Notificare prin e-mail folosind SMTP. Functionalitatea de cautare.

Notificare prin e-mail folosind SMTP

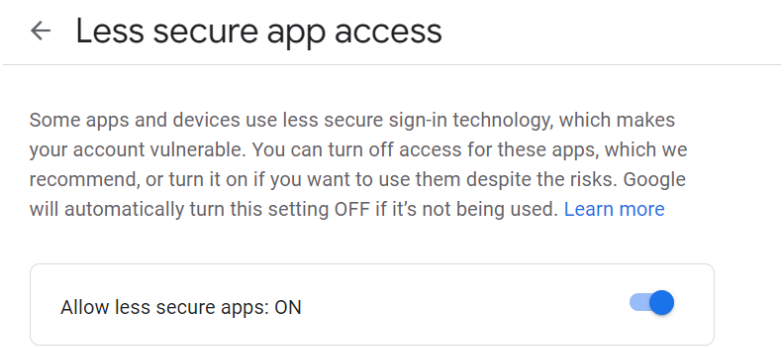
Un server **SMTP** (Simple Mail Transfer Protocol) este o aplicatie cu ajutorul careia se pot trimite/primi e-mailuri, prin intermediul unei adrese configurata in prealabil. Pentru un astfel de proces, serverul SMTP este obligatoriu pentru ca mesajul sa ajunga la destinatie. In momentul in care se trimite e-mailul, mesajul o sa fie un string care ulterior se va trimite catre server. Serverul proceseaza mesajul si il trimite catre destinatie. De asemenea, serverul valideaza si contul catre care urmeaza sa trimita mesajul.

Utilizand aplicatia “Articles” o sa integram SMTP pentru trimiterea de notificari catre utilizatorul care a postat articolul (posesorul articolului) in momentul in care un alt utilizator posteaza un comentariu in cadrul respectivului articol.

```
/**
 * Send an email to an email address with a specific subject and content
 *
 * @param string toEmail - email address to send the email to
 * @param string subject - the subject of the sent email
 * @param string content - content of the email
 *
 * @return void
 */
private void SendEmailNotification(string toEmail, string subject, string content)
{

    /**
     * SMTP server configuration
     * senderEmail - SMTP username
     * senderPassword - SMTP Password
     * smtpServer - SMTP Server / HOST
     * smtpPort - the port used to connect on the SMTP server
     * ----------------------------------------------------------
     * Sending an email using Google SMTP:
     * Gmail by default prevents access for your e-mail account from custom applications. You can set it up to accept the login from your
       application. You need to go to security settings at the followig link https://www.google.com/settings/security/lesssecureapps and
       enable less secure apps. So that you will be able to login from all apps.
     */

    const string senderEmail = "testemaildaw@gmail.com";
    const string senderPassword = "parola";
    const string smtpServer = "smtp.gmail.com";
    const int smtpPort = 587;
```

```
            // Create a new SMTP Client that is used to send emails
            SmtpClient smtpClient = new SmtpClient(smtpServer, smtpPort);
            smtpClient.DeliveryMethod = SmtpDeliveryMethod.Network;
            smtpClient.EnableSsl = true;
            smtpClient.UseDefaultCredentials = false;
            smtpClient.Credentials = new NetworkCredential(senderEmail, senderPassword);

            // Create a new email object
            // @param senderEmail - the email address who sends the email
            // @param toEmail - the recipient of the email
            // @param subject - the subject line of the email
            // @param content - the content of the email
            MailMessage email = new MailMessage(senderEmail, toEmail, subject, content);
            // Set true so the email is received as HTML
            email.IsBodyHtml = true;
            // Set the encoding to UTF8 to allow special characters
            email.BodyEncoding = UTF8Encoding.UTF8;

            try
            {
                // Send the emaill
                System.Diagnostics.Debug.WriteLine("Sending email...");
                smtpClient.Send(email);
                System.Diagnostics.Debug.WriteLine("Email sent!");
            }
            catch (Exception e)
            {
                // Failed to send the e-mail message
                System.Diagnostics.Debug.WriteLine("Error occured while trying to send email");
                System.Diagnostics.Debug.WriteLine(e.Message.ToString());
            }
        }
    }
}
```

https://www.google.com/settings/security/lesssecureapps



```
[HttpPost]
[Authorize(Roles = "User,Editor,Admin")]
public ActionResult Show(Comment comm)
{
    comm.Date = DateTime.Now;
    comm.UserId = User.Identity.GetUserId();
    try
    {
        if (ModelState.IsValid)
        {
            db.Comments.Add(comm);
            db.SaveChanges();


            // Get the article for which the comment was added
            // comm este o instanta primita de la view, care inca nu are legatura cu articolul pentru care a fost adaugat
            Article commArticle = db.Articles.Find(comm.ArticleId);
            string authorEmail = commArticle.User.Email;

            // Create the notification body
            string notificationBody = "<p>A fost adaugat un nou comentariu la articolul Dvs cu titlul:</p>";
            notificationBody += "<p><strong>" + commArticle.Title + "<strong></p>";
            notificationBody += "<br />";
            notificationBody += "Comentariul adaugat este: <br /><br /><em>";
            notificationBody += comm.Content;
            notificationBody += "</em>";
            notificationBody += "<br /><br />O zi frumoasa!";

            // Send a notification to the article author that a new comment has been added to the article
            SendEmailNotification(authorEmail, "Un nou comentariu a fost adaugat la articolul Dvs.", notificationBody);

            return Redirect("/Articles/Show/" + comm.ArticleId);
        }

        else
        {
            Article a = db.Articles.Find(comm.ArticleId);

            SetAccessRights();

            return View(a);
        }

    }

    catch (Exception e)
    {
        Article a = db.Articles.Find(comm.ArticleId);

        SetAccessRights();

        return View(a);
    }

}
```

Functionalitatea de cautare

In pagina de listare a tuturor articolelor vom integra functionalitatea de cautare a acestora dupa titlu, continut, dar si dupa continutul comentariilor postate de utilizatori in cadrul articolelor respective.

Index.cshtml

```
<div class="container">
        <br />
        <div class="row justify-content-center">
            <div class="col-12 col-md-10 col-lg-8">
                <form class="card card-sm" method="GET">
                    <div class="card-body row no-gutters align-items-center">
                        <div class="col-auto">
                            <i class="fas fa-search h4 text-body"></i>
                        </div>
                        <!--end of col-->
                        <div class="col">
                            <input class="form-control form-control-lg form-control-
borderless" type="text" name="search"
                            value="@ViewBag.SearchString"
                            placeholder="Search topics or keywords">
                        </div>
                        <!--end of col-->
                        <div class="col-auto">
                            <button class="btn btn-lg btn-success"
type="submit">Search</button>
                        </div>
                        <!--end of col-->
                    </div>
                </form>
            </div>
            <!--end of col-->
        </div>
    </div>
```

```
    <div>
       <nav aria-label="Page navigation">
           <ul class="pagination">
               <li>
                   @if (ViewBag.search != "")
                   {
                       <a href="/Articles/Index?page=1&search=@ViewBag.SearchString"
aria-label="Previous">
                           <span aria-hidden="true">&laquo;</span>
                       </a>
                   }
                   else
                   {
                       <a href="/Articles/Index?page=1" aria-label="Previous">
                           <span aria-hidden="true">&laquo;</span>
                       </a>
                   }


               </li>

               @for (int i = 1; i <= ViewBag.lastPage; i++)
               {
                   if (ViewBag.search != "")
                   {
                       <li><a
href="/Articles/Index?page=@i&search=@ViewBag.SearchString">@(i)</a></li>
                   }
                   else
                   {
                       <li><a href="/Articles/Index?page=@i">@(i)</a></li>
                   }

               }

                       <li>
                           @if (ViewBag.search != "")
                           {
                               <a
href="/Articles/Index?page=@(ViewBag.lastPage)&search=@ViewBag.SearchString" aria-
label="Next">
                                   <span aria-hidden="true">&raquo;</span>
                               </a>
                           }
                           else
                           {
                               <a href="/Articles/Index?page=@(ViewBag.lastPage)" aria-
label="Next">
                                   <span aria-hidden="true">&raquo;</span>
                               </a>
                           }

                       </li>
           </ul>
       </nav>
   </div>
```

```
// GET: Article
[Authorize(Roles = "User,Editor,Admin")]
public ActionResult Index()
{

    var articles = db.Articles.Include("Category").Include("User").OrderBy(a => a.Date);
    var search = "";

    if(Request.Params.Get("search") != null)
    {
        search = Request.Params.Get("search").Trim(); // trim whitespace from search string
        // Search in articles (title and content)
        List<int> articleIds = db.Articles.Where(
            at => at.Title.Contains(search)
            || at.Content.Contains(search)
            ).Select(a => a.Id).ToList();

        // Search in comments (content)
        List<int> commentIds = db.Comments.Where(c => c.Content.Contains(search))
            .Select(com => com.ArticleId).ToList();

        // Unique list of articles
        List<int> mergedIds = articleIds.Union(commentIds).ToList();


        // List of articles that contain the search string either in article title, content or comments
        articles = db.Articles.Where(article => mergedIds.Contains(article.Id)).Include("Category").Include("User").OrderBy(a => a.Date);




    }


      var totalItems = articles.Count();
      var currentPage = Convert.ToInt32(Request.Params.Get("page"));

      var offset = 0;

      if (!currentPage.Equals(0))
      {
          offset = (currentPage - 1) * this._perPage;
      }

      var paginatedArticles = articles.Skip(offset).Take(this._perPage);

      if (TempData.ContainsKey("message"))
      {
          ViewBag.message = TempData["message"].ToString();
      }

      //ViewBag.perPage = this._perPage;
      ViewBag.total = totalItems;
      ViewBag.lastPage = Math.Ceiling((float)totalItems / (float)this._perPage);
      ViewBag.Articles = paginatedArticles;
      ViewBag.SearchString = search;

      return View();
  }
```