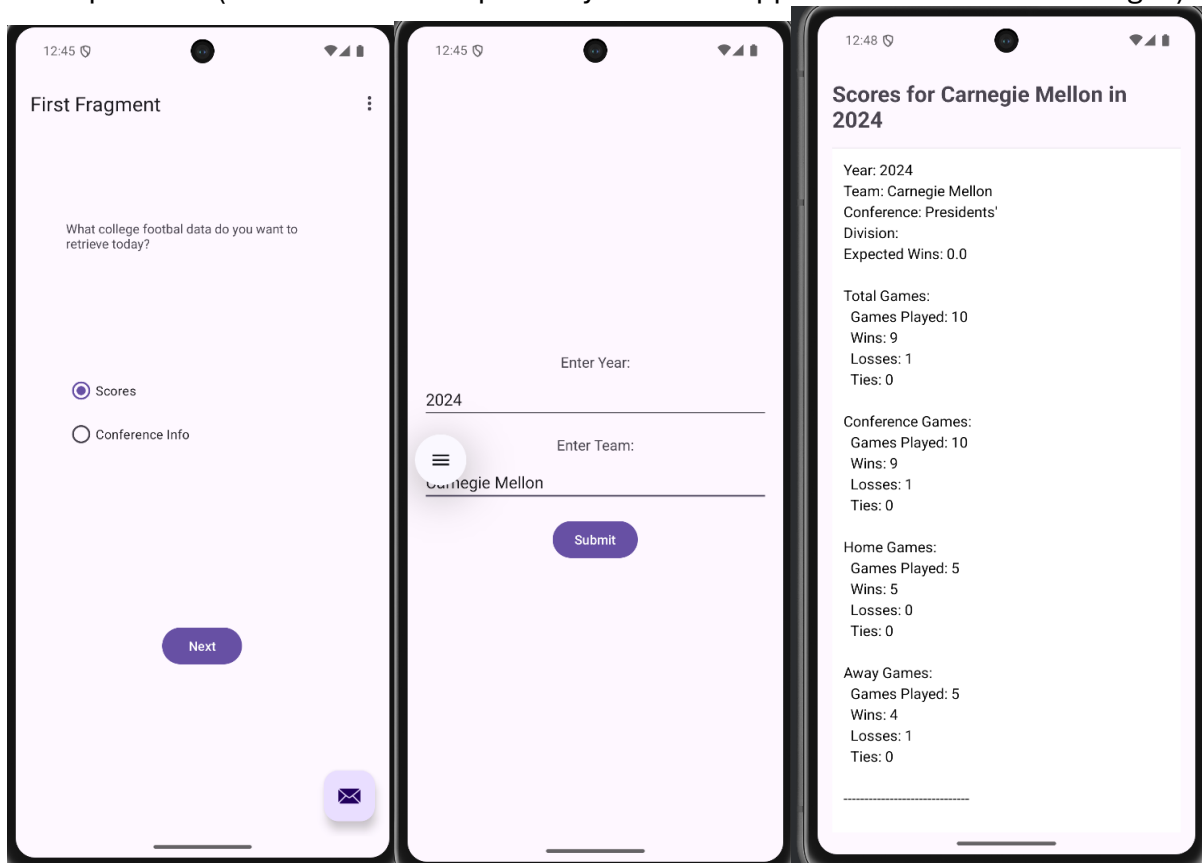


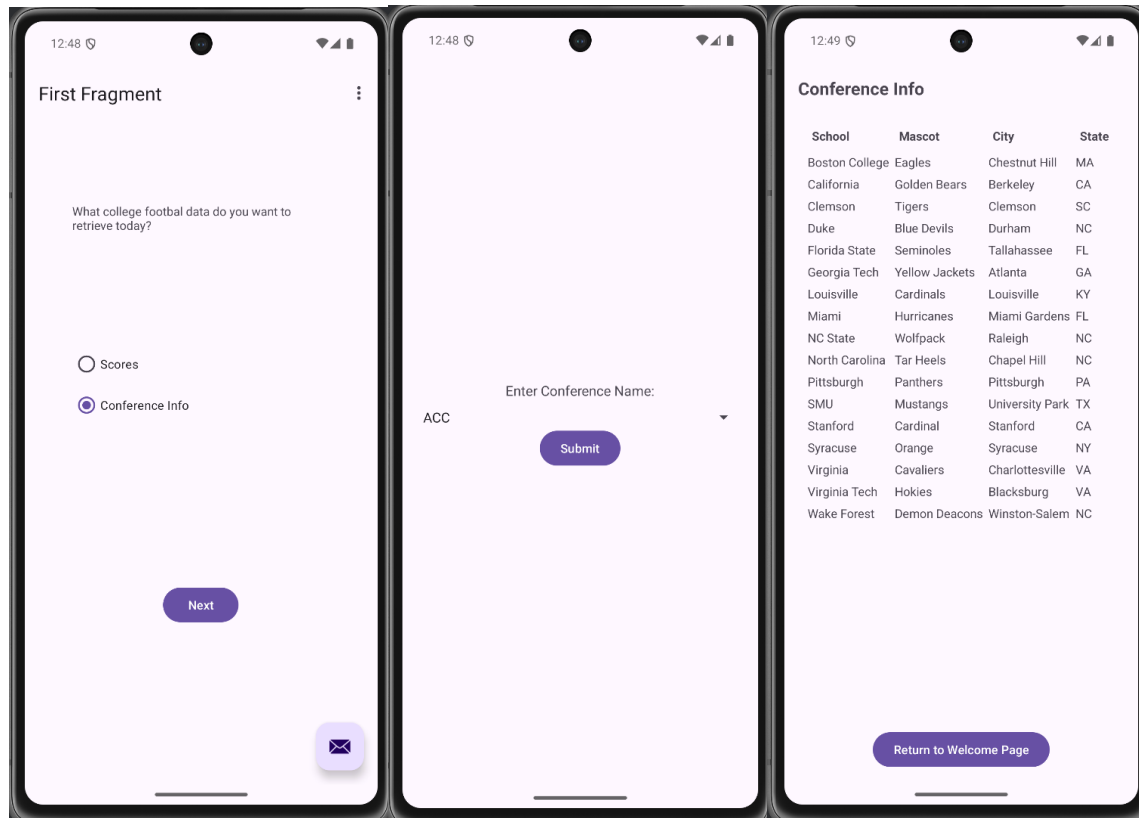
Project 4  
Haowen Weng  
Andrew ID: hweng

## Distributed Application Requirements

### 1. Implement a native Android application

- Has at least three different kinds of Views in your Layout (TextView, EditText, ImageView, or anything that extends android.view.View).
- Requires input from the user
- Makes an HTTP request (using an appropriate HTTP method) to your web service
- Receives and parses an XML or JSON formatted reply from your web service
- Displays new information to the user
- Is repeatable (i.e. the user can repeatedly reuse the application without restarting it.)





## 2. Implement a web service

```
private static final String CFB_API_URL = "https://api.collegefootballdata.com"; // 2 usages
private static final String API_KEY = "1gViricKRSebxrhgPLG/z1pIC729JqxThAFmohjvS7LnnFzCzT4jHaJ/c5gC9k0e"; // Replace with your CFB API key 2 usages

@Override // no usages
protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
    String type = request.getParameter("type");

    if (type == null || type.isEmpty()) {
        response.setStatus(HttpServletResponse.SC_BAD_REQUEST);
        response.getWriter().write("{\"error\": \"Request type is required\"}");
        return;
    }

    String phoneModel;
    String requestType;
    String requestDate;
    String userInputYear;
    String userInputTeam;
    String userInputConference;
    int statusCode;

    switch (type) {
        case "getScores":
            handleGetScores(request, response);
            // log data
            phoneModel = request.getHeader("User-Agent");
            requestType = "getScores";
            requestDate = LocalDateTime.now().toString();
            userInputYear = request.getParameter("year");
            userInputTeam = request.getParameter("team");
            statusCode = HttpServletResponse.SC_OK;
            // send to mongo
            MongoDB.logScoresRequestData(phoneModel, requestType, requestDate, userInputYear, userInputTeam, statusCode);
            break;
    }
}
```

```

private void handleGetScores(HttpServletRequest request, HttpServletResponse response) throws IOException {
    String year = request.getParameter("year");
    String team = request.getParameter("team");

    // Validate the input parameters
    if (year == null || year.isEmpty() || team == null || team.isEmpty()) {
        sendErrorResponse(response, HttpServletResponse.SC_BAD_REQUEST, "Year and team parameters are required.");
        return;
    }

    // Replace spaces in team name with '+'
    team = team.replace(" ", "+");

    // Construct the CFBD API URL for fetching scores
    String apiUrl = CFBD_API_URL + "/records?year=" + year + "&team=" + team;

    HttpURLConnection connection = null;

    try {
        // Set up the API connection
        URL url = new URL(apiUrl);
        connection = (HttpURLConnection) url.openConnection();
        connection.setRequestMethod("GET");
        connection.setRequestProperty("Authorization", "Bearer " + API_KEY);
        connection.setRequestProperty("Content-Type", "application/json");

        int statusCode = connection.getResponseCode();

        if (statusCode == HttpURLConnection.HTTP_OK) {
            // Read the API response
            Scanner scanner = new Scanner(connection.getInputStream());
            StringBuilder result = new StringBuilder();
            while (scanner.hasNextLine()) {
                result.append(scanner.nextLine());
            }
            scanner.close();

            // Parse the API response
            JSONArray scoresArray = new JSONArray(result.toString());
            JSONArray formattedScoresArray = new JSONArray();
        }
    } catch (IOException e) {
        // Handle exception
    }
}

```

a. Implement a simple (can be a single path) API.

**I used the CollegeFootballDataBase API to retrieve data.**

b. Receives an HTTP request from the native Android application

**The web servlet receive either GetScores or GetConferenceInfo HTTP request**

c. Executes business logic appropriate to your application. This includes fetching XML or JSON information from some 3rd party API and processing the response.

d. Replies to the Android application with an XML or JSON formatted response. The schema of the response can be of your own design.

**The servlet replies to the application with results from the API in the getScores/getConferenceInfo methods.**

### 3. Handle error conditions

## Web Service Logging and Analysis Dashboard Requirements

### 4. Log useful information

At least 6 pieces of information is logged for each request/reply with the mobile phone. It should include information about the request from the mobile phone, information about the request and reply to the 3rd party API, and information about the reply to the mobile phone. (You should NOT log data from interactions from the operations dashboard.)

```
_id: ObjectId('673c379357a5af52bd0052ce')
phoneModel : "Dalvik/2.1.0 (Linux; U; Android 15; sdk_gphone64_arm64 Build/AE3A.2408..."
requestType : "getConferenceInfo"
requestDate : "2024-11-19"
userInputConference : "ACC"
statusCode : 200
```

```
_id: ObjectId('673c37f757a5af52bd0052cf')
phoneModel : "Dalvik/2.1.0 (Linux; U; Android 15; sdk_gphone64_arm64 Build/AE3A.2408..."
requestType : "getScores"
requestDate : "2024-11-19"
userInputYear : "2024"
userInputTeam : "Carnegie Mellon"
statusCode : 200
```

## 5. Store the log information in a database

The web service can connect, store, and retrieve information from a MongoDB database in the cloud.

```
// log data
phoneModel = request.getHeader( s: "User-Agent");
requestType = "getScores";
requestDate = LocalDateTime.now().toString();
userInputYear = request.getParameter( s: "year");
userInputTeam = request.getParameter( s: "team");
statusCode = HttpServletResponse.SC_OK;
// send to mongo
MongoDB.logScoresRequestData(phoneModel,requestType,requestDate,userInputYear,userInputTeam,statusCode);

// log data
phoneModel = request.getHeader( s: "User-Agent");
requestType = "getConferenceInfo";
requestDate = LocalDateTime.now().toString();
userInputConference = request.getParameter( s: "conference");
statusCode = HttpServletResponse.SC_OK;
// send to mongo
MongoDB.logConferenceInfoRequestData(phoneModel,requestType,requestDate,userInputConference,statusCode);
```

```
public static List<Document> lastFiveScoresRequest(){ 1 usage
// Access the database
MongoDatabase database = mongoClient.getDatabase(s: "connectionLog"); // Replace with your database name

// Access the collection
MongoCollection<Document> collection = database.getCollection(s: "getScores"); // Replace with your collection name

// Query to fetch the last five requests
MongoCursor<Document> cursor = collection
    .find()
    .sort(new Document("requestDate", -1)) // Sort by requestDate descending
    .limit(l: 5)
    .iterator();

// Convert MongoCursor results to a List
List<Document> lastFiveRequests = new ArrayList<>();
while (cursor.hasNext()) {
    lastFiveRequests.add(cursor.next());
}

return lastFiveRequests;
}
```

The web sevlet is able to log user connections to the MongoDB, and there is also a mongoDB class handling storing and retrieving data from the database. I would retrieve data that are used for the dashboard(lastFiveScoresRequest, lastFiveConferenceRequest, numberOfTotalRequest, and Top used phone models).

6. Display operations analytics and full logs on a web-based dashboard

CFDB Logging and Analyzing

Category	Details
Number of Requests	21
Number of Conference Info Requests	15
Number of Scores Requests	6
Top Five Phone Models	1. Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/605.1.15 (KHTML, like Gecko) Version/16.6 Safari/605.1.15 (12 occurrences) 2. Dalvik/2.1.0 (Linux; U; Android 15; sdk_gphone64_arm64 Build/AE3A.240806.005) (9 occurrences)

Last Five Scores Requests


_id	Phone Model	Request Date	User Input year	User Input team	Status Code
673d780c5076952362e4ec32	Dalvik/2.1.0 (Linux; U; Android 15; sdk_gphone64_arm64 Build/AE3A.240806.005)	2024-11-20T05:47:55.817853028	2024	Carnegie Mellon	200
673d4edc5f39c2070a9ed88e	Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/605.1.15 (KHTML, like Gecko) Version/16.6 Safari/605.1.15	2024-11-19T21:52:12.797522	2024	carnegie mellon	200
673d4d475f39c2070a9ed88d	Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/605.1.15 (KHTML, like Gecko) Version/16.6 Safari/605.1.15	2024-11-19T21:45:27.251226	2024	carnegie mellon	200
673d4d405f39c2070a9ed88c	Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/605.1.15 (KHTML, like Gecko) Version/16.6 Safari/605.1.15	2024-11-19T21:45:20.280826	2024	carnegie	200
673c37f757a5af52bd0052cf	Dalvik/2.1.0 (Linux; U; Android 15; sdk_gphone64_arm64 Build/AE3A.240806.005)	2024-11-19	2024	Carnegie Mellon	200


Last Five Conference Info Requests

_id	Phone Model	Request Date	User Input Conference	Status Code
673d78555076952362e4ec33	Dalvik/2.1.0 (Linux; U; Android 15; sdk_gphone64_arm64 Build/AE3A.240806.005)	2024-11-20T05:49:09.683695129	ACC	200
673d6166eb59b927cd56ed79	Dalvik/2.1.0 (Linux; U; Android 15; sdk_gphone64_arm64 Build/AE3A.240806.005)	2024-11-20T04:11:18.263445899	ACC	200
673cc4a1b494bd51be9b2b06	Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/605.1.15 (KHTML, like Gecko) Version/16.6 Safari/605.1.15	2024-11-19	AAC	200
673cc75aae67f062f83cf860	Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/605.1.15 (KHTML, like Gecko) Version/16.6 Safari/605.1.15	2024-11-19	AAC	200
673cc49db494bd51be9b2b04	Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/605.1.15 (KHTML, like Gecko) Version/16.6 Safari/605.1.15	2024-11-19	ACC	200


- a. A unique URL addresses a web interface dashboard for the web service.
- b. The dashboard displays at least 3 interesting operations analytics.
- c. The dashboard displays **formatted** full logs.

## 7. Deploy the web service to GitHub Codespaces








 distributed-systems-project-04-hwweng-cmu Private Watch 1

 **Help us improve GitHub Codespaces**  
Tell us how to make GitHub Codespaces work better for you with three quick questions.

main 1 Branch 0 Tags  t Add file <> Code

 **hwweng-cmu** Add files via upload

838e324 · 21 minutes ago 14 Commits

 Dockerfile	Update Dockerfile	1 hour ago
 Project4APP.zip	Add files via upload	21 minutes ago
 Project4WebServlet-1.0-SNAPSHOT.war	Add files via upload	2 hours ago
 Project4WebServlet.zip	Add files via upload	21 minutes ago
 README.md	Add files via upload	2 days ago
 ROOT.war	Add files via upload	21 minutes ago
 devcontainer.json	Add files via upload	5 hours ago