

ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΕΙΡΑΙΩΣ

Τμήμα Πληροφορικής



Εργασία Μαθήματος «Πληροφοριακά Συστήματα στο Διαδίκτυο»

Απαλλακτική Εργασία	Ανάπτυξη Διαδικτυακού Πληροφοριακού Συστήματος
Όνομα φοιτητή – Αρ. Μητρώου	Ρούντου Άννα – Φανή - Π17113
	Σαγιέντ Ιωσήφ – Π15123
Ημερομηνία παράδοσης	28/09/2021

Εκφώνηση Άσκησης:

Η εργασία είναι υποχρεωτική και έχει βαρύτητα 100% επί της συνολικής βαθμολογίας.

Στην εργασία η κάθε ομάδα, η οποία θα αποτελείται από δύο άτομα το πολύ, θα πρέπει να κατασκευάσει μια web εφαρμογή με JSP/Servlets (εναλλακτικά με C# .net). Ειδικότερα, εκτός των βασικών λειτουργιών που περιγράφονται παρακάτω, θα πρέπει να αναπτύξετε μια αλγοριθμική διαδικασία που θα βρίσκεται στον πυρήνα της εφαρμογής.

Βασικές λειτουργίες:

1. Δημιουργία βάσης δεδομένων.
2. Δημιουργία χρηστών.
3. Διαχείριση χρηστών και κανόνες ελέγχου πρόσβασης. Διαφορετικές κατηγορίες χρηστών θα πρέπει να έχουν και διαφορετικά δικαιώματα πρόσβασης σε συγκεκριμένα αρχεία και δεδομένα.
4. Παρουσίαση και επεξεργασία των δεδομένων που βρίσκονται αποθηκευμένα στη βάση.

Αλγοριθμική διαδικασία

Η εφαρμογή που θα αναπτύξετε θα πρέπει να προσφέρει στους χρήστες της χρήσιμα συμπεράσματα και πληροφορίες. Ένα παράδειγμα αποτελεί μια εφαρμογή που αποφασίζει αν θα πρέπει κάποιος πελάτης να λάβει μια χρηματοπιστωτική υπηρεσία. Σε αυτήν την περίπτωση, η απόφαση θα ληφθεί βάσει δεδομένων που εισάγει ο χρήστης, που βρίσκονται στη βάση αλλά και δεδομένων που ανακτώνται μέσω web services από άλλους οργανισμούς. Συνεπώς, στο πλαίσιο της παραπάνω εφαρμογής θα πρέπει να αναπτύξετε την προαναφερθείσα αλγοριθμική διαδικασία που οδηγεί στη συγκεκριμένη απόφαση. Περισσότερα και αναλυτικότερα παραδείγματα παρουσιάζονται στο μάθημα.

Θέμα Εργασίας

Οδηγός επιλογής και εύρεσης κοκτέιλ, σύμφωνα με τις προτιμήσεις και την τοποθεσία του χρήστη. Σκοπός μας είναι να αναπτύξουμε μια ιστοσελίδα η οποία θα έχει καταχωρημένα σε μια βάση δεδομένων όλα τα είδη των ποτών καθώς και διάφορα κοκτέιλ και ο χρήστης αφού συμπληρώσει τις προτιμήσεις του ή/και τυχόν αλλεργίες, θα έχει την επιλογή είτε να αναζητήσει κάποιο μαγαζί πιθανόν στην περιοχή του(ή σε οποιαδήποτε περιοχή θέλει) από το οποίο θα μπορεί να το αγοράσει, είτε να εμφανιστεί μια ενδεικτική συνταγή ώστε να το φτιάξει ο ίδιος.

ΠΕΡΙΕΧΟΜΕΝΑ

Εγκατάσταση και παραμετροποίηση application server και database server.....	4
Application Server	4
Database Server	4
Μοντέλο 3-Tier	4
Δημιουργία Βάσης Δεδομένων	5
Πίνακας «recipes»	5
Πίνακας «users»	6
Δημιουργία web project.....	7
Πακέτα κλάσεων	7
1. Κλάση «Cocktails.java».....	8
2. Κλάση «User.java».....	9
3. Κλάση «LoginServlet.java».....	11
4. Κλάση «LogoutServlet.java».....	12
5. Κλάση «RegisterUserServlet.java».....	13
6. Κλάση «CocktailServlet.java».....	15
7. Κλάση «CocktailServlet2.java».....	16
8. Κλάση «AddCocktailServlet.java».....	17
9. Κλάση «AdminDrinkServlet.java».....	18
10. Κλάση «DeleteCocktailServlet.java».....	19
11. Κλάση «DrinkServlet.java».....	20
12. Κλάση «DatabaseProcedures.java».....	21
JSPs-HTML	28
Δημιουργία διαδικτυακής διεπαφής.....	29
Κύρια διαδικτυακή διεπαφή	29
Επιτυχής σύνδεση απλού χρήστη	31
Επιτυχής σύνδεση ως Admin	36
Χρήση των Google APIs.....	39
Google Maps API	39
Google Sign-In API	41

Εγκατάσταση και παραμετροποίηση application server και database server

Application Server

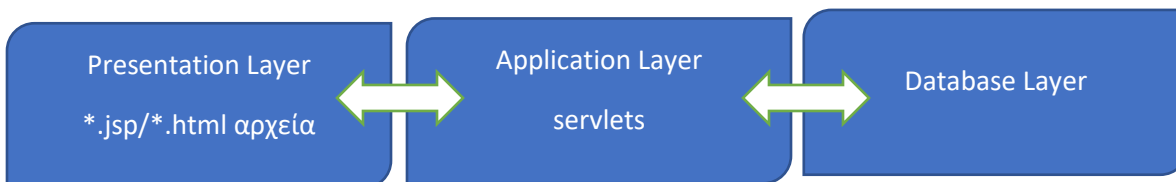
Στην εργασία χρησιμοποιήθηκε ο application server «Tomcat v9», όπου και συνδέθηκε με το προγραμματιστικό περιβάλλον IDE «IntelliJIDEA».

Database Server

Στην εργασία χρησιμοποιήθηκε το Σύστημα Διαχείρισης Βάσης Δεδομένων «postgresql» και συνδέθηκε με τον αντίστοιχο jdbc database connector «PostgreSQL JDBC Driver» με το application server.

Μοντέλο 3-Tier

Χρησιμοποιήσαμε το μοντέλο 3-Tier. Στο πρώτο επίπεδο Presentation layer υπάρχει η αλληλεπίδραση με τον χρήστη (*.jsp, *.html αρχεία). Παίρνει τα δεδομένα και τα στέλνει στο δεύτερο επίπεδο, το Application layer. Εκεί, γίνεται η επικοινωνία με τη βάση δεδομένων. Δρα δηλαδή ως ο μεσολαβητής μεταξύ του Presentation layer και του τρίτου επιπέδου Database layer στο οποίο είναι αποθηκευμένα τα δεδομένα. Γίνεται η σύνδεση με την βάση και εκτελούνται απαιτούμενες ενέργειες (αρχείο Database).



Δημιουργία Βάσης Δεδομένων

Πίνακας «recipes»

Περιέχει τα χαρακτηριστικά των cocktails μέσω των οποίων γίνονται αναζητήσεις στην εφαρμογή αλλά και εισαγωγές cocktail στην βάση (name, base, taste, ingredients, link, image).

Cocktails/postgres@PostgreSQL 13						
Data Output						
	name [PK] character varying (50)	base character varying (30)	taste character varying (50)	ingredients character varying (50)	link text	image bytea
1	Bees Knees	Gin	Sweet&Spicy	NO	https://www.allrecipes.com/recipe/263068/julis-bees-knees/	[binary data]
2	Bloody Mary	Vodka	Spicy	NO	https://www.myrecipes.com/recipe/bloody-mary	[binary data]
3	Caribbean Rum Punch	Rum	Sweet&Spicy	NO	https://www.allrecipes.com/recipe/32349/caribbean-rum-punch/	[binary data]
4	Cosmopolitan	Vodka	Sweet	NO	https://www.myrecipes.com/recipe/cosmopolitan	[binary data]
5	Daiquiri	Rum	Sweet&Sour	NO	https://www.myrecipes.com/recipe/classic-daiquiri	[binary data]
6	Gin and Tonic	Gin	Sour	NO	https://www.myrecipes.com/recipe/gin-tonic	[binary data]
7	Gin Fizz	Gin	Sweet	NO	https://www.myrecipes.com/recipe/ramos-gin-fizz-0	[binary data]
8	Hot Toddy	Whiskey	Sweet	NO	https://www.myrecipes.com/recipe/hot-toddy	[binary data]
9	Manhattan	Whiskey	Sweet&Spicy	NO	https://www.myrecipes.com/recipe/classic-manhattan	[binary data]
10	Mojito	Rum	Sweet	NO	https://www.myrecipes.com/recipe/mojito-2	[binary data]
11	Moscow Mule	Vodka	Sweet&Spicy	NO	https://www.myrecipes.com/recipe/moscow-mule	[binary data]

Κώδικας δημιουργίας πίνακα:

```
CREATE TABLE Recipes (  
    name VARCHAR (50) PRIMARY KEY NOT NULL,  
    base VARCHAR (30) NOT NULL,  
    taste VARCHAR (50) NOT NULL,  
    ingredients VARCHAR (50),  
    link text,  
    image bytea  
);
```

Πίνακας «users»

Περιέχει τα στοιχεία των χρηστών του συστήματος καθώς και τον ρόλο τους (admin/user) και χρειάζεται για την σύνδεση και για επιμέρους λειτουργίες.

Cocktails/postgres@PostgreSQL 13					
Data Output					
	username character varying (50)	email [PK] character varying (100)	password character varying (50)	dateofbirth character varying (20)	category character varying (50)
1	annar17	annarountou@gmail.com	anna123	1999-12-17	User
2	iosif1	iosif@gmail.com	iosif123	1997-10-27	Admin

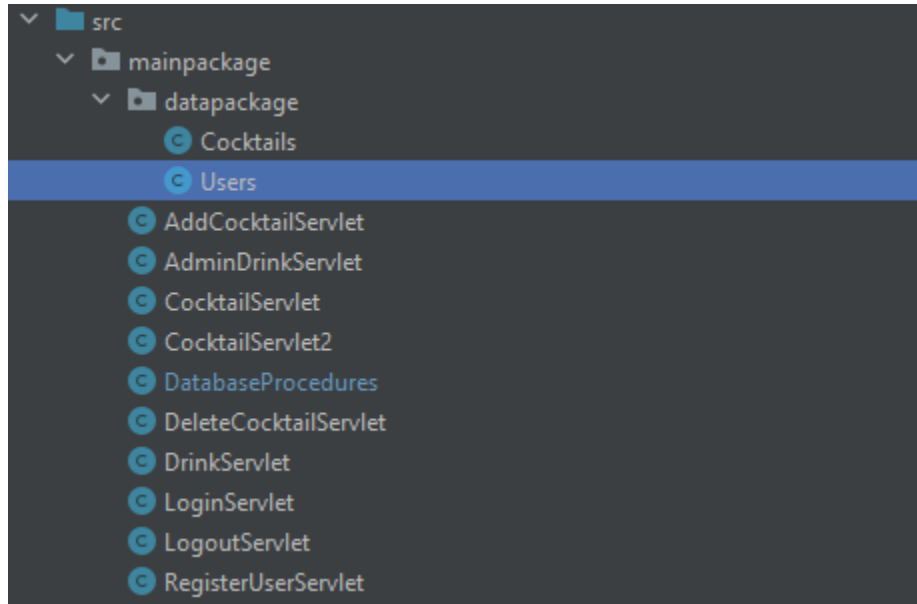
Κώδικας δημιουργίας πίνακα:

```
CREATE TABLE Users (  
    username VARCHAR (50),  
    email VARCHAR (100) PRIMARY KEY,  
    password VARCHAR (50),  
    dateOfBirth VARCHAR (20),  
    category VARCHAR (50)  
);
```

Δημιουργία web project

Πακέτα κλάσεων

- Αρχικά, έχουμε το πακέτο «mainpackage» το οποίο περιέχει όλες τις κλάσεις. Περιέχονται όλα τα Servlets τα οποία της εφαρμογής καθώς και τις κλάσεις για την λειτουργία της. Εντός του «mainpackage» περιέχεται το datapackage.
- Όπως αναφέρθηκε παραπάνω, εντός του «mainpackage» περιέχεται το datapackage στο οποίο δημιουργούνται τα βασικά στοιχεία των cocktails και των χρηστών.



1. Κλάση «Cocktails.java».

Σε αυτή την κλάση δημιουργούνται τα βασικά αντικείμενα με τα χαρακτηριστικά των cocktails.

```
Cocktails.java x
1  package mainpackage.datapackage;
2
3  public class Cocktails implements Comparable<Cocktails> {
4
5      private String name, base, taste, ingredients, link;
6      private String image;
7
8      public Cocktails(String base, String taste, String ingredients) {
9          this.base = base;
10         this.taste = taste;
11         this.ingredients = ingredients;
12     }
13
14     public Cocktails(String name, String base, String taste, String ingredients, String link, String image) { //
15         this.name = name;
16         this.base = base;
17         this.taste = taste;
18         this.ingredients = ingredients;
19         this.link = link;
20         this.image = image;
21     }
22
23     public Cocktails(String base) { this.base = base; }
24
25     public String getName() { return name; }
26
27     public void setName(String name) { this.name = name; }
28
29     public String getBase() { return base; }
30
31     public void setBase(String base) { this.base = base; }
32
33     public String getTaste() { return taste; }
34
35     public void setTaste(String taste) { this.taste = taste; }
36
37     public String getIngredients() { return ingredients; }
38
39     public void setIngredients(String ingredients) { this.ingredients = ingredients; }
40
41     public String getLink() { return link; }
```


2. Κλάση «User.java».

Στην κλάση αυτή δημιουργούνται τα αντικείμενα για τα στοιχεία του χρήστη. Δημιουργούνται αντικείμενα για το όνομα χρήστη, τον κωδικό, το email, την ημερομηνία γέννησης και την κατηγορία χρήστη καθώς και ένα counter το οποίο μετράει πόσοι χρήστες έχουν δημιουργηθεί.

```
Users.java x
1 package mainpackage.datapackage;
2
3 public class Users {
4
5     private String username, email, password, dateOfBirth, category;
6     static int usersCounter;
7
8     public Users(String username, String email, String password, String dateOfBirth, String category) {
9         this.username = username;
10        this.email = email;
11        this.password = password;
12        this.dateOfBirth = dateOfBirth;
13        this.category = category;
14        usersCounter++;
15    }
16
17    public Users(String category) {
18        usersCounter++;
19        this.category = category;
20    };
21
22    public Users() {
23        usersCounter++;
24        System.out.println("User has been created successfully!");
25    }
26
27    //Getters
28    public String getUsername() { return username; }
29
30    public String getEmail() { return email; }
31
32    public String getPassword() { return password; }
33
34    public String getDateOfBirth() { return dateOfBirth; }
35
36    public String getCategory() { return category; }
37
38    public int getUsersCounter() { return usersCounter; }
```

Επίσης τα παραπάνω αντικείμενα λαμβάνουν τιμές, ενώ στη συνέχεια εμφανίζονται τα κατάλληλα μηνύματα για κάθε περίπτωση αλληλεπίδρασης του χρήστη με την εφαρμογή(είσοδος/αποσύνδεση/εγγραφή). Τέλος, θέτουμε περιορισμό για την μορφή με την οποία θα γίνεται δεκτό το email των χρηστών.

```
//Setters
public void setUsername(String username) { this.username = username; }

public void setEmail(String email) { this.email = email; }

public void setPassword(String password) { this.password = password; }

public void setDateOfBirth(String dateOfBirth) { this.dateOfBirth = dateOfBirth; }

public void setCategory(String category) { this.category = "User"; }

//Login
public void Login() { System.out.println("You have logged in successfully!"); }

//Register
public void Register() { System.out.println("Your registration is completed successfully!"); }

//Logout
public void Logout() { System.out.print("You have logged out!"); }

public boolean isValidEmail(String email) {
    String pattern = "^[A-Za-z][A-Z0-9a-z._%+-]+@[A-Za-z0-9.-]+\\.[A-Za-z]{2,6}$";
    try { return email.matches(pattern); }
    catch (NullPointerException e) { return false; }
}
```

3. Κλάση «LoginServlet.java».

Στην παρακάτω κλάση πραγματοποιείται «hash» του κωδικού πρόσβασης. Επίσης διακρίνονται περιπτώσεις.

- Πρώτη περίπτωση αποτελεί η επιτυχημένη σύνδεση στην εφαρμογή ως απλός χρήστης.
- Δεύτερη περίπτωση αποτελεί η επιτυχής σύνδεση ως διαχειριστής (Admin) και τέλος,
- Η αποτυχημένη σύνδεση.

```
59  * @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse response)
60  */
61  @protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
62      String email = request.getParameter("Email");
63      /*SecureRandom random = new SecureRandom();
64      byte bytes[] = new byte[20];
65      random.nextBytes(bytes);
66      String password = getHash(request.getParameter("Password"), random.toString());*/
67      String password = request.getParameter("Password");
68
69      HttpSession session = request.getSession();
70
71      DatabaseProcedures dP = new DatabaseProcedures();
72      response.setContentType("text/plain");
73
74      if (dP.loginValidation(email, password)) {
75          session.setAttribute("Email", email);
76          String username = dP.getUsername(email);
77          session.setAttribute("Username", username);
78          String category = dP.getUserCategory(email);
79          session.setAttribute("Category", category);
80
81          if (category.equals("User")) {
82              response.getWriter().print("Login Successful!");
83              response.setHeader("Refresh", "3;url=mainIndex.jsp");
84          }
85          else if (category.equals("Admin")) {
86              response.getWriter().print("Login Successful!");
87              response.setHeader("Refresh", "3;url=adminsIndex.jsp");
88          }
89      } else {
90          response.getWriter().print("Login Failed!");
91          response.setHeader("Refresh", "3;url=login.jsp");
92      }
93  }
94
95  }
```

4. Κλάση «LogoutServlet.java».

Στην παρακάτω κλάση φαίνεται η διαδικασία αποσύνδεσης χρήστη κατά την οποία θα εμφανιστεί μία κενή σελίδα μαζί με το μήνυμα αποσύνδεσης, ενώ στην συνέχεια θα πραγματοποιηθεί ανανέωση της σελίδα και επιστροφή στην αρχική.

```
LogoutServlet.java
1  package mainpackage;
2
3  import ...
4
10
11  /**
12   * Servlet implementation class LogoutServlet
13   */
14  @WebServlet("/LogoutServlet")
15  public class LogoutServlet extends HttpServlet {
16      private static final long serialVersionUID = 1L;
17
18      /**
19       * @see HttpServlet#HttpServlet()
20       */
21      public LogoutServlet() {
22          super();
23          // TODO Auto-generated constructor stub
24      }
25
26      /**
27       * @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)
28       */
29      @Override
30      protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
31          HttpSession session = request.getSession(false);
32
33          session.invalidate();
34
35          response.setContentType("text/plain");
36          response.getWriter().print("Logged out successfully!");
37          response.setHeader("Refresh", "3;url=login.jsp");
38      }
39
40      /**
41       * @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse response)
42       */
43      @Override
44      protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
45          // TODO Auto-generated method stub
46          doGet(request, response);
47      }
48  }
```

5. Κλάση «RegisterUserServlet.java».

Στην παρακάτω κλάση λαμβάνονται οι τιμές των στοιχείων του χρήστη που εισάγει κατά την εγγραφή του. Ο κωδικός πρόσβασης γίνεται hashed ενώ στην συνέχεια πραγματοποιείται έλεγχος ηλικίας.

- Εάν ο χρήστης είναι άνω των 18 ετών, δηλαδή έχει γεννηθεί πριν το 2003, γίνεται ανανέωση της σελίδας και μεταφέρεται στην κύρια σελίδα της εφαρμογής.
- Στην περίπτωση που είναι μικρότερος των 18, εμφανίζεται το μήνυμα αποτυχίας εγγραφής και πραγματοποιείται ανανέωση της σελίδας.

```
RegisterUserServlet.java
1  package mainpackage;
2
3  import ...
4
18
19  /**
20   * Servlet implementation class RegisterUserServlet
21   */
22  @WebServlet("/RegisterUserServlet")
23  public class RegisterUserServlet extends HttpServlet {
24      private static final long serialVersionUID = 1L;
25
26      public String getHash(String unhashed, String salt) {
27          // Hash the password.
28
29          final String toHash = salt + unhashed + salt;
30          MessageDigest messageDigest = null;
31          try {
32              messageDigest = MessageDigest.getInstance("MD5");
33          } catch (NoSuchAlgorithmException ex) {
34              return "00000000000000000000000000000000";
35          }
36          messageDigest.update(toHash.getBytes(), 0, toHash.length());
37          String hashed = new BigInteger(1, messageDigest.digest()).toString(16);
38          if (hashed.length() < 32) {
39              hashed = "0" + hashed;
40          }
41          return hashed.toUpperCase();
42      }
43
44      /**
45       * @see HttpServlet#HttpServlet()
46       */
47      public RegisterUserServlet() {
48          super();
49          // TODO Auto-generated constructor stub
50      }
51
52      /**
53       * @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)
54       */
55  }
```

```

57     response.getWriter().append("Served at: ").append(request.getContextPath());
58 }
59
60 /**
61  * @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse response)
62  */
63 @protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
64     String username = request.getParameter("Username");
65     String email = request.getParameter("Email");
66     /*SecureRandom random = new SecureRandom();
67     byte bytes[] = new byte[20];
68     random.nextBytes(bytes);
69     String password = getHash(request.getParameter("Password"), random.toString());*/
70     String password = request.getParameter("Password");
71     String dateofbirth = request.getParameter("Birthday");
72     String category = "User";
73
74     String[] str = dateofbirth.split(" ", 2);
75     int date = parseInt(str[0]);
76
77     HttpSession session = request.getSession(true);
78     DatabaseProcedures dP = new DatabaseProcedures();
79
80     // Check if user is over 18
81     if (date < 2003) {
82         Users user = new Users(username, email, password, dateofbirth, category);
83         String result = dP.insertUser(user);
84         response.setContentType("text/plain");
85         response.getWriter().print(result);
86         response.setHeader("Refresh", "3;url=mainIndex.jsp");
87     } else {
88         response.getWriter().print("Registration failed! You must be over 18 years old! :( ");
89         response.setHeader("Refresh", "3;url=register.jsp");
90     }
91 }
92 }

```

6. Κλάση «CocktailServlet.java».

Η κλάση αυτή καλείται όταν ο χρήστης, συνδεδεμένος ως "User", επιλέγει ένα κοκτέιλ από την λίστα. Τρέχει η συνάρτηση που ψάχνει το αντίστοιχο κοκτέιλ στην βάση δεδομένων και μόλις το βρει το στέλνει στην cocktail_page.jsp.

```
CocktailServlet.java
20
21  /**
22   * @see HttpServlet#HttpServlet()
23   */
24  public CocktailServlet() {
25      super();
26      // TODO Auto-generated constructor stub
27  }
28
29  /**
30   * @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)
31   */
32  @protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
33      // TODO Auto-generated method stub
34      HttpSession session = request.getSession( b: true);
35      DatabaseProcedures dP = new DatabaseProcedures();
36
37      String base = request.getParameter( s: "Base");
38      String taste = request.getParameter( s: "Taste");
39      String ingredients = "";
40
41      Cocktails cocktailsList = dP.getCocktail(base, taste, ingredients);
42
43      request.setAttribute( s: "cocktailsList", cocktailsList);
44      request.getRequestDispatcher( s: "cocktail_page.jsp").forward(request, response);
45  }
46
47  /**
48   * @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse response)
49   */
50  @protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
51      HttpSession session = request.getSession( b: true);
52      DatabaseProcedures dP = new DatabaseProcedures();
53
54      String name = request.getParameter( s: "Name");
55
56      Cocktails cocktail = dP.getCocktailN(name);
57
58      request.setAttribute( s: "cocktail", cocktail);
59      request.getRequestDispatcher( s: "cocktail_page.jsp").forward(request, response);
60  }
```

7. Κλάση «CocktailServlet2.java».

Ομοίως με την "CocktailServlet.java", απλώς καλείται όταν ο χρήστης είναι συνδεδεμένος ως "Admin".

```
CocktailServlet2.java
1  package mainpackage;
2
3  import ...
4
12
13  /**
14   * Servlet implementation class mainpackage.CocktailServlet
15   */
16  @WebServlet("/CocktailServlet2")
17  /* RUNS WHEN ADMIN CHOOSES A COCKTAIL FROM "admin_drinks_page.jsp" */
18  public class CocktailServlet2 extends HttpServlet {
19
20      /**
21       * @see HttpServlet#HttpServlet()
22       */
23      public CocktailServlet2() {
24          super();
25          // TODO Auto-generated constructor stub
26      }
27
28      /**
29       * @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)
30       */
31      @Override
32      protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
33          // TODO Auto-generated method stub
34          HttpSession session = request.getSession(true);
35          DatabaseProcedures dP = new DatabaseProcedures();
36
37          String name = request.getParameter("Name");
38
39          Cocktails cocktailsList = dP.getCocktailN(name);
40
41          request.setAttribute("cocktailsList", cocktailsList);
42          request.getRequestDispatcher("cocktail_page.jsp").forward(request, response);
43      }
44
45      /**
46       * @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse response)
47       */
48      @Override
49      protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
```


8. Κλάση «AddCocktailServlet.java»

Παρακάτω φαίνεται η κλάση «AddCocktailServlet» κατά την οποία ο admin εισάγει cocktail. Λαμβάνουν τιμή όλα τα χαρακτηριστικά του cocktail που θα εισαχθεί (name, base, taste, link, image) και μέσω της DatabaseProcedures εισάγονται και αυτά με την σειρά τους στην βάση.

```
21  /* RUNS WHEN ADD COCKTAIL FORM FROM ADMIN'S INDEX IS SUBMITTED */
22  public class AddCocktailServlet extends HttpServlet {
23
24      /**
25       * @see HttpServlet#HttpServlet()
26       */
27      public AddCocktailServlet() {
28          super();
29          // TODO Auto-generated constructor stub
30      }
31
32      /**
33       * @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)
34       */
35      protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
36          // TODO Auto-generated method stub
37      }
38
39      /**
40       * @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse response)
41       */
42      protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
43          String base = request.getParameter("Base");
44          String taste = request.getParameter("Taste");
45          String name = request.getParameter("Name");
46          String link = request.getParameter("Link");
47          Part filePart = request.getPart("Image");
48          InputStream fileContent = filePart.getInputStream();
49          byte[] imageBytes = IOUtils.toByteArray(fileContent);
50
51          Cocktails cocktail = new Cocktails(name, base, taste, ingredients: "", link, image: null);
52
53          DatabaseProcedures dP = new DatabaseProcedures();
54          String result = dP.addCocktail(cocktail, imageBytes);
55          response.setContentType("text/plain");
56          response.getWriter().print(result);
57          response.setHeader("Refresh", "3;url=adminsIndex.jsp");
58      }
59  }
60
```

9. Κλάση «AdminDrinkServlet.java».

Στην παρακάτω κλάση πραγματοποιείται η διαλογή των cocktails ανάλογα με την κατηγορία που θα επιλέξει ο χρήστης.

```
AdminDrinkServlet.java
16  /**
17   * Servlet implementation class mainpackage.CocktailServlet
18   */
19   @WebServlet("/AdminDrinkServlet")
20   /* RUNS WHEN ADMIN CHOOSES A DRINK CATEGORY */
21   public class AdminDrinkServlet extends HttpServlet {
22
23       /**
24        * @see HttpServlet#HttpServlet()
25        */
26       public AdminDrinkServlet() {
27           super();
28           // TODO Auto-generated constructor stub
29       }
30
31       /**
32        * @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)
33        */
34       @protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
35           // TODO Auto-generated method stub
36           HttpSession session = request.getSession( boolean true);
37           DatabaseProcedures dP = new DatabaseProcedures();
38
39           String base = request.getParameter( String "Base");
40           List<Cocktails> cocktailsList = dP.getCocktails(base);
41           Collections.sort(cocktailsList, new Comparator<Cocktails>() {
42               @Override
43               public int compare(Cocktails o1, Cocktails o2) { return o1.getName().compareTo(o2.getName()); }
44           });
45
46           request.setAttribute( String "cocktailsList", cocktailsList);
47           request.setAttribute( String "Base", base);
48           request.getRequestDispatcher( String "admin_drinks_page.jsp").forward(request, response);
49       }
50
51       /**
52        * @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse response)
53        */
54       @protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
55       }
```

10. Κλάση «DeleteCocktailServlet.java».

Πραγματοποιείται επιλογή και διαγραφή του επιθυμητού cocktail από την βάση, ενώ στην συνέχεια γίνεται ανανέωση της σελίδας.

```
DeleteCocktailServlet.java
3  import ...
16
17  /**
18   * Servlet implementation class mainpackage.CocktailServlet
19   */
20  @MultiPartConfig
21  @WebServlet("/DeleteCocktailServlet")
22  /* RUNS WHEN ADMIN CLICKS DELETE BUTTON FROM "admin_drinks_page.jsp" */
23  public class DeleteCocktailServlet extends HttpServlet {
24
25      /**
26       * @see HttpServlet#HttpServlet()
27       */
28      public DeleteCocktailServlet() {
29          super();
30          // TODO Auto-generated constructor stub
31      }
32
33      /**
34       * @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)
35       */
36      protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
37          // TODO Auto-generated method stub
38      }
39
40      /**
41       * @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse response)
42       */
43      @Override
44      protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
45          String name = request.getParameter("cocktailName");
46
47          DatabaseProcedures dP = new DatabaseProcedures();
48          String result = dP.deleteCocktail(name);
49
50          response.setContentType("text/plain");
51          response.getWriter().print(result);
52          response.setHeader("Refresh", "3;url=adminsIndex.jsp");
53      }
54  }
```

11. Κλάση «DrinkServlet.java».

Εκτελείται όταν ο χρήστη επιλέγει κατηγορία cocktail. Συλλέγει τα ποτά τα οποία βρίσκονται στην επιθυμητή κατηγορία και τα στέλνει στην «drink_page.jsp»

```
18  */
19  @WebServlet("/DrinkServlet")
20  /* RUNS WHEN USER CHOOSES A DRINK CATEGORY */
21  public class DrinkServlet extends HttpServlet {
22
23      /**
24       * @see HttpServlet#HttpServlet()
25       */
26      public DrinkServlet() {
27          super();
28          // TODO Auto-generated constructor stub
29      }
30
31      /**
32       * @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)
33       */
34      @Override
35      protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
36          // TODO Auto-generated method stub
37          HttpSession session = request.getSession(true);
38          DatabaseProcedures dP = new DatabaseProcedures();
39
40          String base = request.getParameter("Base");
41          List<Cocktails> cocktailsList = dP.getCocktails(base);
42          Collections.sort(cocktailsList, new Comparator<Cocktails>() {
43              @Override
44              public int compare(Cocktails o1, Cocktails o2) { return o1.getName().compareTo(o2.getName()); }
45          });
46
47          request.setAttribute("cocktailsList", cocktailsList);
48          request.setAttribute("Base", base);
49          request.getRequestDispatcher("drinks_page.jsp").forward(request, response);
50      }
51
52      /**
53       * @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse response)
54       */
55      @Override
56      protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
57      }
58  }
59  }
```

12. Κλάση «DatabaseProcedures.java».

Στην παρακάτω κλάση πραγματοποιείται η σύνδεση με την βάση δεδομένων και υπάρχουν όλες οι αντίστοιχες συναρτήσεις που χρησιμοποιούνται για αναζήτηση, εισαγωγή ή διαγραφή δεδομένων από την βάση.

Στην συνάρτηση "insertUser()" δημιουργείται το αντικείμενο user με όλα τα απαραίτητα στοιχεία για την εισαγωγή του στην βάση. Επίσης πραγματοποιείται έλεγχος εάν υπάρχει άλλος χρήστης με αυτό το email στην βάση.

```
DatabaseProcedures.java
16
17 public class DatabaseProcedures { // DAO
18
19     private final String url = "jdbc:postgresql://localhost:5432/Cocktails";
20     private final String user = "postgres";
21     private final String password = "1234";
22     private final String dbDriver = "org.postgresql.Driver";
23
24     public DatabaseProcedures() {
25     }
26
27     public void loadDriver(String dbDriver) {
28         try {
29             Class.forName(dbDriver);
30         } catch (ClassNotFoundException e) {
31             e.printStackTrace();
32         }
33     }
34
35     private Connection getConnection() {
36         Connection connection = null;
37         try {
38             connection = DriverManager.getConnection(url, user, password);
39         } catch (SQLException e) {
40             e.printStackTrace();
41         }
42         return connection;
43     }
44
45     public String insertUser(Object user) {
46
47         this.loadDriver(dbDriver);
48         Connection connection = this.getConnection();
49         String result = "User has been inserted successfully!";
50         String query = "INSERT INTO users(username, email, password, dateofbirth, category) VALUES (?, ?, ?, ?, ?)";
51
52         try {
53             PreparedStatement ps = connection.prepareStatement(query);
54             ps.setString(1, ((Users) user).getUsername());
55             ps.setString(2, ((Users) user).getEmail());
```

Στην συνάρτηση "loginValidation()" πραγματοποιείται έλεγχος των στοιχείων που εισάγονται από τον χρήστη.

```
DatabaseProcedures.java
60      ps.executeUpdate();
61      ps.close();
62      connection.close();
63  } catch (SQLException e) {
64      result = "User insertion failed! \r\nCheck the database connection or check if there is already a client with the same email!";
65      e.printStackTrace();
66  }
67      return result;
68  }
69
70  public boolean loginValidation(String email, String password) {
71
72      this.loadDriver(dbDriver);
73      Connection connection = this.getConnection();
74      boolean result = false;
75      String query = "SELECT email, password FROM users WHERE email = ?";
76
77      try {
78          PreparedStatement ps = connection.prepareStatement(query);
79          ps.setString(1, email);
80          ResultSet resultSet = ps.executeQuery();
81
82          if (resultSet.next()) {
83              String pass = resultSet.getString("password");
84              if (pass.equals(password)) {
85                  result = true;
86              } else {
87                  result = false;
88              }
89          } else {
90              result = false;
91          }
92
93          ps.close();
94          connection.close();
95      } catch (SQLException e) {
96          e.printStackTrace();
97      }
98      return result;
99  }
```

Στην συνάρτηση "getUserCategory()" ελέγχουμε εάν ο χρήστης είναι "Admin" ή "User" ώστε να εμφανιστεί η αντίστοιχη αρχική σελίδα.

```
DatabaseProcedures.java
101 public String getUsername(String email) {
102
103     this.loadDriver(dbDriver);
104     Connection connection = this.getConnection();
105     String result = "";
106     String query = "SELECT email, username FROM users WHERE email = ?";
107
108     try {
109         PreparedStatement ps = connection.prepareStatement(query);
110         ps.setString( parameterIndex: 1, email);
111         ResultSet resultSet = ps.executeQuery();
112
113         if (resultSet.next()) {
114             result = resultSet.getString( columnLabel: "username");
115         } else {
116             result = "Failed";
117         }
118
119         ps.close();
120         connection.close();
121     } catch (SQLException e) {
122         e.printStackTrace();
123     }
124     return result;
125 }
126
127 public String getUserCategory(String email) {
128
129     this.loadDriver(dbDriver);
130     Connection connection = this.getConnection();
131     String result = "";
132     String query = "SELECT email, category FROM users WHERE email = ?";
133
134     try {
135         PreparedStatement ps = connection.prepareStatement(query);
136         ps.setString( parameterIndex: 1, email);
137         ResultSet resultSet = ps.executeQuery();
138
139         if (resultSet.next()) {
140             result = resultSet.getString( columnLabel: "category");
```

Η συνάρτηση "getCocktail()" αναζητεί στην βάση το κοκτέιλ με τα χαρακτηριστικά που επέλεξε ο χρήστης, δηλαδή την βάση και την γεύση, και επιστρέφει το αντίστοιχο κοκτέιλ.

```
DatabaseProcedures.java
146      connection.close();
147    } catch (SQLException e) {
148      e.printStackTrace();
149    }
150    return result;
151  }
152
153  public Cocktails getCocktail(String base, String taste, String ingredients) {
154
155    String name = "", link = "";
156    this.loadDriver(dbDriver);
157    Connection connection = this.getConnection();
158    PreparedStatement ps;
159    ResultSet resultSet;
160    Cocktails cocktail = new Cocktails(base, taste, ingredients);
161
162    try {
163      String query1 = "SELECT name, link FROM recipes WHERE base = ? AND taste = ?";
164
165      ps = connection.prepareStatement(query1); //Get name and link from database/table recipes.
166      ps.setString(1, base);
167      ps.setString(2, taste);
168      //ps.setString(3, ingredients);
169      resultSet = ps.executeQuery();
170
171      if (resultSet.next()) {
172        name = resultSet.getString("name");
173        cocktail.setName(name);
174        link = resultSet.getString("link");
175        cocktail.setLink(link);
176      }
177
178      resultSet.close();
179      ps.close();
180      connection.close();
181      return cocktail;
182    } catch (SQLException e) {
183      e.printStackTrace();
184      return null;
185    }
186  }
```


Η συνάρτηση "getCocktailN()" επιστρέφει το όνομα του κοκτέιλ που επέλεξε ο χρήστης.

```
DatabaseProcedures.java x
186 }
187
188 public Cocktails getCocktailN(String name) {
189
190     this.loadDriver(dbDriver);
191     Connection connection = this.getConnection();
192     PreparedStatement ps;
193     ResultSet resultSet;
194     Cocktails cocktail = new Cocktails(name, base: "", taste: "", ingredients: "", link: "", image: null);
195
196     try {
197         String query1 = "SELECT base, taste, link FROM recipes WHERE name = ?";
198
199         ps = connection.prepareStatement(query1); //Get everything from database/table recipes for specific cocktail
200         ps.setString(1, name);
201         resultSet = ps.executeQuery();
202
203         if (resultSet.next()) {
204             String base = resultSet.getString("base");
205             cocktail.setBase(base);
206             String taste = resultSet.getString("taste");
207             cocktail.setTaste(taste);
208             String link = resultSet.getString("link");
209             cocktail.setLink(link);
210         }
211         resultSet.close();
212         ps.close();
213
214         //cocktail = Cocktails(name, base, taste, ingredients, link);
215
216         connection.close();
217         return cocktail;
218     } catch (SQLException e) {
219         e.printStackTrace();
220         return null;
221     }
222 }
223
224 public List<Cocktails> getCocktails(String base) {
225
```

Η συνάρτηση "getCocktails()" επιστρέφει μια λίστα με όλα τα κοκτέιλ που έχουν βάση το ποτό που επέλεξε ο χρήστης.

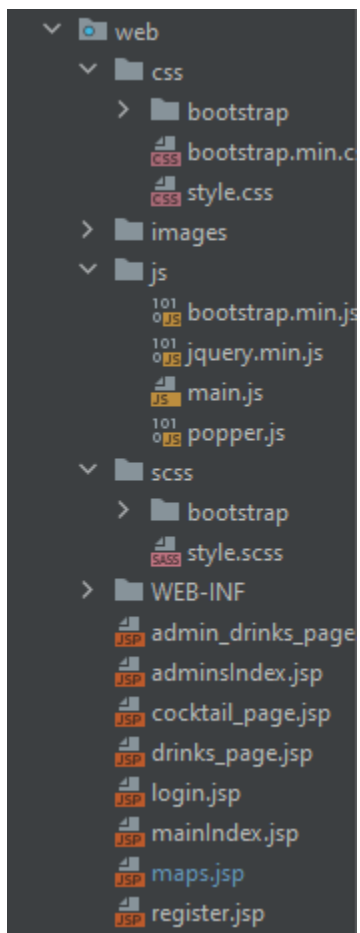
```
DatabaseProcedures.java x
223
224 public List<Cocktails> getCocktails(String base) {
225
226     this.loadDriver(dbDriver);
227     Connection connection = this.getConnection();
228     PreparedStatement ps;
229     ResultSet resultSet;
230     String query = "SELECT * FROM recipes WHERE base = ?;";
231     List<Cocktails> cocktailsList = new ArrayList<>();
232
233     try {
234         ps = connection.prepareStatement(query);
235         ps.setString( parameterIndex: 1, base);
236         resultSet = ps.executeQuery();
237
238         while ( resultSet.next() ) {
239             String name = resultSet.getString( columnLabel: "name");
240             String taste = resultSet.getString( columnLabel: "taste");
241             String link = resultSet.getString( columnLabel: "link");
242             byte[] image = resultSet.getBytes( columnLabel: "image");
243             String encoded = Base64.getEncoder().encodeToString(image);
244
245             Cocktails c = new Cocktails(name, base, taste, ingredients: "", link, encoded);
246             cocktailsList.add(c);
247         }
248         resultSet.close();
249         ps.close();
250         connection.close();
251         return cocktailsList;
252     } catch (SQLException e) {
253         e.printStackTrace();
254         return cocktailsList;
255     }
256 }
257
258 @ public String addCocktail(Cocktails cocktail, byte[] imageBytes) {
259
260     this.loadDriver(dbDriver);
261     Connection connection = this.getConnection();
262     String result = "";
```

Στην συνάρτηση "addCocktail()" πραγματοποιείται η εισαγωγή και ο έλεγχος εάν υπάρχει ήδη το cocktail το οποίο επιθυμούμε να εισάγουμε.

Τέλος, στην συνάρτηση "deleteCocktail()" πραγματοποιείται η διαγραφή του επιθυμητού cocktail από την βάση.

```
DatabaseProcedures.java
258 @ public String addCocktail(Cocktails cocktail, byte[] imageBytes) {
259
260     this.loadDriver(dbDriver);
261     Connection connection = this.getConnection();
262     String result = "";
263     String query = "INSERT INTO recipes(name, base, taste, ingredients, link, image) VALUES (?, ?, ?, 'NO', ?, ?)";
264
265     try {
266         PreparedStatement ps= connection.prepareStatement(query);
267         ps.setString( parameterIndex: 1, cocktail.getName());
268         ps.setString( parameterIndex: 2, cocktail.getBase());
269         ps.setString( parameterIndex: 3, cocktail.getTaste());
270         ps.setString( parameterIndex: 4, cocktail.getLink());
271         ps.setBytes( parameterIndex: 5, imageBytes);
272
273         ps.executeUpdate();
274         ps.close();
275         result += "Cocktail has been added successfully!";
276         connection.close();
277     } catch (SQLException e) {
278         result += "Cocktail insertion failed! \r\nCheck the database connection or check if there is already a cocktail with the same name!";
279         e.printStackTrace();
280     }
281     return result;
282 }
283
284 public String deleteCocktail(String name) {
285
286     this.loadDriver(dbDriver);
287     Connection connection = this.getConnection();
288     String result = "";
289     String query = "DELETE FROM recipes WHERE name = ?";
290
291     try {
292         PreparedStatement ps= connection.prepareStatement(query);
293         ps.setString( parameterIndex: 1, name);
294
295         ps.executeUpdate();
296         ps.close();
297         result += "Cocktail has been deleted successfully!";
```

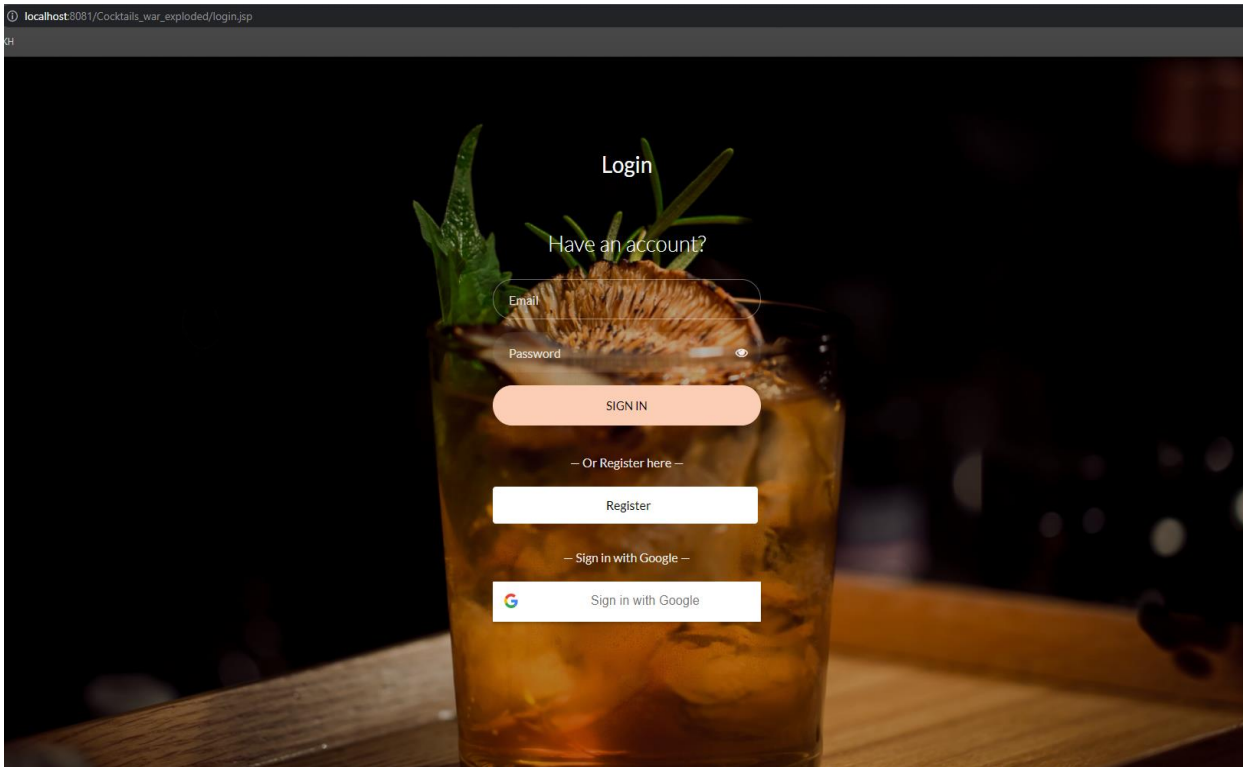
JSPs-HTML



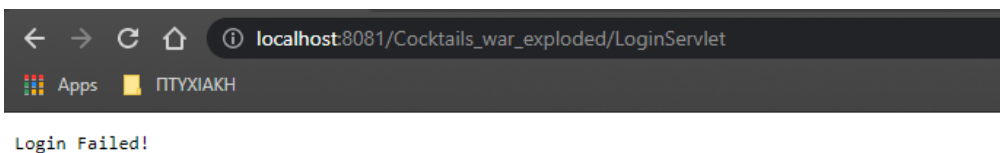
Δημιουργία διαδικτυακής διεπαφής

Κύρια διαδικτυακή διεπαφή

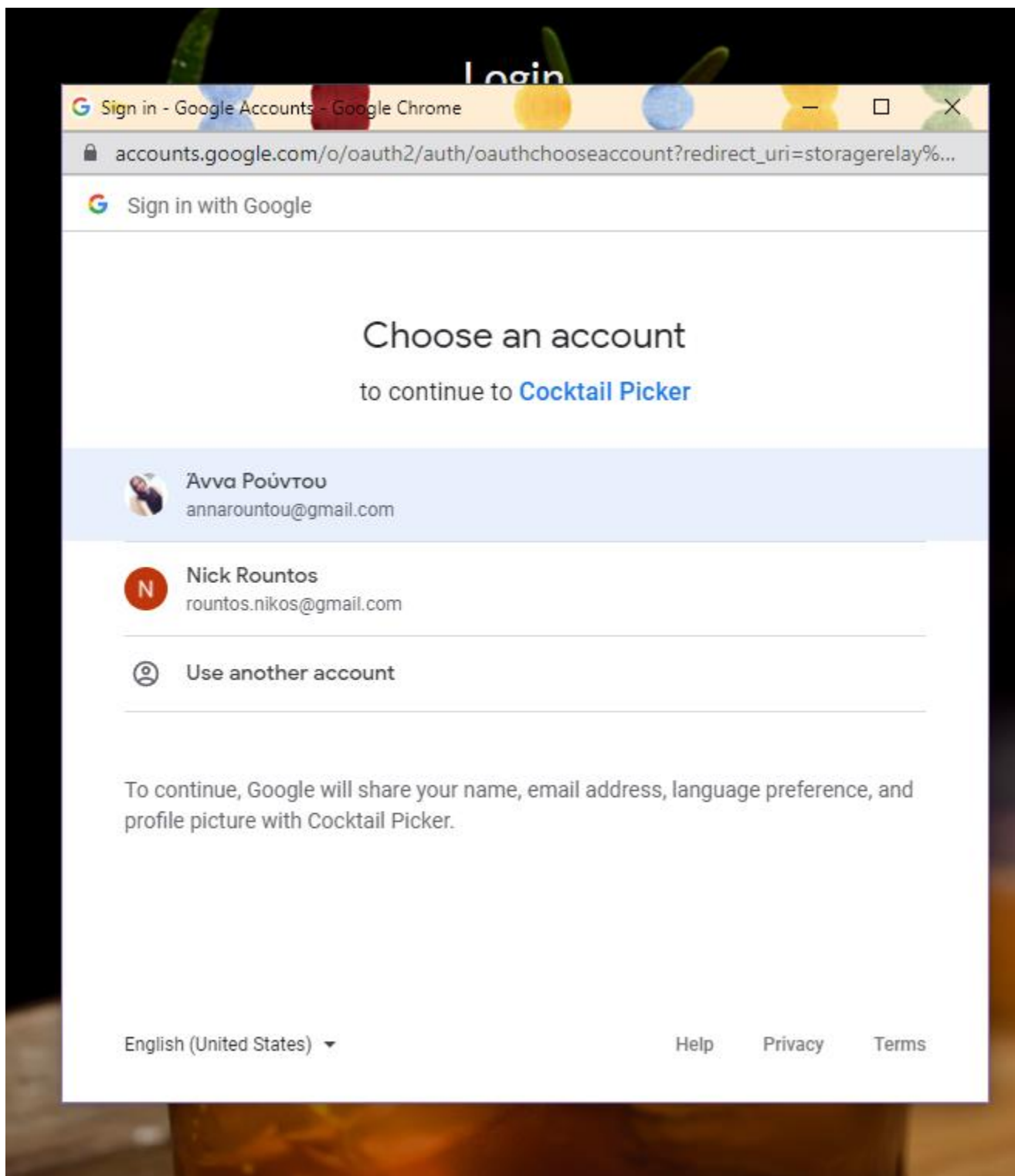
Η «login.jsp» αποτελεί την βασική διαδικτυακή διεπαφή που χρησιμοποιούν όλοι οι χρήστες της εφαρμογής. Οι χρήστες κάνουν login μέσω αυτής.



Γίνεται εισαγωγή στοιχείων χρήστη ο οποίος δεν είναι εγγεγραμμένος ή εάν έχει εισάγει λανθασμένα στοιχεία. Λαμβάνουμε το παρακάτω μήνυμα.



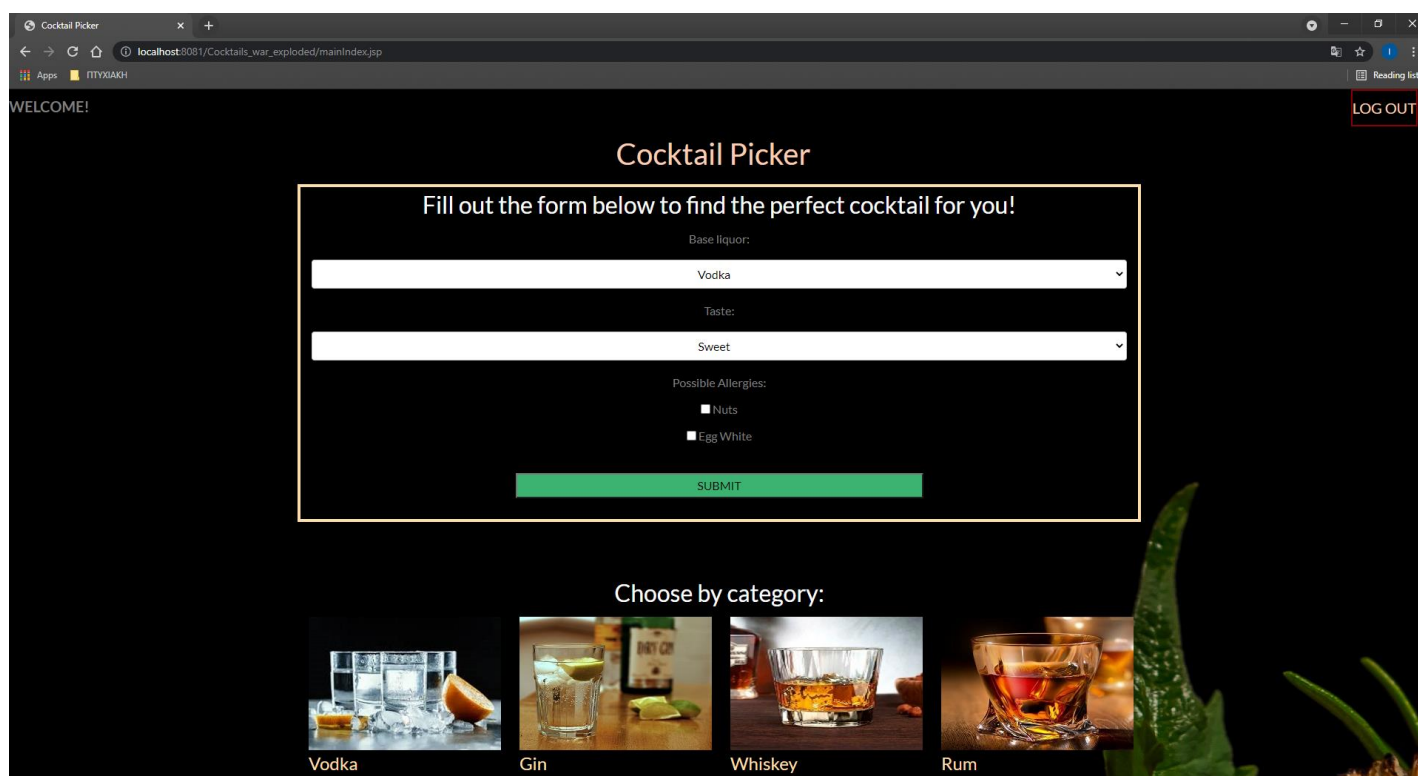
Σύνδεση στην εφαρμογή επίσης μπορεί να γίνει και μέσω Google, όπως φαίνεται στην εικόνα παρακάτω. Όσοι χρήστες συνδέονται στην εφαρμογή μέσω Google είναι αυτομάτως απλοί χρήστες.



Επιτυχής σύνδεση απλού χρήστη

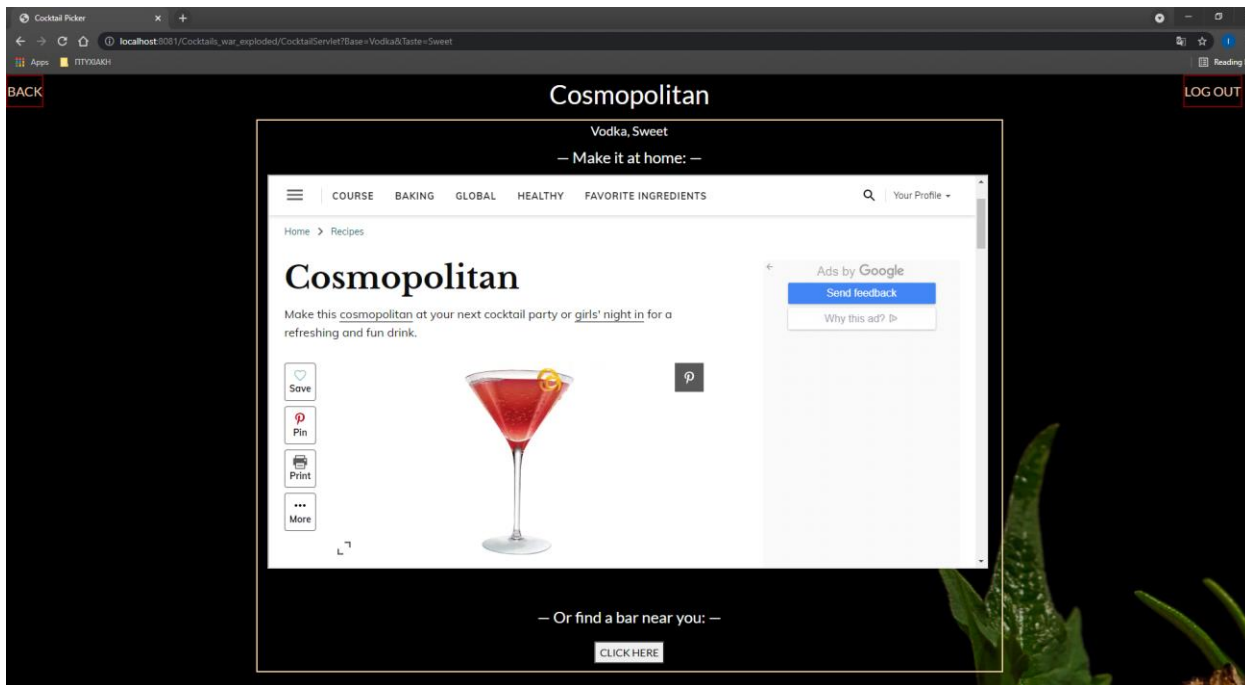
Αφού πραγματοποιηθεί εισαγωγή ενός απλού χρήστη στην εφαρμογή εμφανίζεται η αρχική σελίδα κατά την οποία ο χρήστης μπορεί να επιλέξει μέσω των drop-down λιστών το ποτό και την γεύση του cocktail που επιθυμεί. Επίσης, έχει την δυνατότητα να επιλέξει κατηγορία ποτών πατώντας τις εικόνες στο κάτω σημείο της σελίδας. Στο πάνω δεξιά μέρος της σελίδας υπάρχει το κουμπί «LOG OUT».

Να σημειωθεί ότι καθ' όλη την διάρκεια χρήσης της εφαρμογής ο χρήστης έχει την δυνατότητα επιστροφής στην προηγούμενη σελίδα μέσω της επιλογής BACK (επάνω αριστερά εκτός της αρχικής σελίδας) ή της αποσύνδεσης μέσω του κουμπιού «LOG OUT»



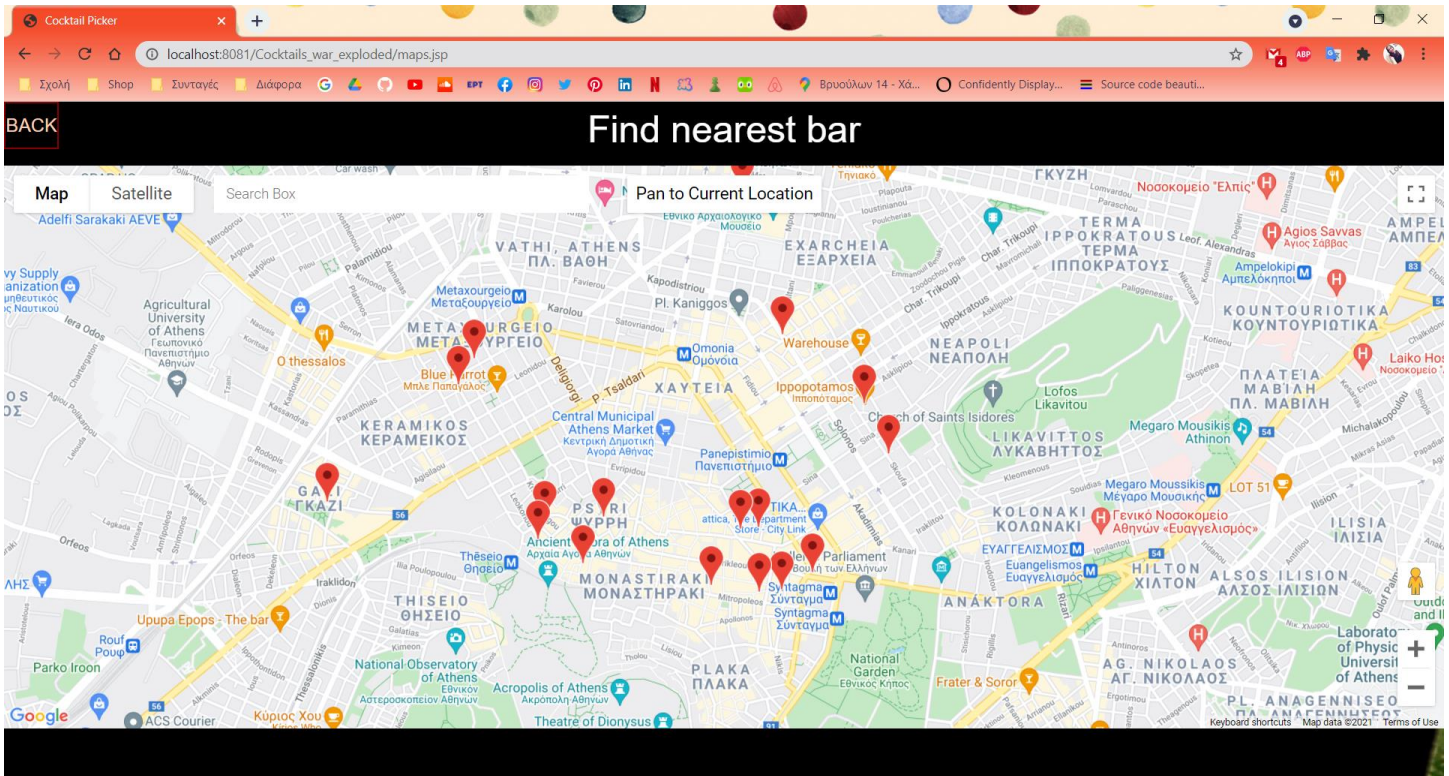
The screenshot shows a web browser window with the title "Cocktail Picker". The address bar shows the URL "localhost:3081/Cocktails_war_exploded/mainIndex.jsp". The page has a dark background and a "WELCOME!" message in the top left. In the top right corner, there is a "LOG OUT" button. The main content area features a form titled "Cocktail Picker" with the instruction "Fill out the form below to find the perfect cocktail for you!". The form contains three dropdown menus: "Base liquor:" with "Vodka" selected, "Taste:" with "Sweet" selected, and "Possible Allergies:" with checkboxes for "Nuts" and "Egg White". A green "SUBMIT" button is at the bottom of the form. Below the form, there is a section titled "Choose by category:" with four image-based buttons labeled "Vodka", "Gin", "Whiskey", and "Rum". Each button shows a glass of the respective cocktail.

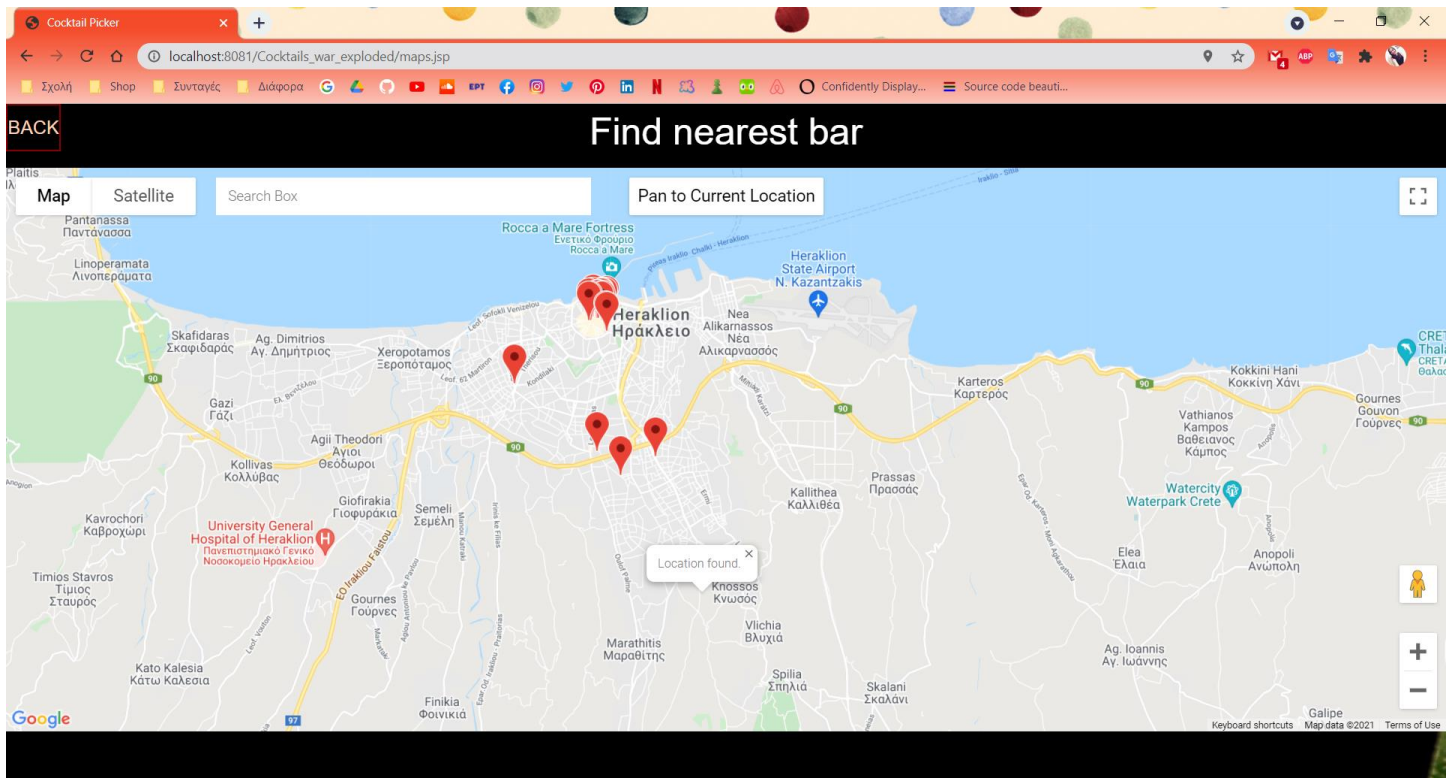
Στην περίπτωση που επιλέξει ποτό ανάλογα όχι μόνο με την κατηγορία του ποτού, αλλά και με την γεύση του, εμφανίζεται η συνταγή του προτεινόμενου cocktail με τα επιθυμητά χαρακτηριστικά. Στο χαμηλότερο σημείο της σελίδας υπάρχει το κουμπί «CLICK HERE» το οποίο του εμφανίζει τον χάρτη με τα bar σε οποιαδήποτε περιοχή επιθυμεί ο χρήστης.



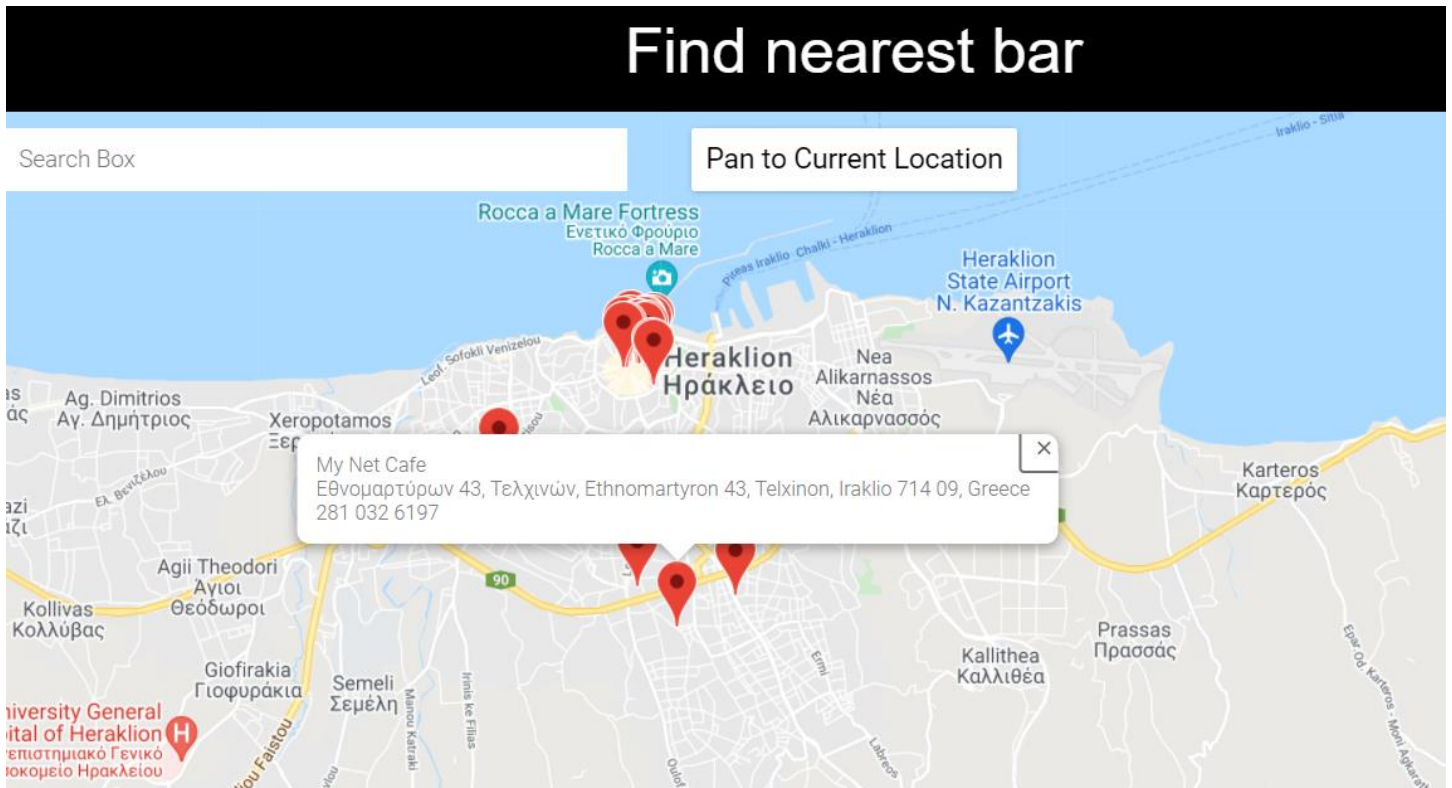
Όταν πατήσει το κουμπί «CLICK HERE» μεταφέρετε στην παρακάτω σελίδα. Η σελίδα αυτή εμφανίζει έναν χάρτη με αρχική τοποθεσία το κέντρο της Αθήνας. Όπως βλέπουμε στην φωτογραφία υπάρχουν κάποια κόκκινα pins, τα οποία είναι όλα τα μπαρ που υπάρχουν γύρω από αυτή την τοποθεσία σε απόσταση περίπου 2 χιλιομέτρων.

Οι επιλογές του χρήστη είναι είτε να ψάξει κάποιο μπαρ μόνος του είτε να πατήσει το κουμπί "Pan to Current Location" το οποίο τον μεταφέρει στην τοποθεσία του και εμφανίζει πάλι με κόκκινα βελάνια τα τριγύρω μπαρ(βλέπετε δεύτερη φωτογραφία).

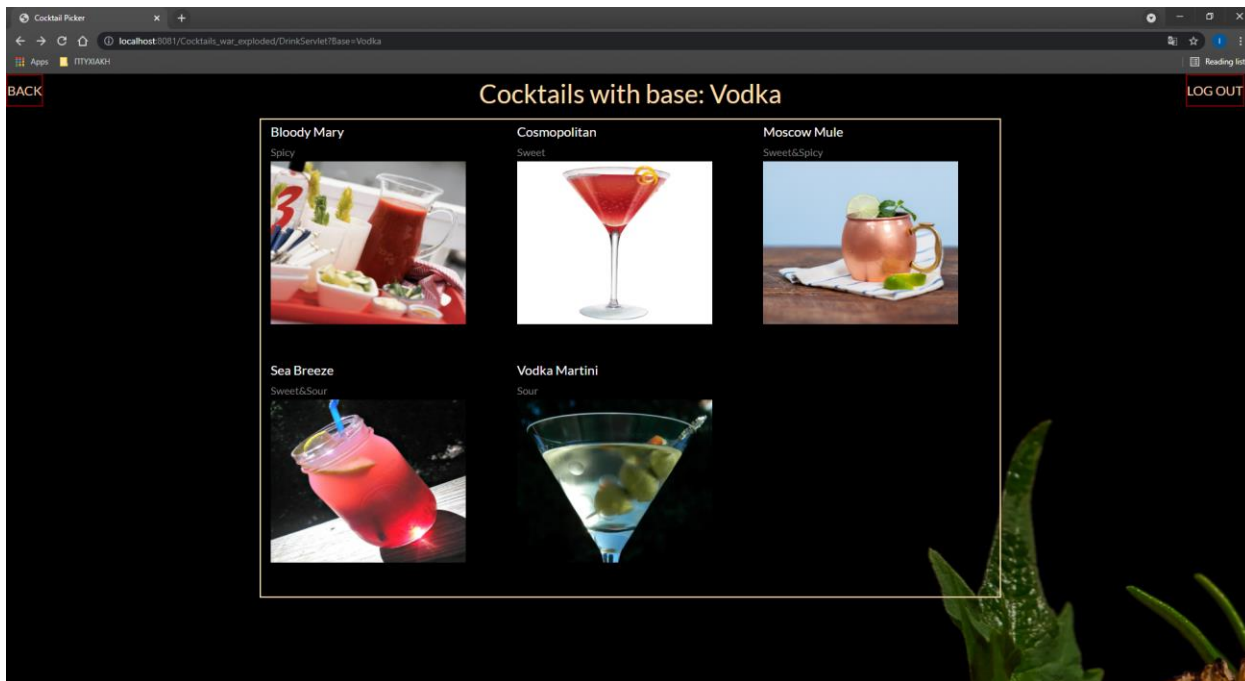




Τα βελάκια περιέχουν το όνομα, την διεύθυνση και το τηλέφωνο του κάθε μπαρ.



Εάν ο χρήστης επιλέξει ποτό με βάση μόνο το είδος του βασικού ποτού, μέσω των φωτογραφιών, εμφανίζονται όλα τα ποτά τα οποία διαθέτουν ως βάση το επιθυμητό ποτό. Αφού ο χρήστης διαλέξει το συγκεκριμένο ποτό που επιθυμεί, εμφανίζεται η παραπάνω φωτογραφία. (Προηγούμενη περίπτωση)



Επιτυχής σύνδεση ως Admin

Κατά την επιτυχή σύνδεση του admin εμφανίζεται η σελίδα κατά την οποία διαλέγει την βάση, την γεύση, το όνομα, τον σύνδεσμο της σελίδας από τον οποίο θα ληφθεί η συνταγή και τέλος μία φωτογραφία με σκοπό να εισάγει ένα νέο cocktail στην βάση δεδομένων.

localhost:8081/Cocktails_war_explored/adminIndex.jsp

Cocktail Picker

Fill out the form below to add a cocktail:

Base liquor:
Vodka

Taste:
Sweet

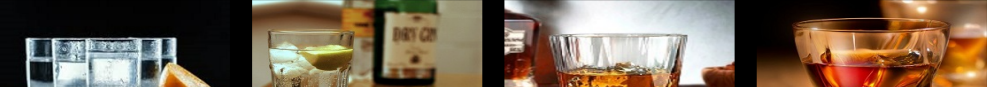
Name of cocktail:
Name

Link of recipe:
Link

Select image for cocktail:
Choose Files No file chosen

SUBMIT

Choose by category:



Για να μπορέσει να εισαχθεί ένα cocktail επιτυχώς στην βάση είναι αναγκαίο να συμπληρωθούν όλες οι παραπάνω πληροφορίες του.

Cocktail Picker

Fill out the form below to add a cocktail:

Base liquor:

Vodka

Taste:

Sweet

Name of cocktail:

Name

Please fill out this field.

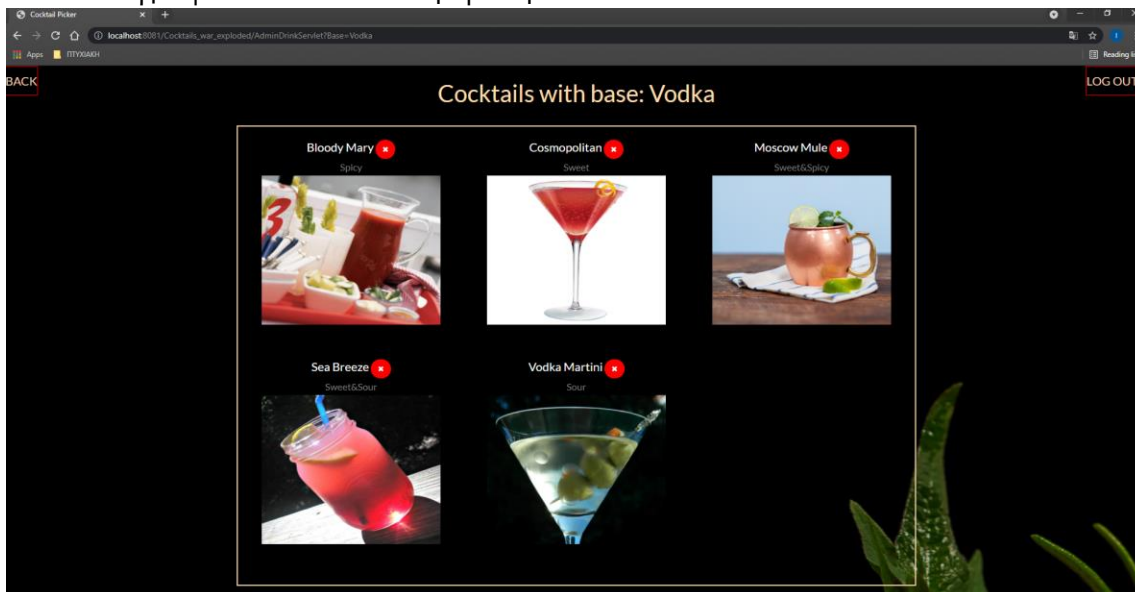
Link

Select image for cocktail:

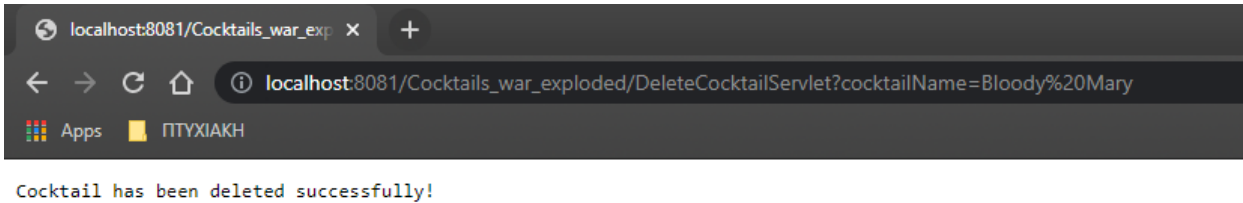
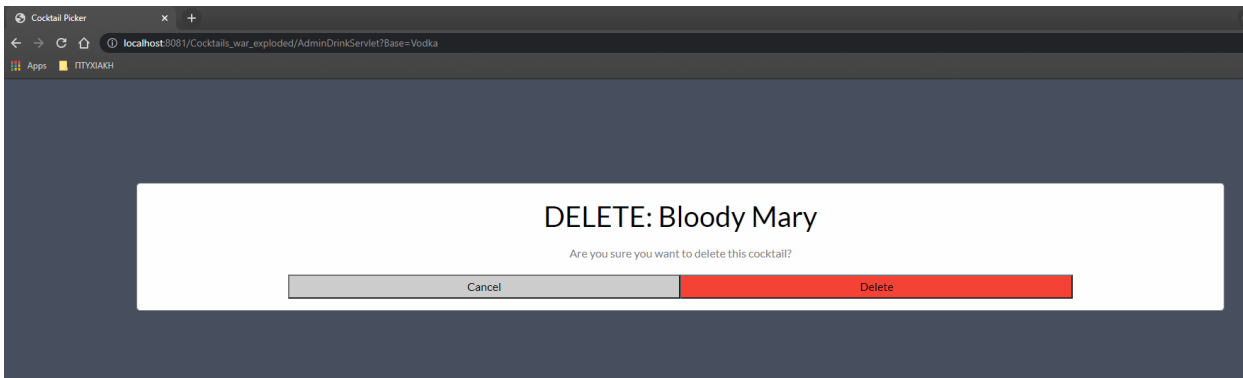
Choose Files No file chosen

SUBMIT

Τέλος, εάν ο admin επιλέξει τις φωτογραφίες των πότων, ανά κατηγορία δηλαδή, εμφανίζονται τα αντίστοιχα ποτά της εκάστοτε κατηγορίας. Δίπλα από το όνομα κάθε ποτού υπάρχει ένα κόκκινο κουμπί το οποίο διαγράφει το ποτό από την βάση.



Περίπτωση επιλογής διαγραφής του ποτού «Bloody Mary»:



Ανανεωμένη λίστα cocktail μετά την διαγραφή του «Bloody Mary»:



Χρήση των Google APIs

Google Maps API

Για να χρησιμοποιήσουμε το παραπάνω API χρειάστηκε να φτιάξουμε ένα project στο "Google Cloud Platform", να ενεργοποιήσουμε τα αντίστοιχα APIs και να δημιουργήσουμε ένα API key.

Δημιουργία project:

Google Cloud Platform Cocktails

Search products and resources

DASHBOARDACTIVITYRECOMMENDATIONS

Quick Access

Cocktails

API APIs & Services

Accessed Aug 17, 2021

Cocktails

API Credentials – APIs & Services – Cockt...

Accessed 3 minutes ago

Cocktails

API OAuth consen

Accessed Aug

Project info

Project name
Cocktails

Project ID
cocktails-323213

Project number
140271236674

ADD PEOPLE TO THIS PROJECT

Go to project settings

Resources

This project has no resources

Trace

API APIs

Requests (requests/sec)

No data is available for the selected ti


1:15 1:30 1:45

Go to APIs overview

Ενεργοποίηση APIs:

Google Cloud Platform Cocktails

←



Maps JavaScript API

Google


Maps for your website

MANAGE ✓ API Enabled

OVERVIEW DOCUMENTATION SUPPORT

Google Cloud Platform Cocktails

←



Places API

Google Enterprise API

Get detailed information about 100 million places

MANAGE ✓ API Enabled

OVERVIEW PRICING DOCUMENTATION SUPPORT

Δημιουργία API κλειδιού:

Google Cloud Platform Cocktails Search products and resources

API APIs & Services

Dashboard Library Credentials OAuth consent screen Domain verification Page usage agreements

← Restrict and rename API key REGENERATE KEY DELETE


Name *
API key 1

API Key
AIzaSyBaSgU8JV7VZSce0m0SDzQTN_pLIFavT_8

Use this key in your application by passing it with `key=API_KEY` parameter.

Creation date
September 9, 2021 at 2:39:23 PM GMT+3

Key restrictions

 This key is unrestricted. Restrictions help prevent unauthorized use and quota theft. [Learn more](#)

Application restrictions

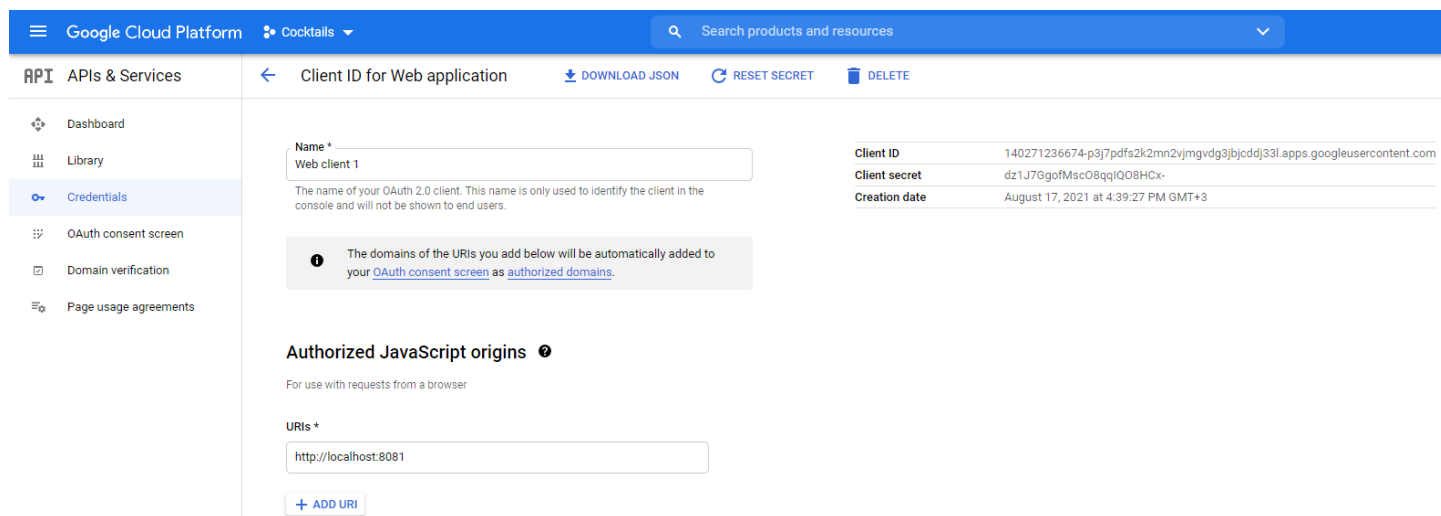
Και τέλος, ενσωμάτωση και χρήση στο project:

```
246 <script
247   src="https://maps.googleapis.com/maps/api/js?key=AIzaSyBaSgU8JV7VZSce0m0SDzQTN_pLIFavT_8&callback=initMap&libraries=places&v=weekly"
248   async
249 ></script>
```


Google Sign-In API

Ομοίως με το παραπάνω, πρέπει να δημιουργήσουμε ένα Client ID και να το ενσωματώσουμε στο project μας.

Δημιουργία Client ID:



Google Cloud Platform Cocktails

Search products and resources

APIs & Services

Client ID for Web application

DOWNLOAD JSON RESET SECRET DELETE

Dashboard

Library

Credentials

OAuth consent screen

Domain verification

Page usage agreements

Name *

Web client 1

The name of your OAuth 2.0 client. This name is only used to identify the client in the console and will not be shown to end users.

Client ID

140271236674-p3j7pdfs2k2mn2vjmgvdg3bjcddj33l.apps.googleusercontent.com

Client secret

dz1J7GgofMscO8qIQ08HCx-

Creation date

August 17, 2021 at 4:39:27 PM GMT+3

The domains of the URIs you add below will be automatically added to your [OAuth consent screen](#) as [authorized domains](#).

Authorized JavaScript origins

For use with requests from a browser

URIs *

http://localhost:8081

+ ADD URI

Και ενσωμάτωση στο project:

```
22 <!-- Google sign-in -->
23 <meta name="google-signin-scope" content="profile email">
24 <meta name="google-signin-client_id" content="140271236674-p3j7pdfs2k2mn2vjmgvdg3bjcddj33l.apps.googleusercontent.com">
25 <script src="https://apis.google.com/js/platform.js" async defer></script>
```