker**, Collaborate, manage projects, and reach new productivity peaks. From high rises to the home office, the way your team works is unique—accomplish it all with Tasker.
Task management is as integral to the project management discipline as a ball to the game of football. Without task management software, there is no project management software. For this reason, task management is the primary foundation of any project management application. In other words, there could not be a good project management software without a very good task management software as an integral part of it.

© TASKER. ALL RIGHTS RESERVED. |

Sign Up Page:

In the Sign-Up page, the user will register here or make their user profile.



Username

Email

Password

Repeat Password

Contact No.

Gender

●Male

●Female

Security Question

what is your fav color

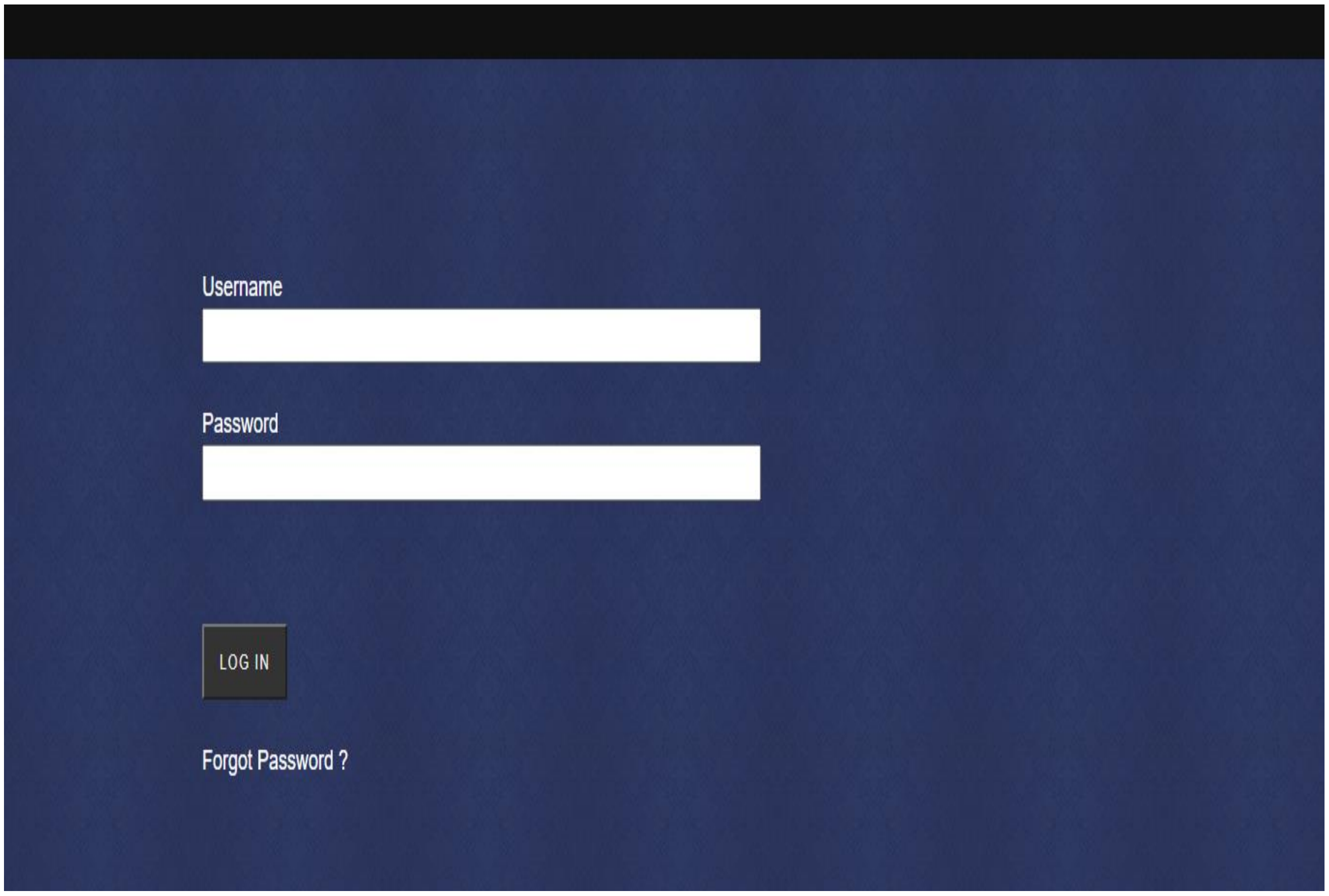
Security Answer

SIGN IN

© UNTITLED. ALL RIGHTS RESERVED. |

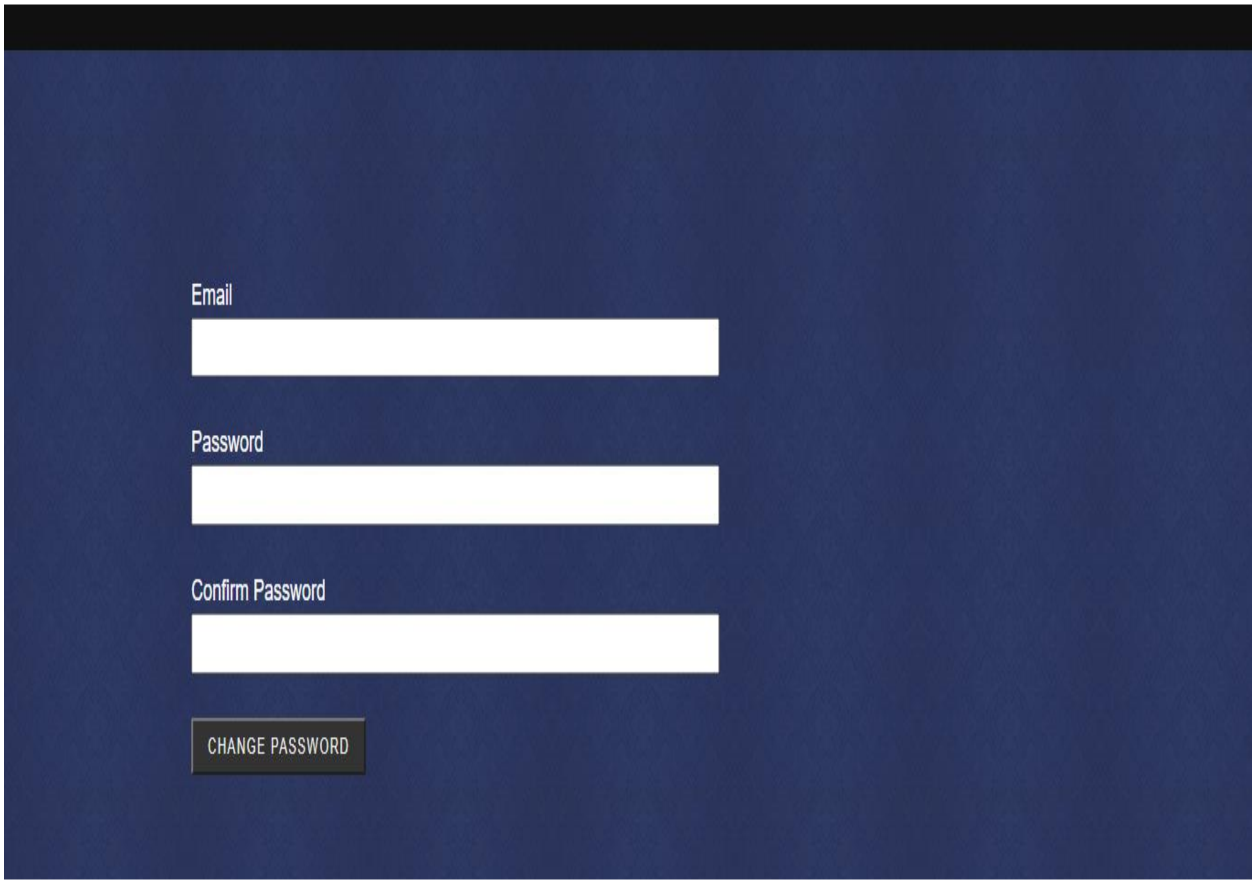
Login Page:

In the Login Page, the user uses the information as username and password to login. This information is retrieved from the database.



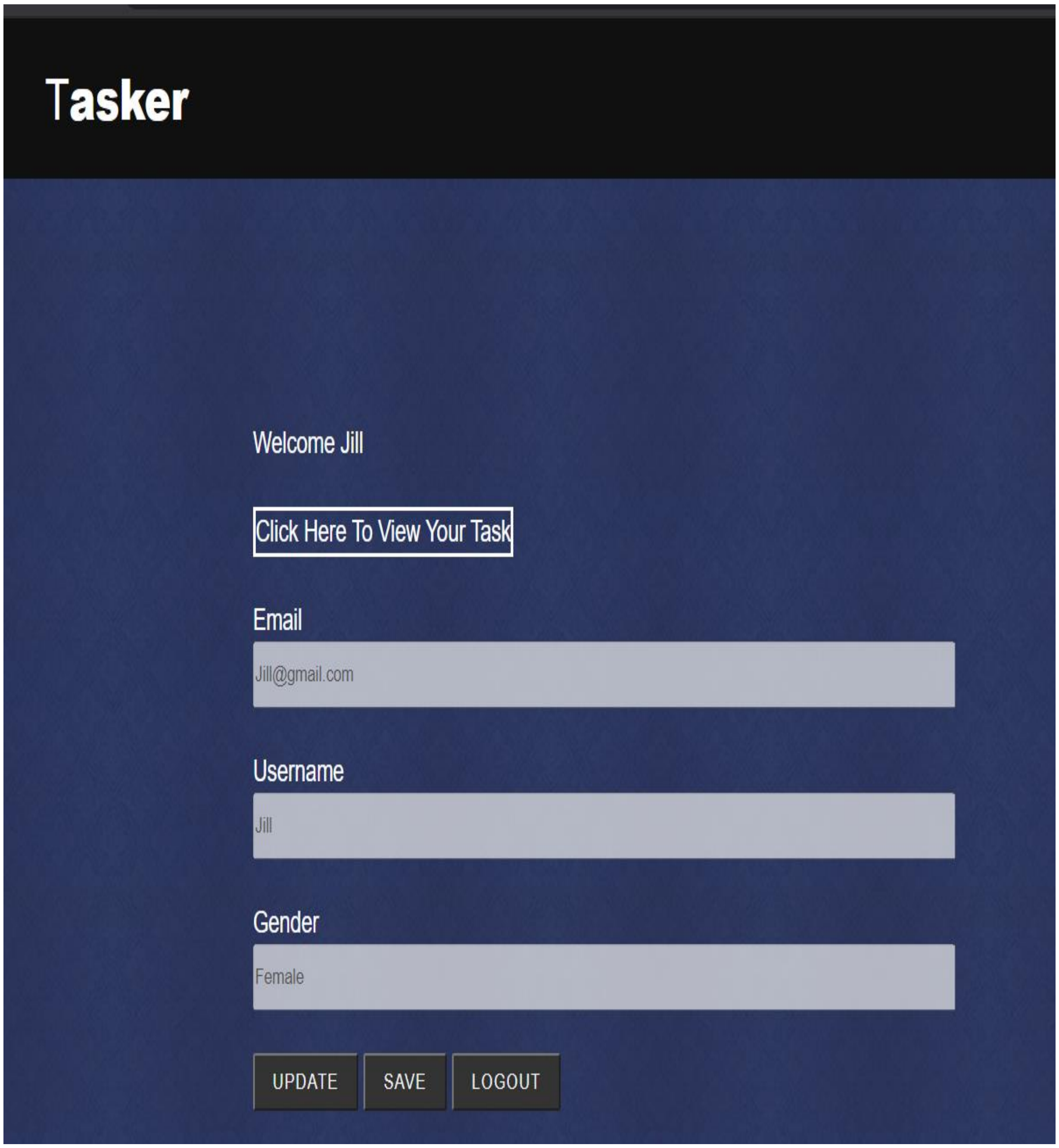
Forgot Password Page:

In this page, if the user forgets their password. They can visit this page and change their password. All they need to do is to put their username here and they can change their password.



User Profile Page:

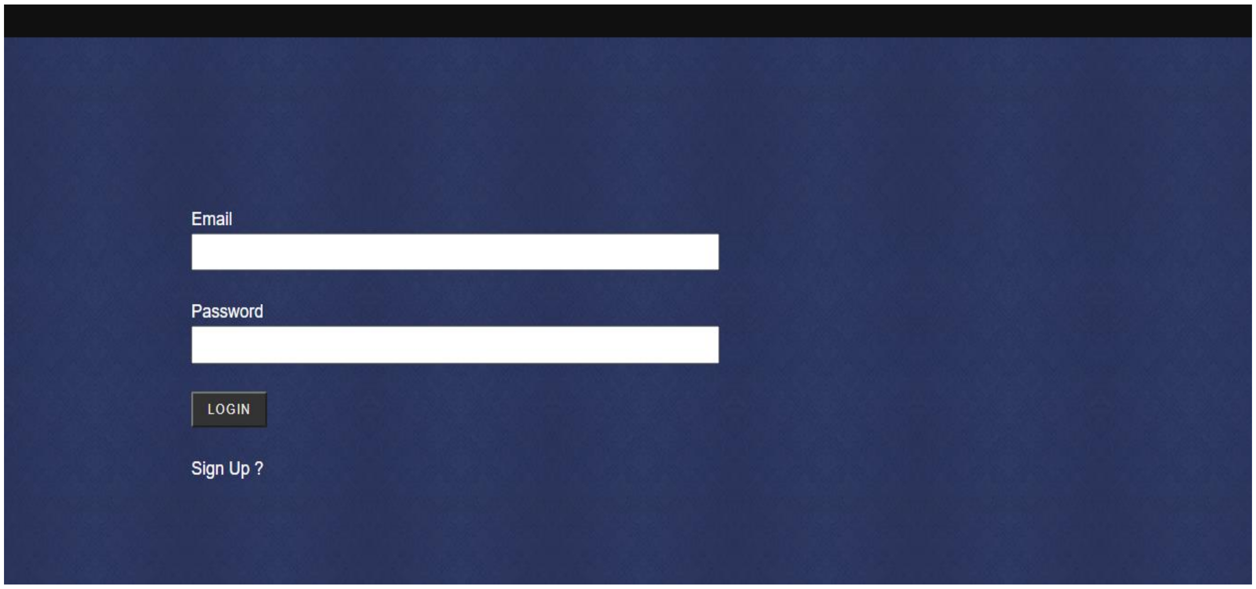
After the user logs in their account, the user profile is the first page they will see. Here they will see their information of their account. If the user wants to change something i.e. Email, Username or Gender, they can do that from here. The textbox is not accessible until the user clicks on the update button. After this with the updated information the user has to click save. The new information will be updated in the database.



Email
Jill@gmail.com
Username
Jill
Gender
Female

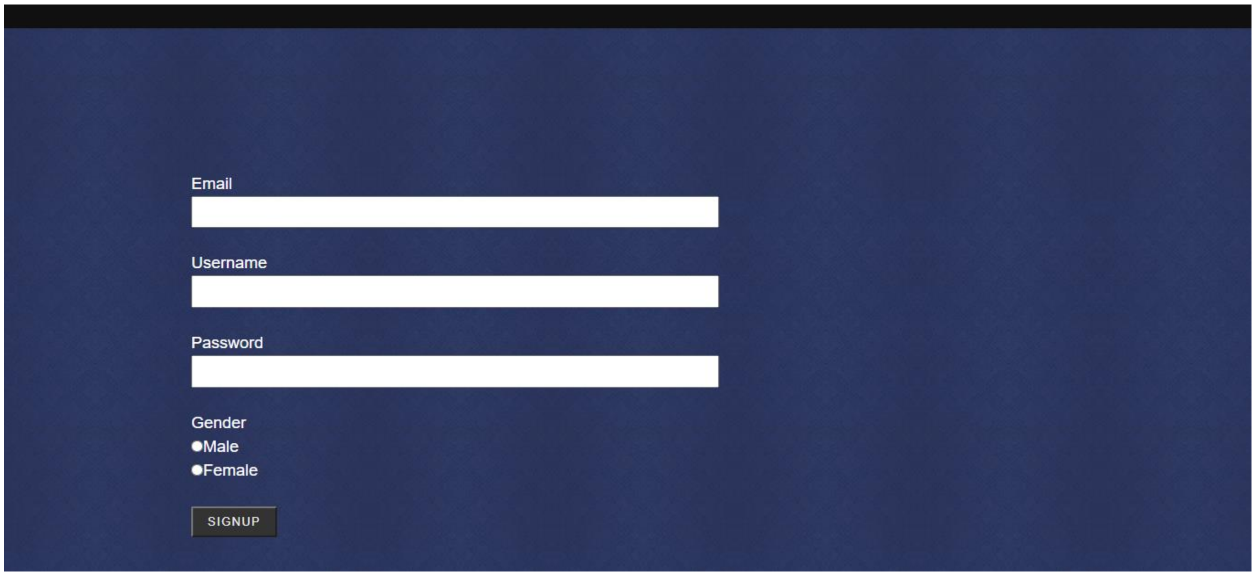
Admin Login Page:

Page where Admin can log in.



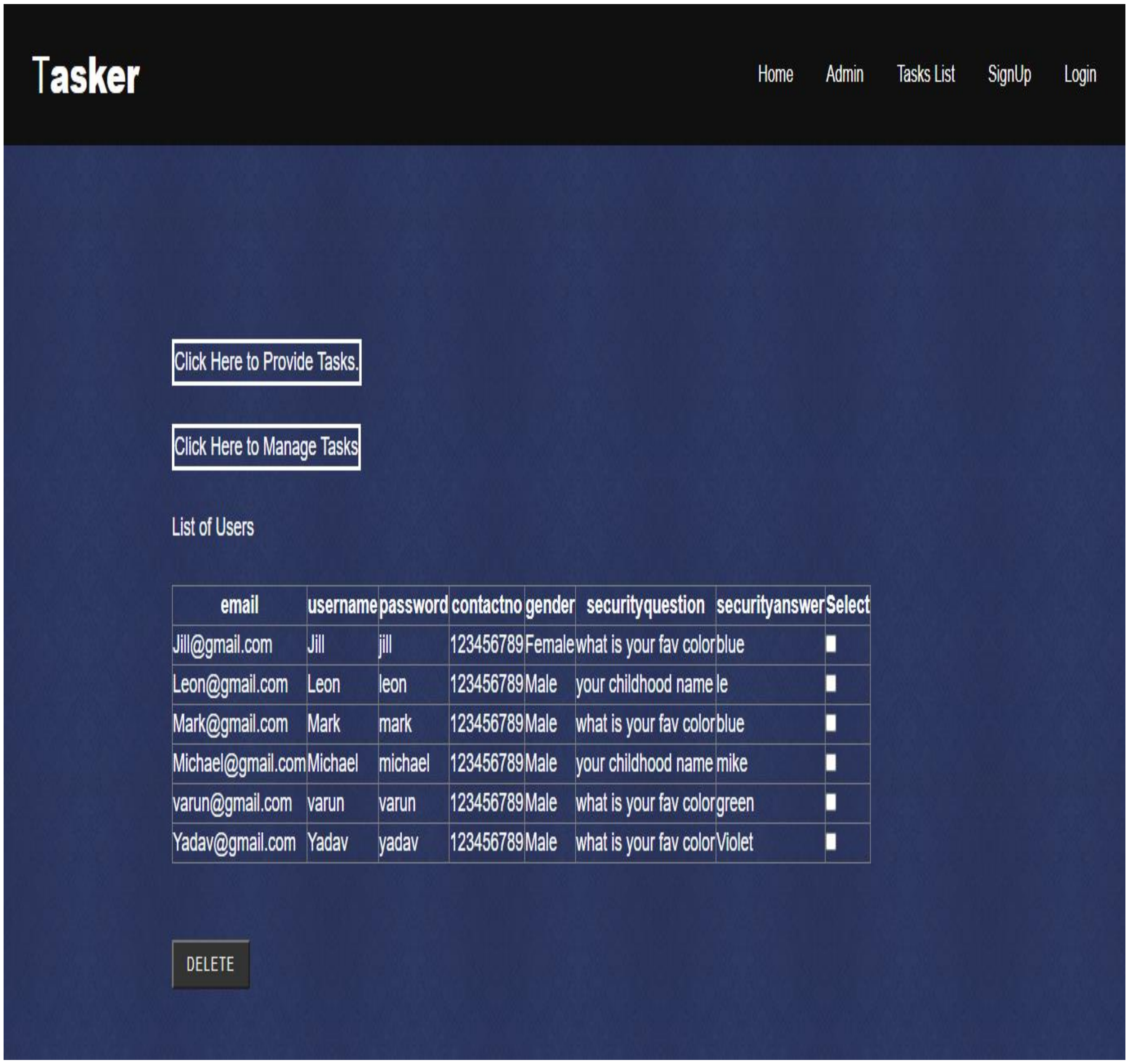
Admin Sign Up Page:

Page where Admin can Sign up and create an account.



Admin Grid View Page:

Here Admin can manage users by deleting them if required. Admin can click on two Link Buttons either to provide tasks or manage tasks.

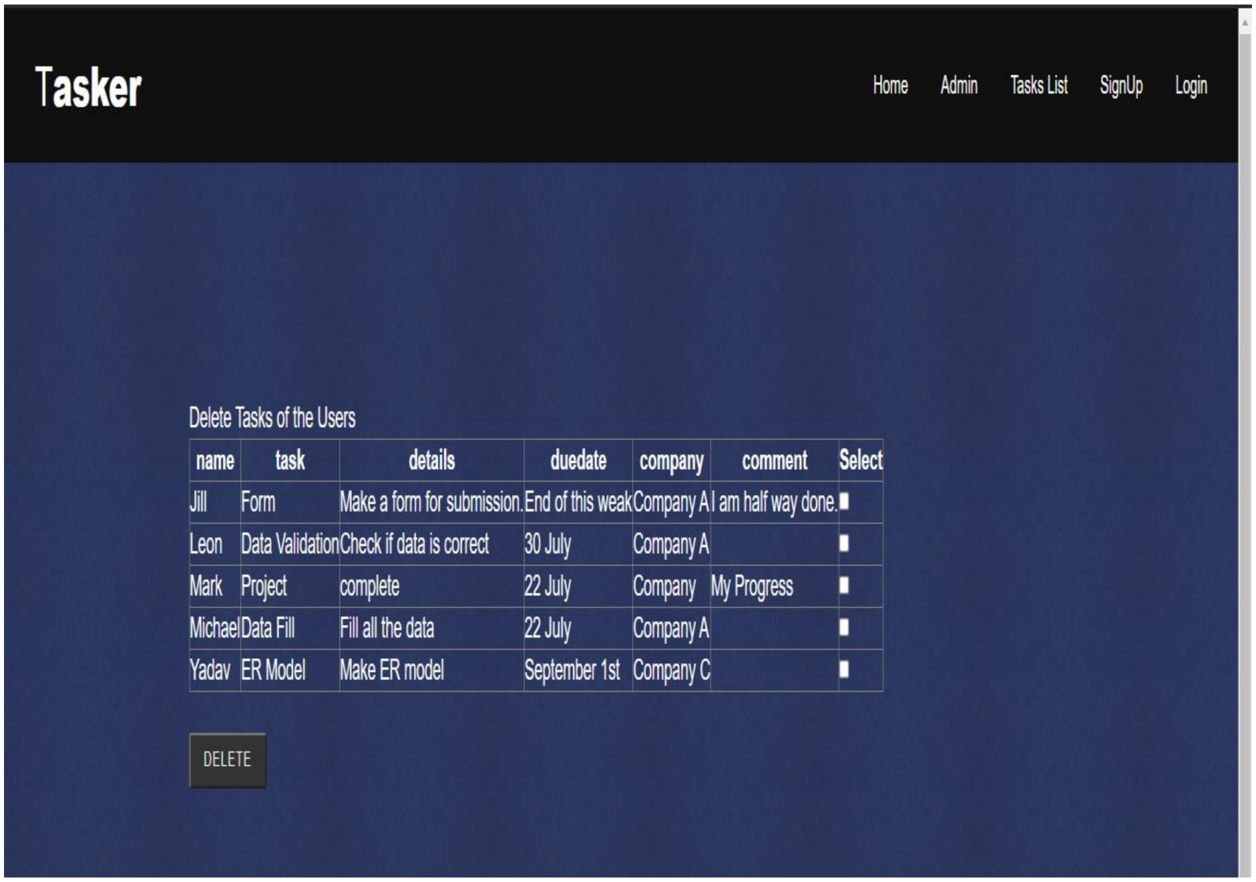


List of Users

email	username	password	contactno	gender	securityquestion	securityanswer	Select
Jill@gmail.com	Jill	jill	123456789	Female	what is your fav color	blue	☐
Leon@gmail.com	Leon	leon	123456789	Male	your childhood name	le	☐
Mark@gmail.com	Mark	mark	123456789	Male	what is your fav color	blue	☐
Michael@gmail.com	Michael	michael	123456789	Male	your childhood name	mike	☐
varun@gmail.com	varun	varun	123456789	Male	what is your fav color	green	☐
Yadav@gmail.com	Yadav	yadav	123456789	Male	what is your fav color	Violet	☐

Manage Task Page:

Here admin can manage tasks. Admin can delete tasks by deleting them as per requirement.



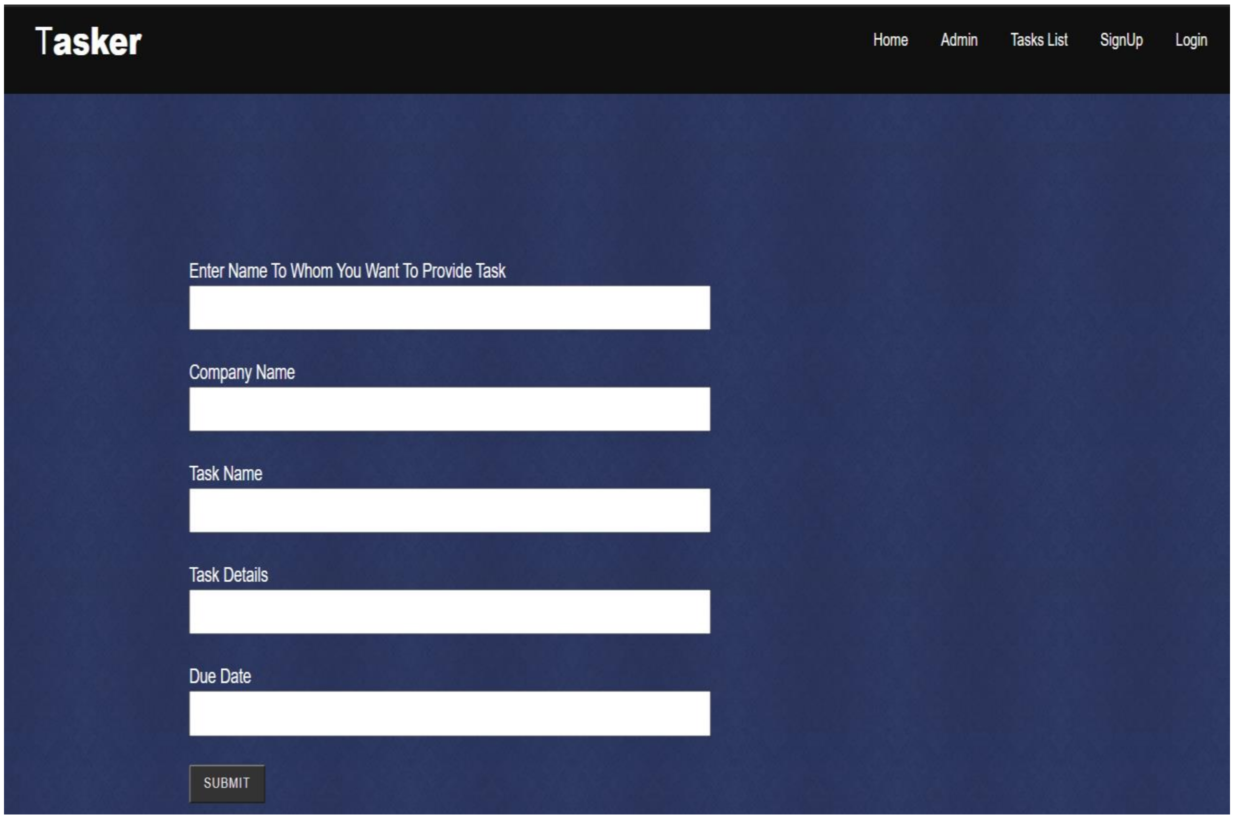
Delete Tasks of the Users

name	task	details	duedate	company	comment	Select
Jill	Form	Make a form for submission.	End of this weak	Company A	I am half way done.	■
Leon	Data Validation	Check if data is correct	30 July	Company A		■
Mark	Project	complete	22 July	Company	My Progress	■
Michael	Data Fill	Fill all the data	22 July	Company A		■
Yadav	ER Model	Make ER model	September 1st	Company C		■

Provide Task Page:

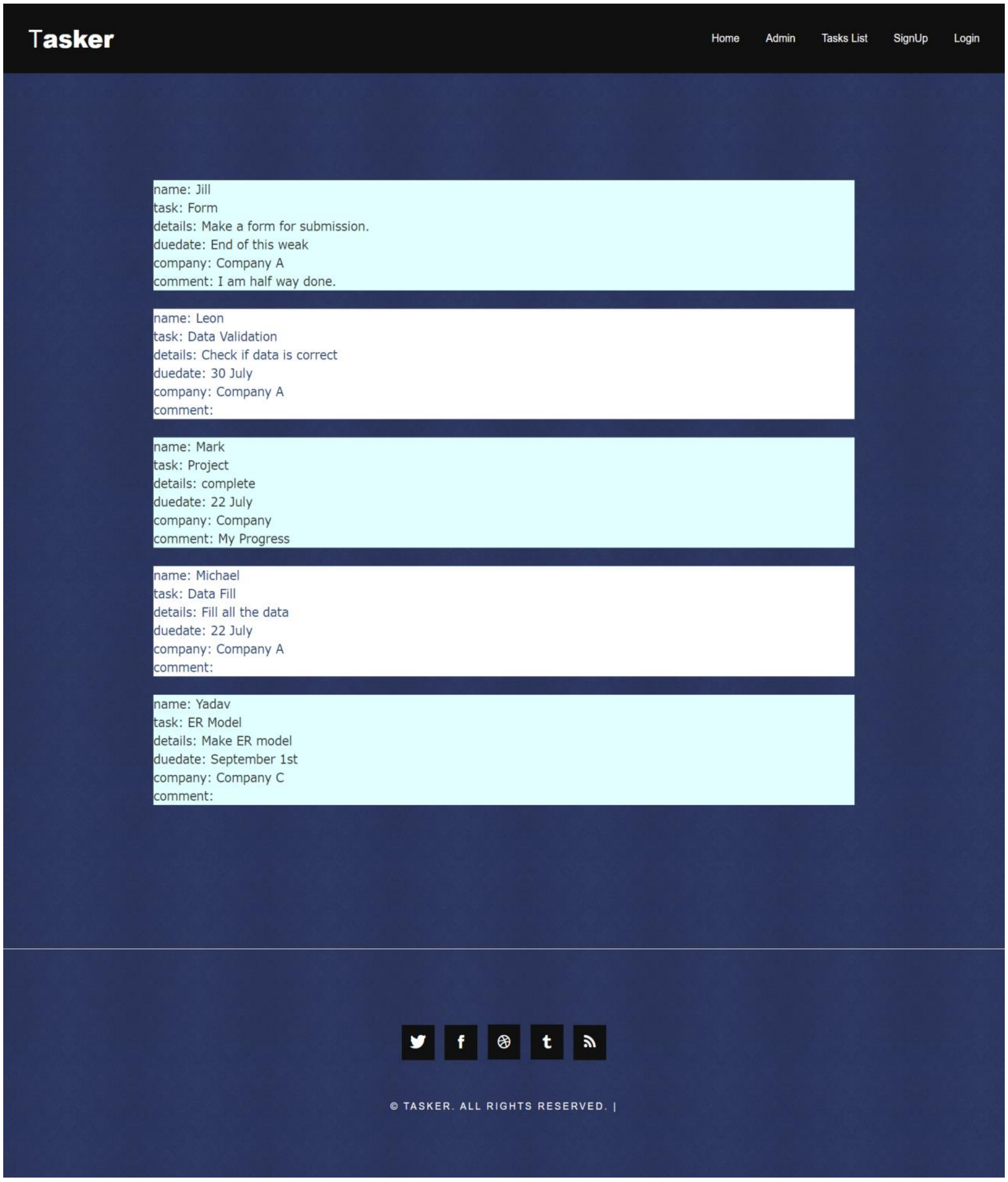
Here admin can provide information of tasks of various employees. Admin has to give correct name. The name should be same the name of the username from user signs up.

Admin can prove tasks details and Last date for the task to be completed.



Task List Page:

In this page all the added tasks can be seen of various employees. In this section we can also see the comment written by the employee.



name: Jill
task: Form
details: Make a form for submission.
duedate: End of this weak
company: Company A
comment: I am half way done.

name: Leon
task: Data Validation
details: Check if data is correct
duedate: 30 July
company: Company A
comment:

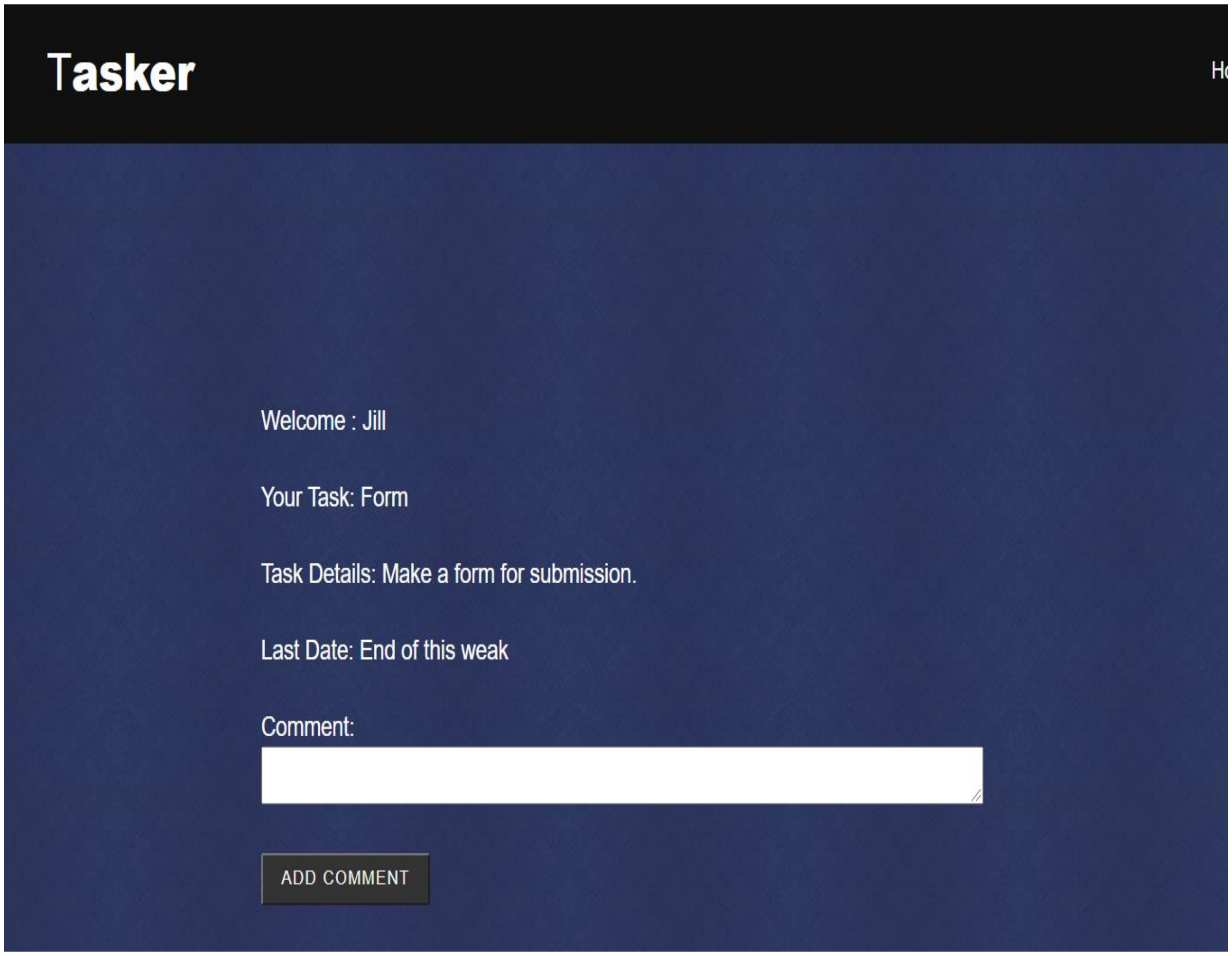
name: Mark
task: Project
details: complete
duedate: 22 July
company: Company
comment: My Progress

name: Michael
task: Data Fill
details: Fill all the data
duedate: 22 July
company: Company A
comment:

name: Yadav
task: ER Model
details: Make ER model
duedate: September 1st
company: Company C
comment:

My Task Page:

In this page user can see their task information. The employee can also comment to let the admin know about their progress in the task.



Welcome : Jill

Your Task: Form

Task Details: Make a form for submission.

Last Date: End of this weak

Comment:

Testing

ASP.NET validation controls validate the user input data to ensure that useless, unauthenticated, or contradictory data don't get stored.

ASP.NET provides the following validation controls:

- Required Field Validator
- Range Validator
- Compare Validator
- Regular Expression Validator
- Custom Validator
- Validation Summary

Required Field Validator Control:

The Required Field Validator control ensures that the required field is not empty. It is generally tied to a text box to force input into the text box.

```
<asp:RequiredFieldValidator ID="RequiredFieldValidator3" runat="server"

Style="top: 98px; left: 367px; position: absolute; height: 26px; width: 162px"

ErrorMessage="password required" ControlToValidate="TextBox2">

</asp:RequiredFieldValidator>
```

Compare Validator Control:

The Compare Validator control allows you to make comparison to compare data entered in an input control with a constant value or a value in a different control.

It can most commonly be used when you need to confirm password entered by the user at the registration time. The data is always case sensitive.

```
<asp:RequiredFieldValidator ID="RequiredFieldValidator2" runat="server" Style="top: 145px;

    left: 367px; position: absolute; height: 26px; width: 162px" ErrorMessage="password required"

    ControlToValidate="TextBox3"></asp:RequiredFieldValidator>
```

RegularExpression Validator Control:

A regular expression is a powerful pattern matching language that can be used to identify simple and complex characters sequence that would otherwise require writing code to perform.

Using RegularExpressionValidator server control, you can check a user's input based on a pattern that you define using a regular expression.

It is used to validate complex expressions. These expressions can be phone number, email address, zip code and many more. Using Regular Expression Validator is very simple. Simply set the ValidationExpression property to any type of expression you want and it will validate it.

If you don't find your desired regular expression, you can create your custom one.

In the example I have checked the email id format:

```
<asp:RegularExpressionValidator ID="RegularExpressionValidator1" runat="server" Style="top: 234px;
      left: 366px; position: absolute; height: 22px; width: 177px"
      ErrorMessage="RegularExpressionValidator" ControlToValidate="TextBox5"
      ValidationExpression="\w+([-+.']\w+)*@\w+([-.]\w+)*\.\w+([-
.]\w+)*"></asp:RegularExpressionValidator>
```

Conclusion

Task management includes planning, analyzing, evaluating and reporting about a particular task's progress. It an important aspect of the management a project because it helps to follow every task thoroughly. Regardless of the size and complexity of the task, task management is an absolute necessity when it comes to completing a project at time and budget.

In order for the process of task management to be effective and efficient, it is imperative that the project manager adheres to a schedule that will cover all important aspects of each task such as status, time, deadline, costs, and assigned resources.

It is good to use knowledge available to us and use it for the benefit of ourselves and others around us.

After the whole project has been developed, I have learned some important aspects of website building and importance of task management.

"The best preparation for good work tomorrow is to do good work today". The project "**Tasker**" can be easily enhanced with the advent of new technology and new changes can be added with more features.

BIBLIOGRAPHY

- www.google.com
- www.w3shools.com
- https://stackoverflow.com/
- https://www.tutorialspoint.com/index.htm
- https://www.javatpoint.com/asp-net-tutorial
- www.msdn.microsoft.com
- en.wikipedia.org
- ASP.NET 3.5: Stephen Walther, Pearson Education, 2005
- Professional ASP.NET 3.5 in C#, Bill Evjen Wiley Publications,2008
- ASP.NET 4 Unleashed by Stephen Walther, Kevin Hoffman, Nate Dudek