

Ejercicio No. 4. Instituto (JSP, Gráficas y Login)

Barrera Pérez Carlos Tonatihu
Profesor: José Asunción Enríquez Zárate
Web Application Development
Grupo: 3CM9

3 de junio de 2019

Índice

1. Introducción	3
2. Desarrollo	3
2.1. Código	3
2.2. Pruebas	37
3. Conclusiones	44

1. Introducción

Este ejercicio tuvo como objetivo modificar el ejercicio 3 y en lugar de utilizar solo servlets, utilizar JSPs para la vista, es importante señalar que la base de datos se modificó ya que ahora también se incluye la funcionalidad de inicio de sesión y se tienen perfiles de profesor y alumno, la base de datos final es la que se muestra en la figura 1.

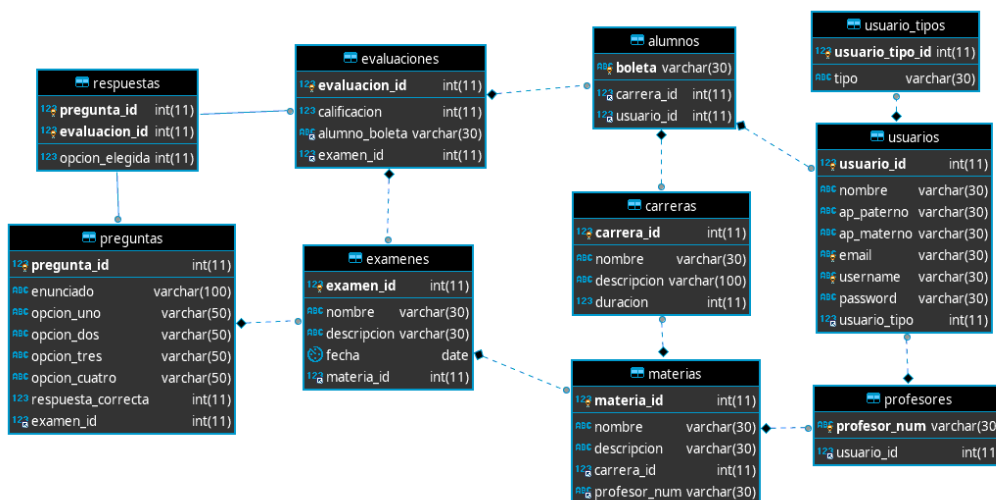


Figura 1: Base de datos que se trabajó en MySQL

Otra funcionalidad que se le agregó es la generación de gráfica de cantidad de alumnos por carrera, así como el envío de dicha imagen al correo del usuario cuya sesión se encuentre activa y también se manda un correo si se requiere recuperar la contraseña.

Finalmente, se utiliza un filtro para manejar la autenticación de los usuarios y poder controlar las sesiones.

2. Desarrollo

2.1. Código

```
1 package utils;  
2  
3 import java.util.Map;
```

```

4 import java.util.Properties;
5 import java.util.logging.Level;
6 import java.util.logging.Logger;
7 import javax.activation.DataHandler;
8 import javax.activation.DataSource;
9 import javax.activation.FileDataSource;
10 import javax.mail.Authenticator;
11 import javax.mail.BodyPart;
12 import javax.mail.Message;
13 import javax.mail.MessagingException;
14 import javax.mail.PasswordAuthentication;
15 import javax.mail.Session;
16 import javax.mail.Transport;
17 import javax.mail.internet.AddressException;
18 import javax.mail.internet.InternetAddress;
19 import javax.mail.internet.MimeBodyPart;
20 import javax.mail.internet.MimeMessage;
21 import javax.mail.internet.MimeMultipart;
22
23 /**
24  *
25  * @author tonatihu
26  * Created on 10-Mar-2019
27  */
28 public class EnvioEmail {
29     private String asunto;
30     private String destinatario;
31     private String mensaje;
32     private Properties props;
33     private static final String USERNAME = "
webappdevtona@outlook.com";
34     private static final String PASSWORD = "jodanse90";
35     private final Session session;
36
37     public EnvioEmail() {
38         props = new Properties();
39         props.put("mail.smtp.auth", "true");
40         props.put("mail.smtp.starttls.enable", "true");
41         props.put("mail.smtp.host", "smtp.office365.com");
42         props.put("mail.smtp.port", "587");
43         session = Session.getInstance(props, new
Authenticator() {
44             @Override

```

```

45         protected PasswordAuthentication
getPasswordAuthentication() {
46         return new PasswordAuthentication(USERNAME,
PASSWORD);
47     }
48     });
49 }
50
51 public String getAsunto() {
52     return asunto;
53 }
54
55 public void setAsunto(String asunto) {
56     this.asunto = asunto;
57 }
58
59 public String getDestinatario() {
60     return destinatario;
61 }
62
63 public void setDestinatario(String destinatario) {
64     this.destinatario = destinatario;
65 }
66
67 public String getMensaje() {
68     return mensaje;
69 }
70
71 public void setMensaje(String mensaje) {
72     this.mensaje = mensaje;
73 }
74
75 public void enviar(Map<String , String> imagenes , Map<
String , String>
76 archivos) {
77     try {
78         Message message = new MimeMessage(session);
79         message.setFrom(new InternetAddress(USERNAME));
80         message.setRecipients(Message.RecipientType.TO,
81 InternetAddress.parse(destinatario));
82         message.setSubject(asunto);
83
84         MimeMultipart multipart = new MimeMultipart("

```

```

related");
85
86         BodyPart bodyPartMensaje = new MimeBodyPart();
87         bodyPartMensaje.setContent(mensaje, "text/html;
charset=UTF-8");
88         multipart.addBodyPart(bodyPartMensaje);
89
90         for (Map.Entry<String, String> entrada :
imagenes.entrySet()) {
91             BodyPart bodyPartImágenes = new
MimeBodyPart();
92             DataSource fds = new FileDataSource(entrada
.getValue());
93             bodyPartImágenes.setDataHandler(new
DataHandler(fds));
94             bodyPartImágenes.setHeader("Content-ID", "<
" + entrada.getKey()
95 + ">");
96             multipart.addBodyPart(bodyPartImágenes);
97         }
98
99         for (Map.Entry<String, String> entrada :
archivos.entrySet()) {
100             BodyPart bodyPartArchivos = new
MimeBodyPart();
101             DataSource fds2 = new FileDataSource(
entrada.getValue());
102             bodyPartArchivos.setDataHandler(new
DataHandler(fds2));
103             bodyPartArchivos.setFileName(entrada.getKey
());
104             multipart.addBodyPart(bodyPartArchivos);
105         }
106
107         message.setContent(multipart);
108
109         Transport.send(message);
110     } catch (AddressException ex) {
111         Logger.getLogger(EnvioEmail.class.getName()).
log(Level.SEVERE,
112 "Error AddressException", ex);
113     } catch (MessagingException ex) {
114         Logger.getLogger(EnvioEmail.class.getName()).

```

```

        log(Level.SEVERE,
115 "Error MessagingException", ex);
116     }
117 }
118 }

```

Archivo 1: EnvioEmail.java

```

1 package controller;
2
3 import dao.UsuarioDao;
4 import dao.impl.UsuarioDaoImpl;
5 import dto.Usuario;
6 import java.io.IOException;
7 import java.sql.SQLException;
8 import java.util.HashMap;
9 import java.util.Map;
10 import java.util.logging.Level;
11 import java.util.logging.Logger;
12 import javax.servlet.ServletException;
13 import javax.servlet.annotation.WebServlet;
14 import javax.servlet.http.HttpServlet;
15 import javax.servlet.http.HttpServletRequest;
16 import javax.servlet.http.HttpServletResponse;
17 import utils.EnvioEmail;
18
19 /**
20  *
21  * @author tonatihu
22  * Created on 09-Mar-2019
23  */
24 @WebServlet(name="ForgotPassword", urlPatterns={"/
    forgotPassword"})
25 public class RecuperarContraServlet extends HttpServlet {
26
27     /**
28      * Processes requests for both HTTP <code>GET</code>
29      and <code>POST</code>
30      methods.
31      * @param request servlet request
32      * @param response servlet response
33      * @throws ServletException if a servlet-specific error
34      occurs

```

```

33      * @throws IOException if an I/O error occurs
34      */
35      protected void processRequest(HttpServletRequest
request ,
36      HttpServletResponse response)
37      throws ServletException , IOException {
38          request.setCharacterEncoding("UTF-8");
39          String email = request.getParameter("email");
40          UsuarioDao dao = new UsuarioDaoImpl();
41          Map<String , String> imagenesMap = new HashMap<>();
42          Map<String , String> archivosMap = new HashMap<>();
43
44          if (email != null) {
45              try {
46                  EnvioEmail e = new EnvioEmail();
47                  e.setAsunto("Recuperar contrasena de
Instituto");
48                  e.setDestinatario(email);
49                  Usuario u = dao.findByEmail(email);
50                  if (u != null) {
51                      String mensaje = "<h1>Bienvenido a
Instituto </h1><br><h2>Tu
52 contrasena es:</h2> "
53                          + u.getPassword();
54                      e.setMensaje(mensaje);
55                      e.enviar(imagenesMap , archivosMap);
56                      response.sendRedirect("login");
57                      return;
58                  }
59              } catch (SQLException ex) {
60
61          Logger.getLogger(RecuperarContraServlet.class.getName()).
log(Level.SEVERE, null ,
62 ex);
63          }
64
65      }
66      request.getRequestDispatcher("forgotPassword.jsp").
forward(request ,
67 response);
68      }
69
70      // <editor-fold defaultstate="collapsed" desc="

```


HttpServlet methods. Click on

```

71 the + sign on the left to edit the code.">
72 /**
73  * Handles the HTTP <code>GET</code> method.
74  * @param request servlet request
75  * @param response servlet response
76  * @throws ServletException if a servlet-specific error
    occurs
77  * @throws IOException if an I/O error occurs
78  */
79 @Override
80 protected void doGet(HttpServletRequest request,
    HttpServletResponse
81 response)
82     throws ServletException, IOException {
83     processRequest(request, response);
84 }
85
86 /**
87  * Handles the HTTP <code>POST</code> method.
88  * @param request servlet request
89  * @param response servlet response
90  * @throws ServletException if a servlet-specific error
    occurs
91  * @throws IOException if an I/O error occurs
92  */
93 @Override
94 protected void doPost(HttpServletRequest request,
    HttpServletResponse
95 response)
96     throws ServletException, IOException {
97     processRequest(request, response);
98 }
99
100 /**
101  * Returns a short description of the servlet.
102  * @return a String containing servlet description
103  */
104 @Override
105 public String getServletInfo() {
106     return "Short description";
107 }// </editor-fold>
108

```

109 }

Archivo 2: RecuperarContraServlet.java

```
1 package controller;
2
3 import dao.AlumnoDao;
4 import dao.impl.AlumnoDaoImpl;
5 import dto.Datos;
6 import dto.Usuario;
7 import java.io.File;
8 import java.io.IOException;
9 import java.sql.SQLException;
10 import java.util.HashMap;
11 import java.util.List;
12 import java.util.Map;
13 import java.util.logging.Level;
14 import java.util.logging.Logger;
15 import javax.servlet.ServletException;
16 import javax.servlet.annotation.WebServlet;
17 import javax.servlet.http.HttpServlet;
18 import javax.servlet.http.HttpServletRequest;
19 import javax.servlet.http.HttpServletResponse;
20 import javax.servlet.http.HttpSession;
21 import org.jfree.chart.ChartFactory;
22 import org.jfree.chart.ChartUtilities;
23 import org.jfree.chart.JFreeChart;
24 import org.jfree.data.general.DefaultPieDataset;
25 import utils.EnvioEmail;
26 import utils.Paginas;
27
28 /**
29  *
30  * @author tonatihu
31  * Created on 10-Mar-2019
32  */
33 @WebServlet(name="GraficasServlet", urlPatterns={"/graficas"
34         "})
35 public class GraficasServlet extends HttpServlet {
36     private static final Logger LOGGER =
37     Logger.getLogger(GraficasServlet.class.getName());
38
39     /**
```

```

39      * Processes requests for both HTTP GET
    and POST
40 methods.
41      * @param request servlet request
42      * @param response servlet response
43      * @throws ServletException if a servlet-specific error
    occurs
44      * @throws IOException if an I/O error occurs
45      */
46      protected void processRequest(HttpServletRequest
    request ,
47      HttpServletResponse response)
48      throws ServletException , IOException {
49          response.setContentType("text/html; charset=UTF-8");
50          request.setCharacterEncoding("UTF-8");
51          String accion = request.getParameter("action");
52          if (accion.equals("ver"))
53              ver(request , response);
54          else if (accion.equals("generar"))
55              enviar(request , response);
56      }
57
58      // <editor-fold defaultstate="collapsed" desc="
    HttpServlet methods. Click on
59 the + sign on the left to edit the code.">
60      /**
61       * Handles the HTTP GET method.
62       * @param request servlet request
63       * @param response servlet response
64       * @throws ServletException if a servlet-specific error
    occurs
65       * @throws IOException if an I/O error occurs
66       */
67      @Override
68      protected void doGet(HttpServletRequest request ,
    HttpServletResponse
69      response)
70      throws ServletException , IOException {
71          processRequest(request , response);
72      }
73
74      /**
75       * Handles the HTTP POST method.

```

```

76     * @param request servlet request
77     * @param response servlet response
78     * @throws ServletException if a servlet-specific error
    occurs
79     * @throws IOException if an I/O error occurs
80     */
81     @Override
82     protected void doPost(HttpServletRequest request,
    HttpServletResponse
83 response)
84     throws ServletException, IOException {
85         processRequest(request, response);
86     }
87
88     /**
89     * Returns a short description of the servlet.
90     * @return a String containing servlet description
91     */
92     @Override
93     public String getServletInfo() {
94         return "Short description";
95     } // </editor-fold>
96
97     private void generarGrafica(List<Datos> datos) throws
    IOException {
98         DefaultPieDataset dpd = new DefaultPieDataset();
99         datos.forEach((d) -> {
100             dpd.setValue(d.getNombre(), d.getCantidad());
101         });
102         String archivo =
103 getServletConfig().getServletContext().getRealPath("/static
    /img/grafica.png");
104         JFreeChart chart = ChartFactory.createPieChart("
    Cantidad de alumnos por
105 carrera",
106             dpd, true, true, false);
107         ChartUtilities.saveChartAsPNG(new File(archivo),
    chart, 400, 300);
108     }
109
110     private void ver(HttpServletRequest request,
    HttpServletResponse response)
111         throws IOException, ServletException,

```

```

        ServletException {
112         try {
113             request.setAttribute("PAGINA", Paginas.GRAFICAS
        );
114             AlumnoDao dao = new AlumnoDaoImpl();
115             List<Datos> datos = dao.getData();
116             request.setAttribute("lista", datos);
117             generarGrafica(datos);
118             request.getRequestDispatcher("graficas.jsp").
        forward(request,
119 response);
120         } catch (SQLException ex) {
121             Logger.getLogger(GraficasServlet.class.getName
        ()).log(Level.SEVERE,
122 null, ex);
123         }
124     }
125
126     private void enviar(HttpServletRequest request,
        HttpServletResponse
127 response) throws IOException {
128         HttpSession session = request.getSession();
129         Usuario u = (Usuario) session.getAttribute("
        USUARIO_SESSION");
130         EnvioEmail e = new EnvioEmail();
131         e.setAsunto("Envio de la cantidad de alumnos");
132         e.setDestinatario(u.getEmail());
133         String archivo =
134 getServletConfig().getServletContext().getRealPath("/static
        /img/grafica.png");
135         Map<String, String> imagenesMap = new HashMap<>();
136         Map<String, String> archivosMap = new HashMap<>();
137         imagenesMap.put("grafica", archivo);
138         String mensaje = "<h1>Bienvenido a Instituto </h1><
        br><h2>Te mandamos una
139 grafica bien chida</h2> <img src=\"cid:grafica\"/> ";
140         e.setMensaje(mensaje);
141         e.enviar(imagenesMap, archivosMap);
142         response.sendRedirect("graficas?action=ver");
143     }
144 }

```

Archivo 3: GraficasServlet.java

```

1 package interceptor;
2
3 import java.io.IOException;
4 import java.io.PrintWriter;
5 import java.io.StringWriter;
6 import javax.servlet.Filter;
7 import javax.servlet.FilterChain;
8 import javax.servlet.FilterConfig;
9 import javax.servlet.ServletException;
10 import javax.servlet.ServletRequest;
11 import javax.servlet.ServletResponse;
12 import javax.servlet.annotation.WebFilter;
13 import javax.servlet.http.HttpServletRequest;
14 import javax.servlet.http.HttpServletResponse;
15 import javax.servlet.http.HttpSession;
16
17 /**
18  *
19  * @author tonatihu
20  */
21 @WebFilter(filterName = "AuthenticationFilter", urlPatterns
    = {"/*"})
22 public class AuthenticationFilter implements Filter {
23
24     private static final boolean DEBUG = true;
25     private FilterConfig filterConfig = null;
26
27     public AuthenticationFilter() {
28     }
29     private static final String[] NO_REQUIERE_LOGIN = {
30         "/static", "/login", "/signup", "forgotPassword
31     };
32
33     private void doBeforeProcessing(ServletRequest request,
34     ServletResponse
35     response)
36         throws IOException, ServletException {
37         if (DEBUG) {
38             log("AuthenticationFilter: DoBeforeProcessing");
39         }
40     }

```

```

40
41     private void doAfterProcessing(ServletRequest request ,
42     ServletResponse
43     response)
44         throws IOException , ServletException {
45         if (DEBUG) {
46             log("AuthenticationFilter:DoAfterProcessing");
47         }
48     }
49     @Override
50     public void doFilter(ServletRequest req ,
51     ServletResponse resp ,
52     FilterChain chain) throws IOException ,
53     ServletException {
54         if (DEBUG) {
55             log("AuthenticationFilter:doFilter()");
56         }
57         doBeforeProcessing(req , resp);
58
59         HttpServletRequest request = (HttpServletRequest)
60         req;
61         request.setCharacterEncoding("UTF-8");
62         HttpServletResponse response = (HttpServletResponse
63         ) resp;
64         HttpSession session = request.getSession(false);
65
66         boolean isLoggedIn = (session != null &&
67         session.getAttribute("USUARIO_SESSION") != null);
68
69         String url = request.getRequestURI();
70         String logout = request.getParameter("logout");
71         if (isLoggedIn && url.equals("/instituto/login") &&
72         logout == null) {
73             if (DEBUG)
74                 log("Ya inicio sesion pero quiere hacerlo
75                 otra vez");
76             response.sendRedirect("home");
77         } else if (isLoggedIn || url.equals("/instituto/login
78         ") ||
79         loginRequerido(request)) {
80             if (DEBUG)
81                 log("Quiere acceder a cualquier pagina o al

```

```

login");
75         chain.doFilter(req, resp);
76     } else {
77         if (DEBUG)
78             log("No inicio sesion y se salta el login");
;
79         response.sendRedirect("login");
80     }
81
82     doAfterProcessing(req, resp);
83 }
84
85 private boolean loginRequerido(HttpServletRequest
request) {
86     String requestURL = request.getRequestURL().
toString();
87     for (String url : NO_REQUIERE_LOGIN) {
88         if (requestURL.contains(url))
89             return true;
90     }
91     return false;
92 }
93
94 public FilterConfig getFilterConfig() {
95     return (this.filterConfig);
96 }
97
98 public void setFilterConfig(FilterConfig filterConfig)
{
99     this.filterConfig = filterConfig;
100 }
101
102 @Override
103 public void destroy() {
104 }
105
106 @Override
107 public void init(FilterConfig filterConfig) {
108     this.filterConfig = filterConfig;
109     if (filterConfig != null) {
110         if (DEBUG) {
111             log("AuthenticationFilter: Initializing
filter");

```



```

112     }
113 }
114 }
115
116 @Override
117 public String toString() {
118     if (filterConfig == null) {
119         return ("AuthenticationFilter()");
120     }
121     StringBuilder sb = new StringBuilder("
AuthenticationFilter(");
122     sb.append(filterConfig);
123     sb.append(")");
124     return (sb.toString());
125 }
126
127 public static String getStackTrace(Throwable t) {
128     String stackTrace = null;
129     try {
130         StringWriter sw = new StringWriter();
131         PrintWriter pw = new PrintWriter(sw);
132         t.printStackTrace(pw);
133         pw.close();
134         sw.close();
135         stackTrace = sw.getBuffer().toString();
136     } catch (IOException ex) {
137     }
138     return stackTrace;
139 }
140
141 public void log(String msg) {
142     filterConfig.getServletContext().log(msg);
143 }
144
145 }

```

Archivo 4: AuthenticationFilter.java

```

1 package controller;
2
3 import dao.AlumnoDao;
4 import dao.CarreraDao;
5 import dao.UsuarioDao;

```

```

6 import dao.impl.AlumnoDaoImpl;
7 import dao.impl.CarreraDaoImpl;
8 import dao.impl.UsuarioDaoImpl;
9 import dto.Alumno;
10 import dto.Usuario;
11 import java.io.IOException;
12 import java.sql.SQLException;
13 import java.util.List;
14 import java.util.logging.Level;
15 import java.util.logging.Logger;
16 import javax.servlet.ServletException;
17 import javax.servlet.annotation.WebServlet;
18 import javax.servlet.http.HttpServlet;
19 import javax.servlet.http.HttpServletRequest;
20 import javax.servlet.http.HttpServletResponse;
21
22 /**
23  *
24  * @author tonatihu
25  * Created on 10-Mar-2019
26  */
27 @WebServlet(name="AlumnosServlet", urlPatterns={"/alumnos"
    })
28 public class AlumnosServlet extends HttpServlet {
29
30     /**
31      * Processes requests for both HTTP <code>GET</code>
32      and <code>POST</code>
33      methods.
34      * @param request servlet request
35      * @param response servlet response
36      * @throws ServletException if a servlet-specific error
37      occurs
38      * @throws IOException if an I/O error occurs
39      */
40     protected void processRequest(HttpServletRequest
41 request ,
42 HttpServletResponse response)
43     throws ServletException, IOException {
44         response.setContentType("text/html; charset=UTF-8");
45         request.setCharacterEncoding("UTF-8");
46         String action = request.getParameter("action");
47         if (action.equals("ver"))

```

```

45         mostrarAlumnos(request, response);
46     else if (action.equals("eliminar"))
47         eliminarAlumno(request, response);
48
49 }
50
51 // <editor-fold defaultstate="collapsed" desc="
52 // HttpServlet methods. Click on
53 the + sign on the left to edit the code.">
54 /**
55  * Handles the HTTP <code>GET</code> method.
56  * @param request servlet request
57  * @param response servlet response
58  * @throws ServletException if a servlet-specific error
59  * occurs
60  * @throws IOException if an I/O error occurs
61  */
62 @Override
63 protected void doGet(HttpServletRequest request,
64 HttpServletResponse
65 response)
66     throws ServletException, IOException {
67     processRequest(request, response);
68 }
69
70 /**
71  * Handles the HTTP <code>POST</code> method.
72  * @param request servlet request
73  * @param response servlet response
74  * @throws ServletException if a servlet-specific error
75  * occurs
76  * @throws IOException if an I/O error occurs
77  */
78 @Override
79 protected void doPost(HttpServletRequest request,
80 HttpServletResponse
81 response)
82     throws ServletException, IOException {
83     processRequest(request, response);
84 }
85
86 /**
87  * Returns a short description of the servlet.

```

```

83     * @return a String containing servlet description
84     */
85     @Override
86     public String getServletInfo() {
87         return "Short description";
88     }// </editor-fold>
89
90     private void mostrarAlumnos(HttpServletRequest request,
91                                 HttpServletResponse
92 response)
93         throws ServletException, IOException {
94         try {
95             request.setAttribute("PAGINA", 7);
96             AlumnoDao dao = new AlumnoDaoImpl();
97             CarreraDao daoCarrera = new CarreraDaoImpl();
98             List<Alumno> alumnos = dao.readAll();
99             for (Alumno alumno : alumnos) {
100                 alumno.setCarrera(daoCarrera.read(alumno.
101 getCarrera()));
102             }
103             request.setAttribute("alumnos", alumnos);
104             request.getRequestDispatcher("listaAlumnos.jsp"
105 ).forward(request,
106 response);
107         } catch (SQLException ex) {
108             Logger.getLogger(AlumnosServlet.class.getName())
109                 .log(Level.SEVERE,
110 null, ex);
111         }
112     }
113
114     private void eliminarAlumno(HttpServletRequest request,
115                                 HttpServletResponse
116 response)
117         throws IOException {
118         try {
119             UsuarioDao dao = new UsuarioDaoImpl();
120             int id = Integer.parseInt(request.getParameter(
121 "id"));
122             Usuario usuario = new Usuario();
123             usuario.setId(id);
124             dao.delete(usuario);
125             response.sendRedirect("alumnos?action=ver");
126         }
127     }

```

```

120         } catch (SQLException ex) {
121             Logger.getLogger(AlumnosServlet.class.getName())
122                 .log(Level.SEVERE,
123                     null, ex);
124         }
125     }
126 }

```

Archivo 5: AlumnosServlet.java

```

1 package controller;
2
3 import dao.CarreraDao;
4 import dao.impl.CarreraDaoImpl;
5 import dto.Carrera;
6 import java.io.IOException;
7 import java.sql.SQLException;
8 import java.util.List;
9 import java.util.logging.Level;
10 import java.util.logging.Logger;
11 import javax.servlet.ServletException;
12 import javax.servlet.annotation.WebServlet;
13 import javax.servlet.http.HttpServlet;
14 import javax.servlet.http.HttpServletRequest;
15 import javax.servlet.http.HttpServletResponse;
16 import utils.Paginas;
17
18 /**
19  *
20  * @author tonatihu
21  * Created on 10-Mar-2019
22  */
23 @WebServlet(name="CarrerasServlet", urlPatterns={"/carreras"
24     "})
25 public class CarrerasServlet extends HttpServlet {
26     /**
27      * Processes requests for both HTTP <code>GET</code>
28      * and <code>POST</code>
29      * methods.
30      * @param request servlet request
31      * @param response servlet response

```

```

31      * @throws ServletException if a servlet-specific error
      occurs
32      * @throws IOException if an I/O error occurs
33      */
34      protected void processRequest(HttpServletRequest request
request ,
35 HttpServletResponse response)
36      throws ServletException , IOException {
37          response.setContentType("text/html; charset=UTF-8");
38          request.setCharacterEncoding("UTF-8");
39          String action = request.getParameter("action");
40          switch (action) {
41              case "ver":
42                  verCarreras(request , response);
43                  break;
44              case "eliminar":
45                  eliminarCarrera(request , response);
46                  break;
47              case "editar":
48                  editarCarrera(request , response);
49                  break;
50              case "agregar":
51                  agregarCarrera(request , response);
52                  break;
53              case "guardar":
54                  guardarCarrera(request , response);
55                  break;
56              default:
57                  response.sendRedirect("home");
58                  break;
59          }
60      }
61
62      // <editor-fold defaultstate="collapsed" desc="
        HttpServlet methods. Click on
63 the + sign on the left to edit the code.">
64      /**
65       * Handles the HTTP <code>GET</code> method.
66       * @param request servlet request
67       * @param response servlet response
68       * @throws ServletException if a servlet-specific error
        occurs
69       * @throws IOException if an I/O error occurs

```

```

70     */
71     @Override
72     protected void doGet(HttpServletRequest request ,
73         HttpServletResponse
74         response)
75         throws ServletException , IOException {
76         processRequest(request , response);
77     }
78     /**
79     * Handles the HTTP <code>POST</code> method.
80     * @param request servlet request
81     * @param response servlet response
82     * @throws ServletException if a servlet-specific error
83     * occurs
84     * @throws IOException if an I/O error occurs
85     */
86     @Override
87     protected void doPost(HttpServletRequest request ,
88         HttpServletResponse
89         response)
90         throws ServletException , IOException {
91         processRequest(request , response);
92     }
93     /**
94     * Returns a short description of the servlet.
95     * @return a String containing servlet description
96     */
97     @Override
98     public String getServletInfo() {
99         return "Short description";
100     } // </editor-fold>
101     private void verCarreras(HttpServletRequest request ,
102         HttpServletResponse
103         response)
104         throws ServletException , IOException {
105         try {
106             request.setAttribute("PAGINA", Paginas.
107                 MOSTRAR_CARRERAS);
108             CarreraDao dao = new CarreraDaoImpl();
109             List<Carrera> carreras = dao.readAll();

```

```

108         request.setAttribute("carreras", carreras);
109         request.getRequestDispatcher("listaCarreras.jsp
        ").forward(request,
110 response);
111     } catch (SQLException ex) {
112         Logger.getLogger(CarrerasServlet.class.getName
       ()).log(Level.SEVERE,
113 null, ex);
114     }
115 }
116
117 private void eliminarCarrera(HttpServletRequest request
        , HttpServletResponse
118 response) {
119     try {
120         int id = Integer.valueOf(request.getParameter("
        id"));
121         CarreraDao dao = new CarreraDaoImpl();
122         Carrera c = new Carrera();
123         c.setId(id);
124         dao.delete(c);
125         response.sendRedirect("carreras?action=ver");
126     } catch (IOException | SQLException ex) {
127         Logger.getLogger(CarrerasServlet.class.getName
       ()).log(Level.SEVERE,
128 null, ex);
129     }
130 }
131
132 private void editarCarrera(HttpServletRequest request,
        HttpServletResponse
133 response)
134     throws ServletException, IOException {
135     try {
136         CarreraDao dao = new CarreraDaoImpl();
137         int id = Integer.valueOf(request.getParameter("
        id"));
138         Carrera c = new Carrera();
139         c.setId(id);
140         c = dao.read(c);
141         request.setAttribute("carrera", c);
142         request.setAttribute("PAGINA", Paginas.
        MOSTRAR_CARRERAS);

```



```

143         request.getRequestDispatcher("formCarrera.jsp")
144         .forward(request,
145 response);
146     } catch (SQLException ex) {
147         Logger.getLogger(CarrerasServlet.class.getName
148         ()).log(Level.SEVERE,
149 null, ex);
150     }
151     private void agregarCarrera(HttpServletRequest request,
152 HttpServletResponse
153 response)
154         throws ServletException, IOException {
155         request.setAttribute("PAGINA", Paginas.
156 AGREGAR_CARRERA);
157         request.getRequestDispatcher("formCarrera.jsp").
158         forward(request,
159 response);
160     }
161     private void guardarCarrera(HttpServletRequest request,
162 HttpServletResponse
163 response)
164         throws IOException {
165         response.sendRedirect("carreras?action=ver");
166     }
167 }

```

Archivo 6: CarrerasServlet.java

```

1
2 package controller;
3
4 import java.io.IOException;
5 import javax.servlet.ServletException;
6 import javax.servlet.annotation.WebServlet;
7 import javax.servlet.http.HttpServlet;
8 import javax.servlet.http.HttpServletRequest;
9 import javax.servlet.http.HttpServletResponse;
10 import utils.Paginas;
11

```

```

12 /**
13  *
14  * @author tonatihu
15  * Created on 09-Mar-2019
16  */
17 @WebServlet(name="HomeServlet", urlPatterns={"/home"})
18 public class HomeServlet extends HttpServlet {
19
20     /**
21      * Processes requests for both HTTP <code>GET</code>
22      and <code>POST</code>
23      methods.
24      * @param request servlet request
25      * @param response servlet response
26      * @throws ServletException if a servlet-specific error
27      occurs
28      * @throws IOException if an I/O error occurs
29      */
30     protected void processRequest(HttpServletRequest
31     request,
32     HttpServletResponse response)
33     throws ServletException, IOException {
34         request.setAttribute("PAGINA", Paginas.HOME);
35         request.getRequestDispatcher("home.jsp").forward(
36         request, response);
37     }
38
39     // <editor-fold defaultstate="collapsed" desc="
40     HttpServlet methods. Click on
41     the + sign on the left to edit the code.">
42     /**
43      * Handles the HTTP <code>GET</code> method.
44      * @param request servlet request
45      * @param response servlet response
46      * @throws ServletException if a servlet-specific error
47      occurs
48      * @throws IOException if an I/O error occurs
49      */
50     @Override
51     protected void doGet(HttpServletRequest request,
52     HttpServletResponse
53     response)
54     throws ServletException, IOException {

```

```

48         processRequest(request, response);
49     }
50
51     /**
52      * Handles the HTTP <code>POST</code> method.
53      * @param request servlet request
54      * @param response servlet response
55      * @throws ServletException if a servlet-specific error
56      * occurs
57      * @throws IOException if an I/O error occurs
58      */
59     @Override
60     protected void doPost(HttpServletRequest request,
61                           HttpServletResponse
62                           response)
63         throws ServletException, IOException {
64         processRequest(request, response);
65     }
66
67     /**
68      * Returns a short description of the servlet.
69      * @return a String containing servlet description
70      */
71     @Override
72     public String getServletInfo() {
73         return "Short description";
74     } // </editor-fold>
75 }

```

Archivo 7: HomeServlet.java

```

1 package controller;
2
3 import dao.impl.UsuarioDaoImpl;
4 import java.io.IOException;
5 import java.sql.SQLException;
6 import java.util.logging.Level;
7 import java.util.logging.Logger;
8 import javax.servlet.ServletException;
9 import javax.servlet.annotation.WebServlet;
10 import javax.servlet.http.HttpServlet;
11 import javax.servlet.http.HttpServletRequest;

```

```

12 import javax.servlet.http.HttpServletResponse;
13 import dao.UsuarioDao;
14 import dto.Usuario;
15 import javax.servlet.http.HttpSession;
16
17 /**
18  *
19  * @author tonatihu
20  * Created on 09-Mar-2019
21  */
22 @WebServlet(name="Login", urlPatterns={"/login"})
23 public class LoginServlet extends HttpServlet {
24
25     /**
26      * Processes requests for both HTTP <code>GET</code>
27      and <code>POST</code>
28      methods.
29      * @param request servlet request
30      * @param response servlet response
31      * @throws ServletException if a servlet-specific error
32      occurs
33      * @throws IOException if an I/O error occurs
34      */
35     protected void processRequest(HttpServletRequest request,
36 HttpServletResponse response)
37     throws ServletException, IOException {
38         request.setCharacterEncoding("UTF-8");
39         String accion = request.getParameter("logout");
40         if (accion != null)
41             logout(request, response);
42         else {
43             if (request.getMethod().equals("POST"))
44                 login(request, response);
45             else
46                 request.getRequestDispatcher("login.jsp").
47                 forward(request,
48 response);
49         }
50
51         // <editor-fold defaultstate="collapsed" desc="
52         HttpServlet methods. Click on

```

```

50 the + sign on the left to edit the code.">
51 /**
52  * Handles the HTTP <code>GET</code> method.
53  * @param request servlet request
54  * @param response servlet response
55  * @throws ServletException if a servlet-specific error
    occurs
56  * @throws IOException if an I/O error occurs
57  */
58 @Override
59 protected void doGet(HttpServletRequest request,
    HttpServletResponse
60 response)
61     throws ServletException, IOException {
62     processRequest(request, response);
63 }
64
65 /**
66  * Handles the HTTP <code>POST</code> method.
67  * @param request servlet request
68  * @param response servlet response
69  * @throws ServletException if a servlet-specific error
    occurs
70  * @throws IOException if an I/O error occurs
71  */
72 @Override
73 protected void doPost(HttpServletRequest request,
    HttpServletResponse
74 response)
75     throws ServletException, IOException {
76     processRequest(request, response);
77 }
78
79 /**
80  * Returns a short description of the servlet.
81  * @return a String containing servlet description
82  */
83 @Override
84 public String getServletInfo() {
85     return "Short description";
86 } // </editor-fold>
87
88 private void logout(HttpServletRequest request,

```

```

        HttpServletResponse
89 response)
90         throws ServletException , ServletException ,
        IOException {
91         HttpSession session = request.getSession(false);
92         session.invalidate();
93         request.getRequestDispatcher("login.jsp").forward(
        request , response);
94     }
95
96     private void login(HttpServletRequest request ,
        HttpServletResponse
97 response)
98         throws IOException , ServletException {
99         String username = request.getParameter("username");
100        String password = request.getParameter("password");
101        try {
102            UsuarioDao dao = new UsuarioDaoImpl();
103            if (dao.existsByUsernameAndPassword(username ,
        password)) {
104                HttpSession session = request.getSession();
105                Usuario u = dao.findByUsername(username);
106                session.setAttribute("USUARIO_SESSION" , u);
107                response.sendRedirect("home");
108            } else {
109                request.getRequestDispatcher("login.jsp").
        forward(request ,
110 response);
111            }
112        } catch (SQLException ex) {
113            Logger.getLogger(LoginServlet.class.getName()).
        log(Level.SEVERE,
114 null , ex);
115            request.getRequestDispatcher("login.jsp").
        forward(request ,
116 response);
117        }
118    }
119 }

```

Archivo 8: LoginServlet.java

```

1 package controller;

```

```

2
3 import dao.AlumnoDao;
4 import dao.impl.AlumnoDaoImpl;
5 import dao.CarreraDao;
6 import dao.impl.CarreraDaoImpl;
7 import dao.ProfesorDao;
8 import dao.impl.ProfesorDaoImpl;
9 import dto.Alumno;
10 import dto.Carrera;
11 import dto.Profesor;
12 import dto.Usuario;
13 import java.io.IOException;
14 import java.sql.SQLException;
15 import java.util.List;
16 import java.util.logging.Level;
17 import java.util.logging.Logger;
18 import javax.servlet.ServletException;
19 import javax.servlet.annotation.WebServlet;
20 import javax.servlet.http.HttpServlet;
21 import javax.servlet.http.HttpServletRequest;
22 import javax.servlet.http.HttpServletResponse;
23 import javax.servlet.http.HttpSession;
24 import utils.Paginas;
25
26 /**
27  *
28  * @author tonatihu
29  * Created on 09-Mar-2019
30  */
31 @WebServlet(name="PerfilServlet", urlPatterns={"/perfil"})
32 public class PerfilServlet extends HttpServlet {
33
34     /**
35      * Processes requests for both HTTP GET
36      and POST
37      methods.
38      * @param request servlet request
39      * @param response servlet response
40      * @throws ServletException if a servlet-specific error
41      occurs
42      * @throws IOException if an I/O error occurs
43      */
44     protected void processRequest(HttpServletRequest

```

```

        request ,
43 HttpServletResponse response)
44     throws ServletException , IOException {
45         request.setCharacterEncoding("UTF-8");
46         HttpSession session = request.getSession();
47         Usuario usuario = (Usuario) session.getAttribute("
USUARIO_SESSION");
48         int tipo = usuario.getTipo();
49         request.setAttribute("PAGINA", Paginas.PERFIL);
50         if (tipo == 1) {
51             try {
52                 CarreraDao dao = new CarreraDaoImpl();
53                 AlumnoDao daoAlumno = new AlumnoDaoImpl();
54                 Alumno a = daoAlumno.findByUsername(usuario
.getUsername());
55                 List<Carrera> carreras = dao.readAll();
56                 request.setAttribute("carreras", carreras);
57                 request.setAttribute("alumno", a);
58
59 request.getRequestDispatcher("perfilAlumno.jsp").forward(
        request , response);
60             } catch (SQLException ex) {
61
62 Logger.getLogger( PerfilServlet.class.getName()).log( Level .
        SEVERE, null , ex);
63             }
64         } else {
65             try {
66                 ProfesorDao profesorDao = new
        ProfesorDaoImpl();
67                 Profesor profesor =
68 profesorDao.findByUsername(usuario.getUsername());
69                 request.setAttribute("profesor", profesor);
70
71 request.getRequestDispatcher("perfilProfesor.jsp").forward(
        request , response);
72             } catch (SQLException ex) {
73
74 Logger.getLogger( PerfilServlet.class.getName()).log( Level .
        SEVERE, null , ex);
75             }
76         }
77     }

```



```

78
79 // <editor-fold defaultstate="collapsed" desc="
    HttpServlet methods. Click on
80 the + sign on the left to edit the code.">
81 /**
82  * Handles the HTTP <code>GET</code> method.
83  * @param request servlet request
84  * @param response servlet response
85  * @throws ServletException if a servlet-specific error
    occurs
86  * @throws IOException if an I/O error occurs
87  */
88 @Override
89 protected void doGet(HttpServletRequest request,
    HttpServletResponse
90 response)
91     throws ServletException, IOException {
92     processRequest(request, response);
93 }
94
95 /**
96  * Handles the HTTP <code>POST</code> method.
97  * @param request servlet request
98  * @param response servlet response
99  * @throws ServletException if a servlet-specific error
    occurs
100  * @throws IOException if an I/O error occurs
101  */
102 @Override
103 protected void doPost(HttpServletRequest request,
    HttpServletResponse
104 response)
105     throws ServletException, IOException {
106     processRequest(request, response);
107 }
108
109 /**
110  * Returns a short description of the servlet.
111  * @return a String containing servlet description
112  */
113 @Override
114 public String getServletInfo() {
115     return "Short description";

```

```
116     }// </editor-fold>
117
118 }
```

Archivo 9: PerfilServlet.java

```
1 package controller;
2
3 import dao.UsuarioDao;
4 import dao.impl.UsuarioDaoImpl;
5 import dto.Alumno;
6 import dto.Profesor;
7 import java.io.IOException;
8 import java.sql.SQLException;
9 import java.util.logging.Level;
10 import java.util.logging.Logger;
11 import javax.servlet.ServletException;
12 import javax.servlet.annotation.WebServlet;
13 import javax.servlet.http.HttpServlet;
14 import javax.servlet.http.HttpServletRequest;
15 import javax.servlet.http.HttpServletResponse;
16
17 /**
18  *
19  * @author tonatihu
20  * Created on 09-Mar-2019
21  */
22 @WebServlet(name="Signup", urlPatterns={"/signup"})
23 public class SignupServlet extends HttpServlet {
24
25     /**
26      * Processes requests for both HTTP <code>GET</code>
27      and <code>POST</code>
28      methods.
29      * @param request servlet request
30      * @param response servlet response
31      * @throws ServletException if a servlet-specific error
32      occurs
33      * @throws IOException if an I/O error occurs
34      */
35     protected void processRequest(HttpServletRequest
36     request ,
37     HttpServletResponse response)
```

```

35     throws ServletException , IOException {
36         request.setCharacterEncoding("UTF-8");
37         if (request.getMethod().equals("POST")) {
38             String nombre = request.getParameter("nombre");
39             String apPaterno = request.getParameter("
apPaterno");
40             String apMaterno = request.getParameter("
apMaterno");
41             String email = request.getParameter("email");
42             int tipo = Integer.valueOf(request.getParameter
("tipo"));
43             String username = request.getParameter("
username");
44             String contra = request.getParameter("password"
);
45             String boleta = request.getParameter("boleta");
46             UsuarioDao userDao = new UsuarioDaoImpl();
47             try {
48                 if (tipo == 1) {
49                     Alumno a = new Alumno();
50                     a.setApMaterno(apMaterno);
51                     a.setApPaterno(apPaterno);
52                     a.setBoleta(boleta);
53                     a.setEmail(email);
54                     a.setNombre(nombre);
55                     a.setTipo(tipo);
56                     a.setUsername(username);
57                     a.setPassword(contra);
58                     userDao.create(a);
59
60                 } else {
61                     Profesor p = new Profesor();
62                     p.setApMaterno(apMaterno);
63                     p.setApPaterno(apPaterno);
64                     p.setNumeroProfesor(boleta);
65                     p.setEmail(email);
66                     p.setNombre(nombre);
67                     p.setTipo(tipo);
68                     p.setUsername(username);
69                     p.setPassword(contra);
70                     userDao.create(p);
71                 }
72             } catch (Exception e) {

```

```

73         return;
74     } catch (SQLException ex) {
75
76         Logger.getLogger(SignupServlet.class.getName()).log(Level.
            SEVERE, null, ex);
77     }
78 }
79
80     request.getRequestDispatcher("signup.jsp").forward(
        request, response);
81 }
82
83     // <editor-fold defaultstate="collapsed" desc="
HttpServlet methods. Click on
84 the + sign on the left to edit the code.">
85     /**
86      * Handles the HTTP <code>GET</code> method.
87      * @param request servlet request
88      * @param response servlet response
89      * @throws ServletException if a servlet-specific error
        occurs
90      * @throws IOException if an I/O error occurs
91      */
92     @Override
93     protected void doGet(HttpServletRequest request,
        HttpServletResponse
94 response)
95     throws ServletException, IOException {
96         processRequest(request, response);
97     }
98
99     /**
100      * Handles the HTTP <code>POST</code> method.
101      * @param request servlet request
102      * @param response servlet response
103      * @throws ServletException if a servlet-specific error
        occurs
104      * @throws IOException if an I/O error occurs
105      */
106     @Override
107     protected void doPost(HttpServletRequest request,
        HttpServletResponse
108 response)

```

```

109     throws ServletException , IOException {
110         processRequest(request , response);
111     }
112
113     /**
114     * Returns a short description of the servlet.
115     * @return a String containing servlet description
116     */
117     @Override
118     public String getServletInfo() {
119         return "Short description";
120     }// </editor-fold>
121
122 }

```

Archivo 10: SignupServlet.java

2.2. Pruebas

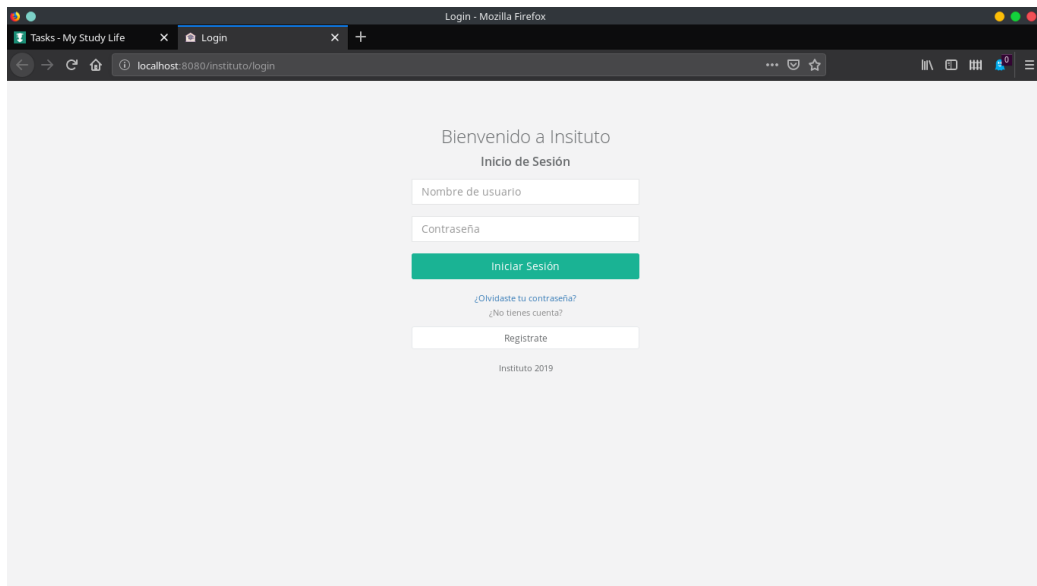


Figura 2: Login

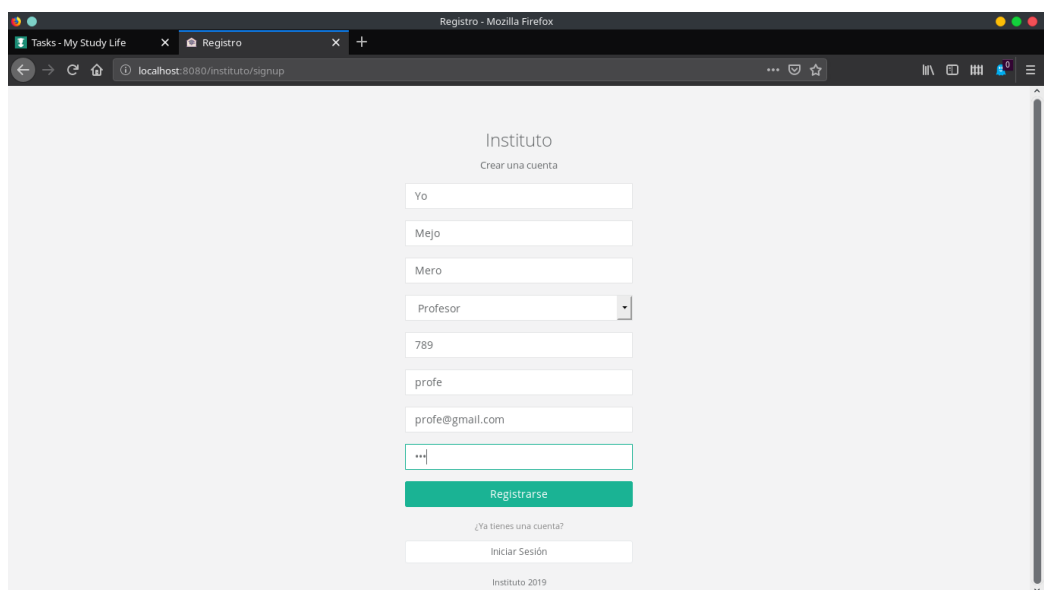


Figura 3: Crear Cuenta

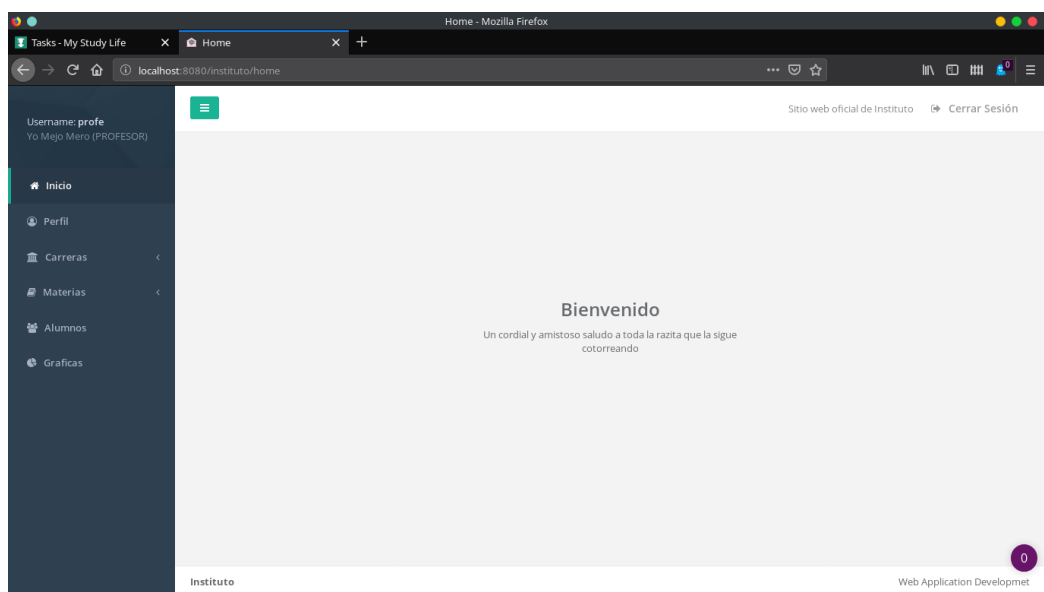


Figura 4: Home

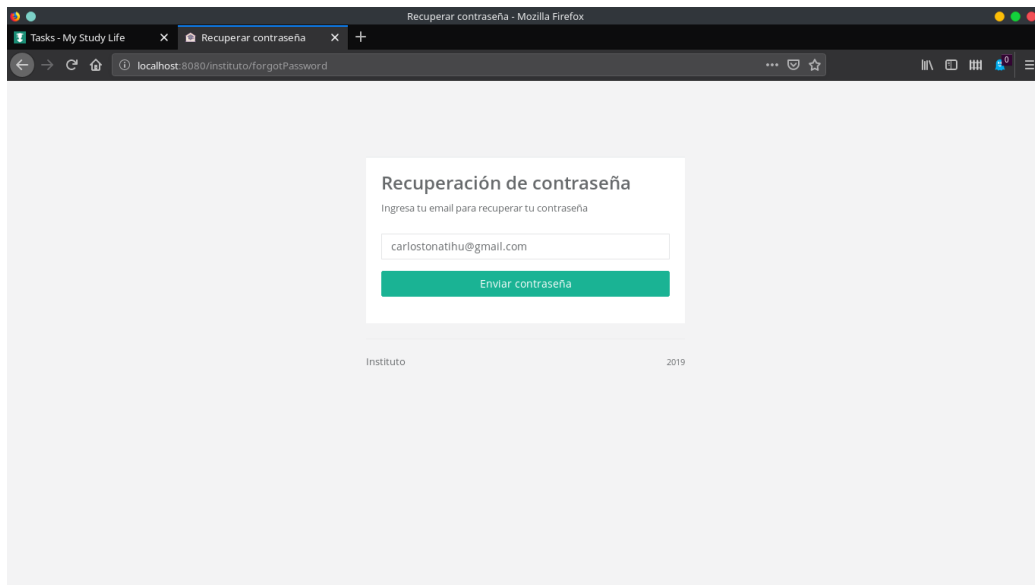


Figura 5: Recuperar contraseña

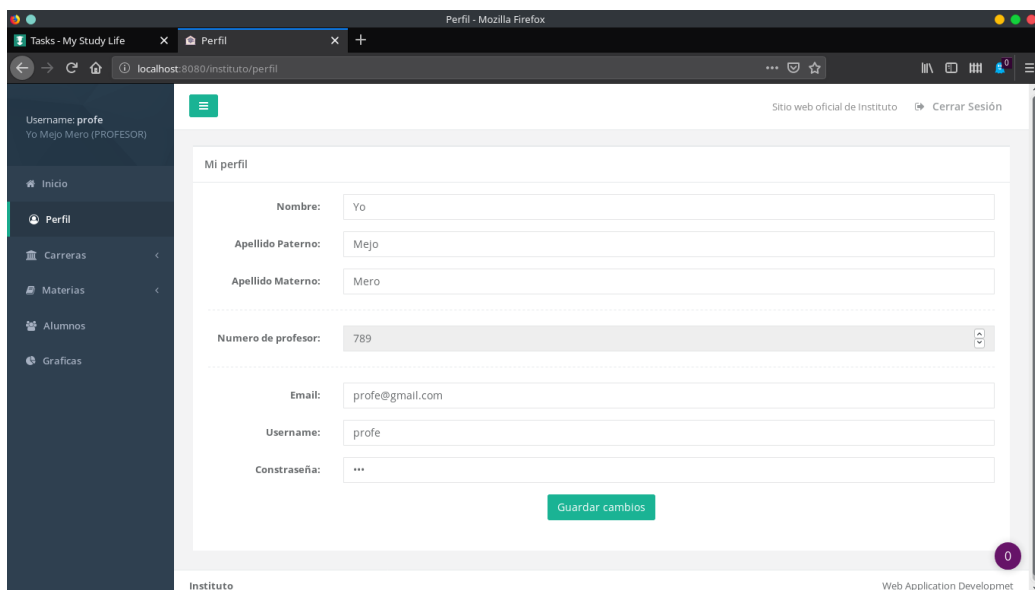


Figura 6: Perfil

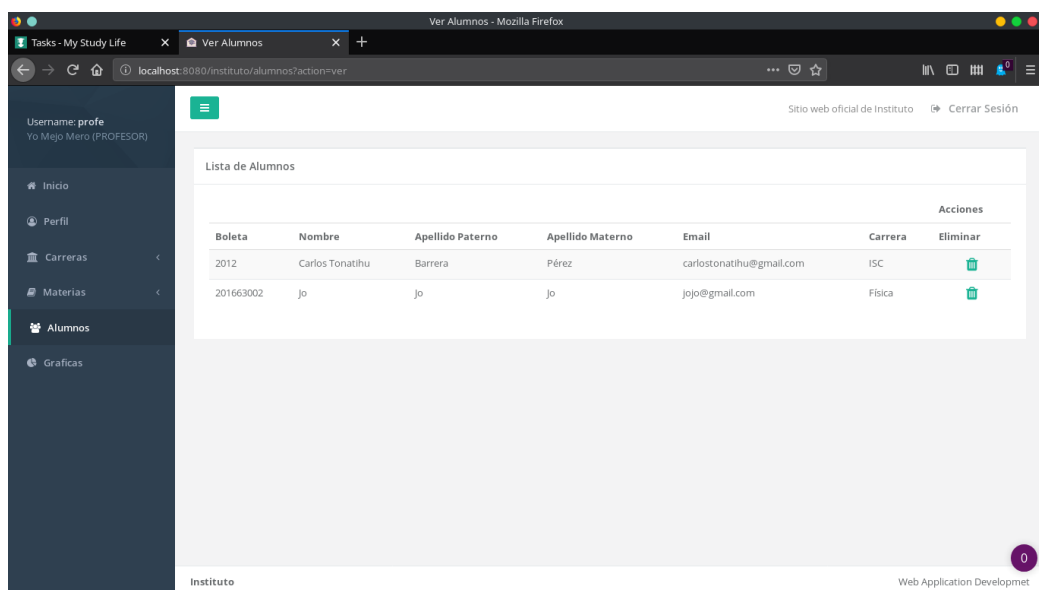


Figura 7: Alumnos

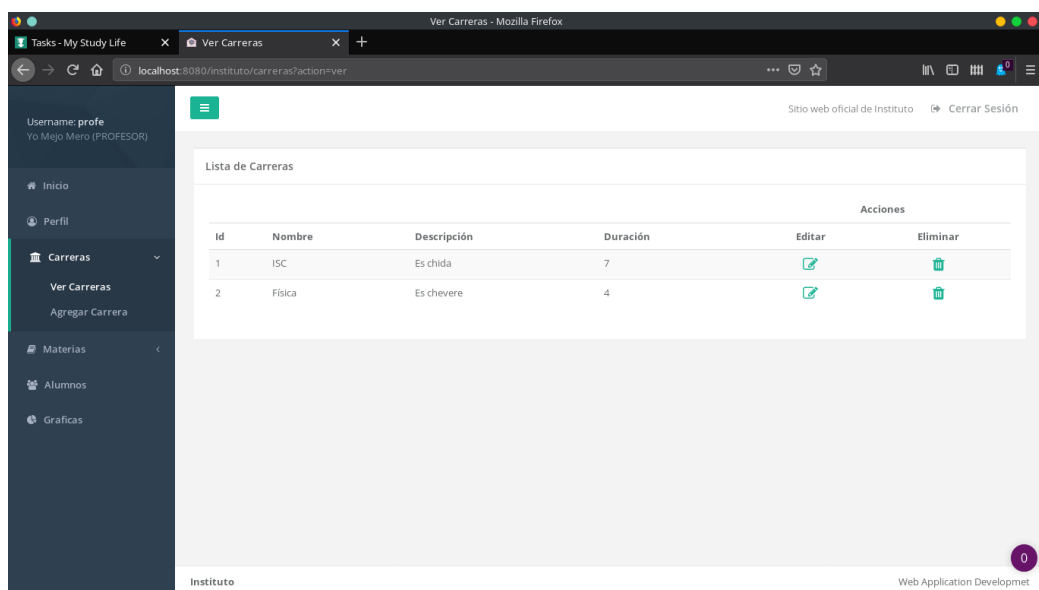


Figura 8: Ver carreras

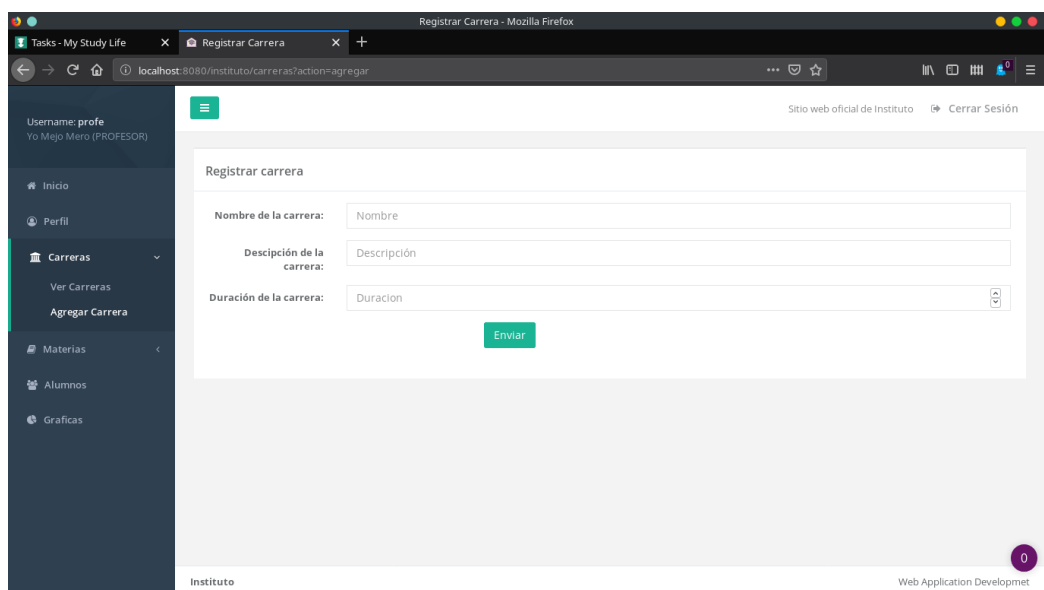


Figura 9: Agregar Carrera

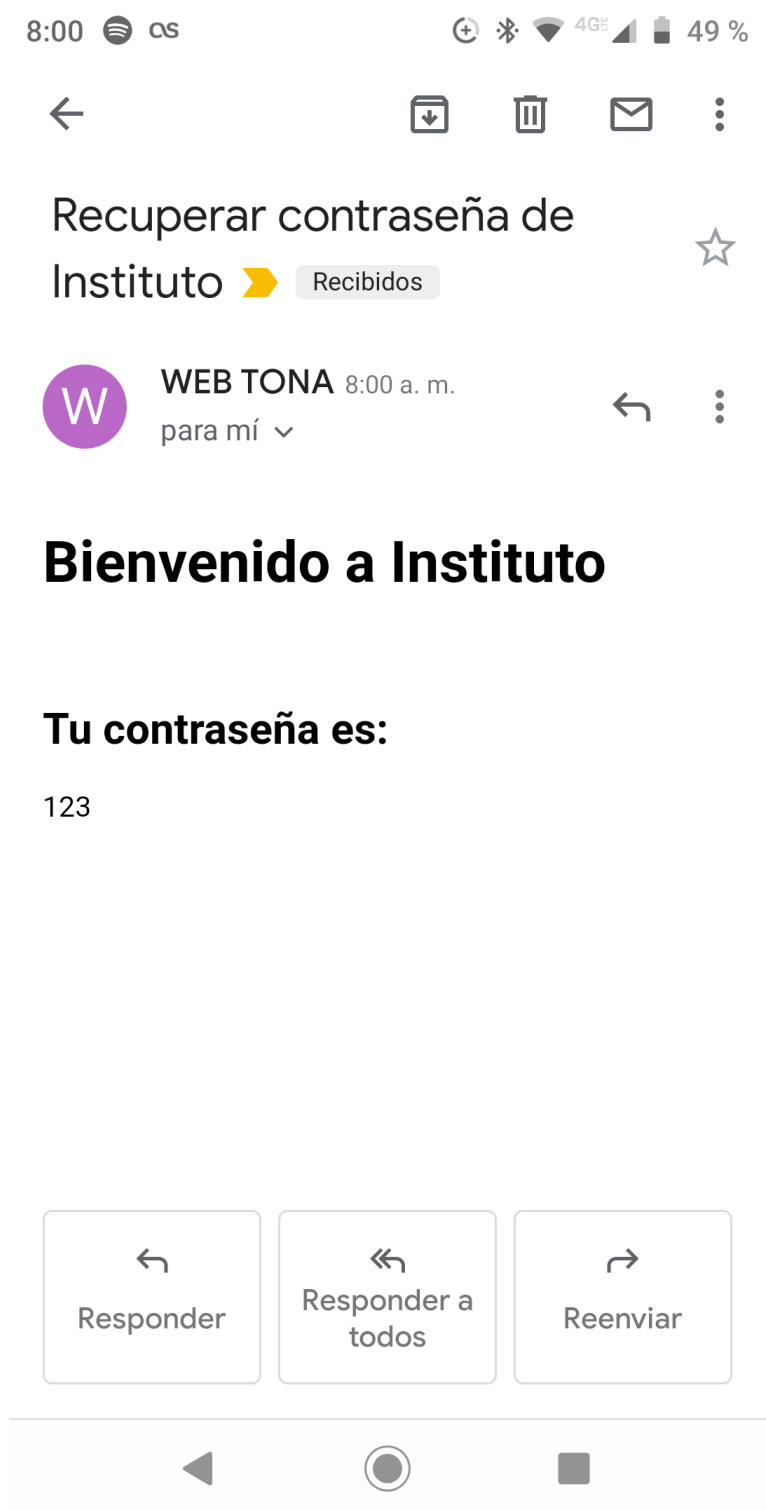


Figura 10: Correo de recuperación de contraseña

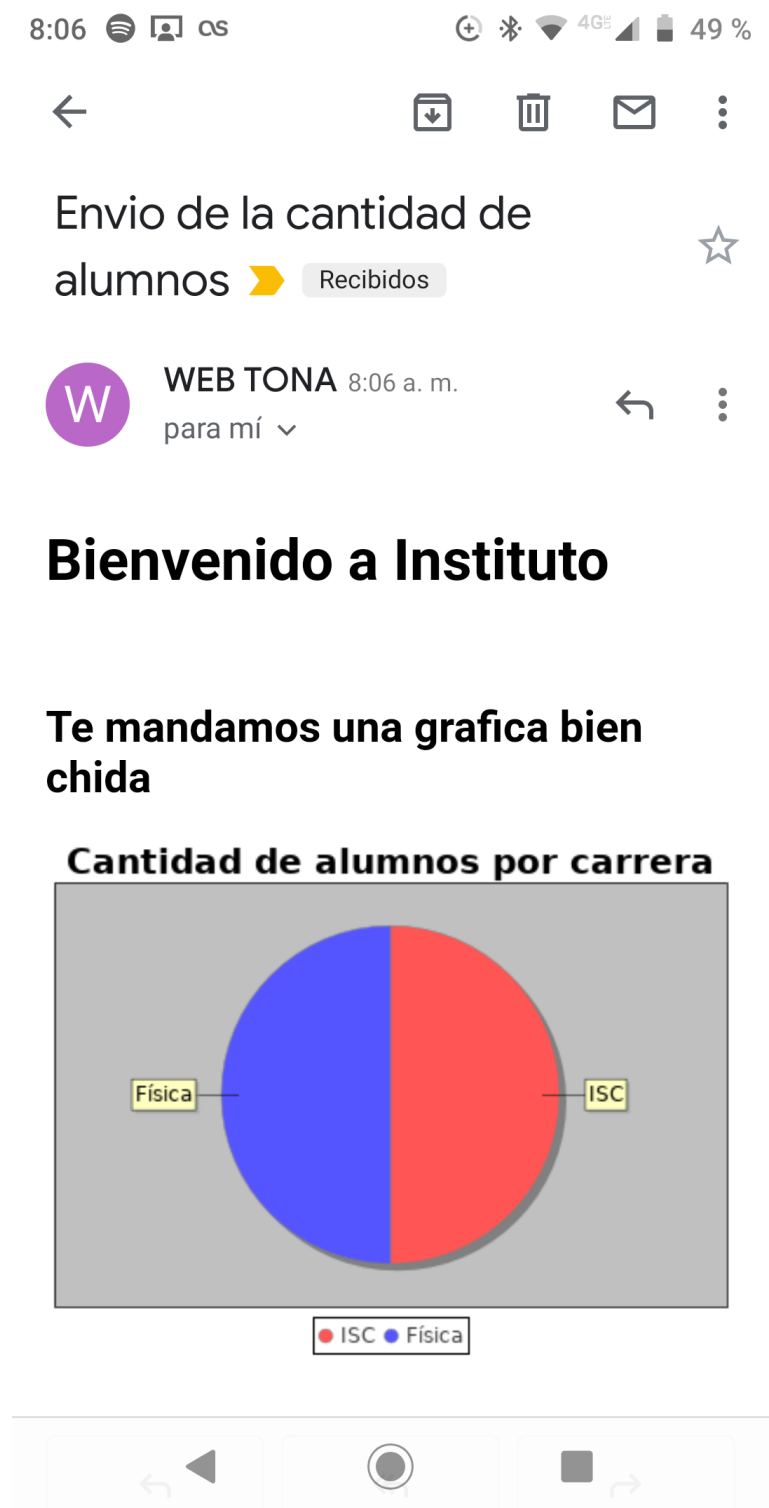


Figura 11: Correo de envió de gráficas

3. Conclusiones

Esta practica resulto difícil en un inicio debido a la nueva funcionalidad de envío de correos y generación de gráficas. Sin embargo, después de que se investigo un poco al respecto se pudo lograr el objetivo que se planteo en un inicio. Así que en un futuro la implementación de este tipo de características sera más sencillo de hacer, he incluso se puede mejorar.