



Lunch With Friends

Amanda Zhang, Amir Hegazy, Brenna Chen,
Jaemyung Choi, Soumika Guduru

What is Lunch With Friends?

It's a web application designed to help you meet people who are interested in going to the same restaurants.



Flow of Our App

- Log in with Google
- Search for a restaurant
- Click the restaurant you'd like to visit
- Select a group or single option
- Get matched to another user
- Enter a chat room with them!



Continuing Professional Development

- **Server endpoints**
- **Path parameters**
- JSON Object
- Object serialization into a database
- Session variables
- Google Maps API
- Google Sign-in/OAuth 2.0 API
- Google Cloud Platform



Server Endpoints + Path Parameters

```
1 package CSCI201_LunchWithFriends;
2
3 import java.io.IOException;
4
24 @ServerEndpoint(value = "/chatroomServer/{userID}/{room}")
25 public class ChatRoomServer {
26     /*
27     * Map of key: room number value: set of sessions (iow set of users in a chat)
28     * in easier terms- the key is the chatName/roomName, and the value is the set of users in that chat
29     */
30     private static Map<String, Set<Session>> chatRoomMap = (Map<String, Set<Session>>) Collections.synchronizedMap(new HashMap<String, Set<Session>>());
31
32     @OnOpen
33     public void open(Session session, @PathParam("userID") String userID, @PathParam("room") String room) throws IOException {
34         session.getUserProperties().put("userID", userID);
35         session.getUserProperties().put("room", room);
36         /*
37         * getChat --> Either finds the room this user will chat inside
38         * if no such room exists --> Creates chatroom + stores it inside chatRoomMap as <room, this users session>
39         * chatroom will be a set of user sessions who can message each other
40         * -->in this case, the set of other users this user session with chat with
41         */
42         Set<Session> chatroom = getChat(room);
43         /*
44         * Adding current user's session into created chatroom
45         */
46         chatroom.add(session);
47         session.getBasicRemote().sendText(makeText("Chat System", "you are now connected as "+userID+" in ChatRoom: "+room+"!"));
48     }
49
50     @OnMessage
51     public void recievedMessage(String message, Session session) throws IOException
52     {
53         String userID = (String) session.getUserProperties().get("userID");
54         String room = (String) session.getUserProperties().get("room");
55         /*
56         * find the chatroom the user is in */
57         Set<Session> chatroom = getChat(room);
58         /*
59         * Sends message to all chat users in the same chatroom */
60         for(Session rs : chatroom) {
61             if(rs.isOpen()) {
62                 rs.getBasicRemote().sendText(makeText(userID, message));
63             }
64         }
65     }
66 }
```

Continuing Professional Development

- Server endpoints
- Path parameters
- **JSON Object**
- Object serialization into a database
- Session variables
- Google Maps API
- Google Sign-in/OAuth 2.0 API
- Google Cloud Platform



JSONObject

Server Endpoint

```
private String makeText(String userID, String msg) {
    JSONObject jsonObj = Json.createObjectBuilder().add("msg", userID+": "+msg).build();
    StringWriter sWriter = new StringWriter();
    try(JsonWriter jWriter = Json.createWriter(sWriter)){
        jWriter.write(jsonObj);
    }
    return sWriter.toString();
}
```

Chat Client (chatBasic.jsp)

```
var getp1 = "${userID}";
var p1 = String(getp1);
var getp2 = "${room}";
var p2 = String(getp2);

var socket = new WebSocket(
    "ws://localhost:8080/LunchWithFriends/chatroomServer" + "/" + p1
    + "/" + p2);

socket.onmessage = function processMessage(ricievedMessage) {
    var mdata = JSON.parse(ricievedMessage.data);
    if (mdata.msg != null)
        msgTextArea.value += mdata.msg + "\n";
}
```

Continuing Professional Development

- Server endpoints
- Path parameters
- JSON Object
- **Object serialization into a database**
- Session variables
- Google Maps API
- Google Sign-in/OAuth 2.0 API
- Google Cloud Platform



Continuing Professional Development

- Server endpoints
- Path parameters
- JSON Object
- Object serialization into a database
- **Session variables**
- Google Maps API
- Google Sign-in/OAuth 2.0 API
- Google Cloud Platform



Continuing Professional Development

- Server endpoints
- Path parameters
- JSON Object
- Object serialization into a database
- Session variables
- **Google Maps API**
- Google Sign-in/OAuth 2.0 API
- Google Cloud Platform



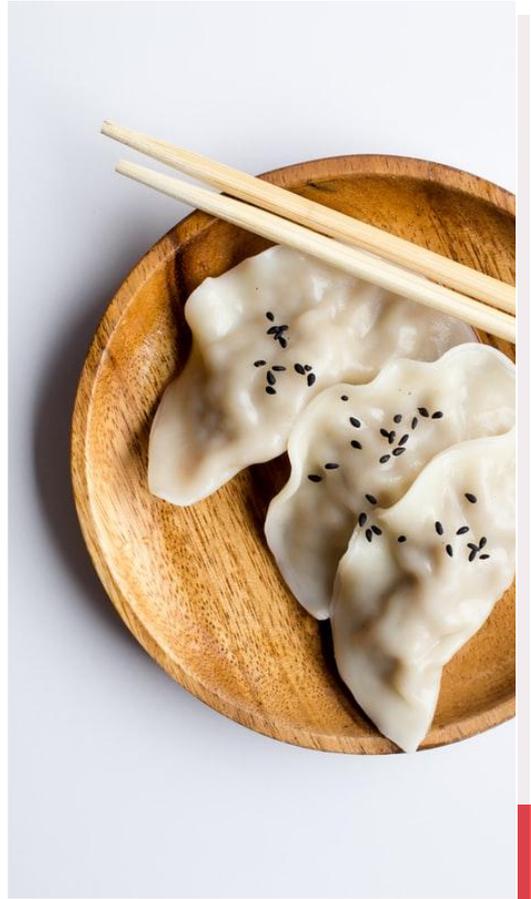
Continuing Professional Development

- Server endpoints
- Path parameters
- JSON Object
- Object serialization into a database
- Session variables
- Google Maps API
- **Google Sign-in/OAuth 2.0 API**
- Google Cloud Platform



Continuing Professional Development

- Server endpoints
- Path parameters
- JSON Object
- Object serialization into a database
- Session variables
- Google Maps API
- Google Sign-in/OAuth 2.0 API
- **Google Cloud Platform**



Google Cloud Platform

The screenshot shows the Google Cloud Platform dashboard for the project 'CSCI201-LunchWithFriends'. The interface includes a top navigation bar with the project name, a search bar, and utility icons. Below the navigation bar are tabs for 'DASHBOARD', 'ACTIVITY', and 'RECOMMENDATIONS', along with a 'CUSTOMIZE' link. A notification banner at the top reads 'How Google Cloud is helping during COVID-19. Learn more'. The main content area is divided into several sections:

- Project info:** Displays project details such as name, ID, and number, with a link to 'ADD PEOPLE TO THIS PROJECT' and 'Go to project settings'.
- Resources:** Shows 'SQL' with '1 instance' and a link to 'Go to the SQL dashboard'.
- Trace:** Indicates 'No trace data from the past 7 days'.
- SQL Metrics:** A line chart titled 'Storage used (bytes)' shows usage increasing from 10:30 to 11:15 AM, reaching 1215MB. A legend indicates 'database/disk/bytes_used: 1.186GiB'.
- API APIs:** A bar chart titled 'Requests (requests/sec)' shows request rates between 0.018/s and 0.019/s.
- Google Cloud Platform status:** States 'All services normal' and provides a link to 'Go to Cloud status dashboard'.
- Billing:** Shows 'Estimated charges' of 'USD \$0.00' for the period 'Nov 1 - 22, 2020', with a link to 'View detailed charges'.
- Monitoring:** Offers options to 'Set up alerting policies', 'Create uptime checks', and a link to 'View all dashboards'.

Skills and Tools

- Eclipse
- Apache Tomcat
- GSON
- Javax
- Google Cloud Platform
- MySQL
- Yelp API
- Google Maps API
- OkHttpClient



Design and Development Decisions

What worked

- Strong communication + helping each other on tasks
- Clear GUI mockup & database schema to base front & back-end
- Clear step-by-step process for user to follow in detailed design document
- Flexible project design to tailor to issues/time constraints



What didn't work

- Difficulty connecting parts together, not enough familiarity with team members' code
- Testing was difficult b/c many parts were interdependent
- Not setting hard deadlines for coding portions -> lots of delay due to interdependent parts



Outside Courses

- CS103: Introduction to Programming
- CS104: Data Structures and OOP
- CS270: Introduction to Algorithms
- ITP104: Web Publishing
- EE109: Introduction to Embedded Systems



Data Structures

- HashMap
- ArrayList
- Synchronized Map
- Synchronized Set
- Map
- Set

Why did we choose these data structures?



Multithreading + Networking

- Chat Server Endpoint
- Synchronized Java Collections
- Apache Tomcat + Servlets
 - Servlet containers use threading to serve the same servlet to different requests



User Login Functionality

Guest User

- Can search & view restaurants

Authenticated User (Log-in through Google)

- Can search & view restaurants
- Can see other people interested in the same restaurant
- Can be matched to other users
 - A chatroom will be created between the users



“

Thanks for watching!

If you have any questions,
feel free to email us:

Brenna Chen <brennajc@usc.edu>

Jae Choi <jaemyung@usc.edu>

Mika Guduru <sguduru@usc.edu>

Amir Hegazy <amirhega@usc.edu>

Amanda Zhang <amzhang@usc.edu>



Code
Demo