**Sports Event** : It validates the username and password for the authentication process.

package com.sportsevent.backend.entity;

import java.sql.Date;

import java.util.List;

public class Event {

private String id;

private String name;

private String location;

private String time;

private Date date;

private String eventType;

private List<Participant> participants;

public String getId() {

return id;

}

public void setId(String id) {

this.id = id;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public String getLocation() {

return location;

}

public void setLocation(String location) {

this.location = location;

}

public String getTime() {

return time;

}

public void setTime(String time) {

this.time = time;

}

public Date getDate() {

return date;

}

public void setDate(Date date) {

this.date = date;

}

public String getEventType() {

return eventType;

}

public void setEventType(String eventType) {

this.eventType = eventType;

}

public List<Participant> getParticipants() {

return participants;

}

public void setParticipants(List<Participant> participants) {

this.participants = participants;

}

}

**Event Control:**

It allows users to identifying the upcoming and ongoing events in an organization.

package com.sportsevent.backend.rest;

import java.util.List;

import java.util.Map;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.http.HttpStatus;

import org.springframework.http.ResponseEntity;

import org.springframework.stereotype.Controller;

import org.springframework.web.bind.annotation.CrossOrigin;

import org.springframework.web.bind.annotation.GetMapping;

import org.springframework.web.bind.annotation.PostMapping;

import org.springframework.web.bind.annotation.RequestBody;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.web.bind.annotation.RequestParam;

import com.sportsevent.backend.entity.Event;

import com.sportsevent.backend.entity.User;

import com.sportsevent.backend.service.EventService;

import com.sportsevent.backend.service.UserService;

//@CrossOrigin(origins = "http://localhost:3000")

@Controller

@RequestMapping("/sports")

public class EventController {

@Autowired

EventService eventService;

public EventController(EventService eventService) {

this.eventService = eventService;

}

@GetMapping(value = "/events/ongoing")

ResponseEntity<List<Event>> verifyOngoingEvents(){

return ResponseEntity.status(HttpStatus.OK).body(eventService.getEventDetails());

}

@GetMapping(value = "/events/upcoming")

ResponseEntity<List<Event>> verifyUpcomingEvents(){

return ResponseEntity.status(HttpStatus.OK).body(eventService.getUpcomingEventDetails());

}

}

**Event Service:**

package com.sportsevent.backend.service;

import java.util.List;

import com.sportsevent.backend.entity.Event;

public interface EventService {

public List<Event> getEventDetails();

public List<Event> getUpcomingEventDetails();

}

**Event Type:**

It allows users to identifying the upcoming and ongoing event locations in an organization.

package com.sportsevent.backend.service;

import java.sql.Date;

import java.sql.PreparedStatement;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.util.ArrayList;

import java.util.Comparator;

import java.util.HashMap;

import java.util.List;

import java.util.Map;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Service;

import com.sportsevent.backend.entity.Event;

import com.sportsevent.backend.entity.Participant;

import com.sportsevent.backend.entity.User;

@Service

public class EventServiceImpl implements EventService{

private SportsEventComponent sportsEventComponent;

@Autowired

public EventServiceImpl(SportsEventComponent sportsEventComponent) {

this.sportsEventComponent = sportsEventComponent;

}

@Override

public List<Event> getEventDetails() {

List<Event> eventList = new ArrayList<Event>();

String participantSelect = "Select \* from participant";

Map<String,List<Participant>> participantMap = new HashMap<>();

try {

PreparedStatement participantSelectStatement = sportsEventComponent.getConnection().prepareStatement(participantSelect);

ResultSet rs = participantSelectStatement.executeQuery();

while(rs.next()) {

String id = rs.getString("id");

String name = rs.getString("name");

String eventId = rs.getString("event\_id");

Participant participant = new Participant();

participant.setId(id);

participant.setName(name);

participant.setEventId(eventId);

List<Participant> participantList = new ArrayList<Participant>();

if(participantMap.containsKey(eventId)) {

participantList = participantMap.get(eventId);

}

participantList.add(participant);

participantMap.put(eventId,participantList);

}

String selectEvent = "Select \* from event where event\_Type="+"'ONGOING'";

PreparedStatement eventStatement = sportsEventComponent.getConnection().prepareStatement(selectEvent);

ResultSet resultSet = eventStatement.executeQuery();

while(resultSet.next()) {

String id = resultSet.getString("id");

String name = resultSet.getString("name");

String location = resultSet.getString("location");

String time = resultSet.getString("time");

String eventType = resultSet.getString("event\_type");

Date date = resultSet.getDate("date");

List<Participant> participants = participantMap.get(id) == null ?

new ArrayList<Participant>() : participantMap.get(id);

Event event = new Event();

event.setId(id);

event.setName(name);

event.setLocation(location);

event.setTime(time);

event.setEventType(eventType);

event.setDate(date);

event.setParticipants(participants);

eventList.add(event);

}

} catch (SQLException e) {

e.printStackTrace();

}

return eventList;

}

@Override

public List<Event> getUpcomingEventDetails() {

List<Event> eventList = new ArrayList<Event>();

String participantSelect = "Select \* from participant";

Map<String,List<Participant>> participantMap = new HashMap<>();

try {

PreparedStatement participantSelectStatement = sportsEventComponent.getConnection().prepareStatement(participantSelect);

ResultSet rs = participantSelectStatement.executeQuery();

while(rs.next()) {

String id = rs.getString("id");

String name = rs.getString("name");

String eventId = rs.getString("event\_id");

Participant participant = new Participant();

participant.setId(id);

participant.setName(name);

participant.setEventId(eventId);

List<Participant> participantList = new ArrayList<Participant>();

if(participantMap.containsKey(eventId)) {

participantList = participantMap.get(eventId);

}

participantList.add(participant);

participantMap.put(eventId,participantList);

}

String selectEvent = "Select \* from event where event\_Type="+"'UPCOMING'";

PreparedStatement eventStatement = sportsEventComponent.getConnection().prepareStatement(selectEvent);

ResultSet resultSet = eventStatement.executeQuery();

while(resultSet.next()) {

String id = resultSet.getString("id");

String name = resultSet.getString("name");

String location = resultSet.getString("location");

String time = resultSet.getString("time");

String eventType = resultSet.getString("event\_type");

Date date = resultSet.getDate("date");

List<Participant> participants = participantMap.get(id) == null ?

new ArrayList<Participant>() : participantMap.get(id);

Event event = new Event();

event.setId(id);

event.setName(name);

event.setLocation(location);

event.setTime(time);

event.setEventType(eventType);

event.setDate(date);

event.setParticipants(participants);

eventList.add(event);

}

} catch (SQLException e) {

e.printStackTrace();

}

return eventList;

}

}

**Participant:**

It allows users to identifying the team members for particular sports using the discussion forum by posting the requirements.

**package** com.sportsevent.backend.entity;

**public** **class** Participant {

**private** String id;

**private** String name;

**private** String eventId;

**public** String getId() {

**return** id;

}

**public** **void** setId(String id) {

**this**.id = id;

}

**public** String getName() {

**return** name;

}

**public** **void** setName(String name) {

**this**.name = name;

}

**public** String getEventId() {

**return** eventId;

}

**public** **void** setEventId(String eventId) {

**this**.eventId = eventId;

}

}

**Event DBconnection:**

package com.sportsevent.backend.service;

import java.sql.Connection;

import org.springframework.stereotype.Component;

import com.sportsevent.backend.database.DBConnection;

@Component

public class SportsEventComponent {

private Connection conn;

public SportsEventComponent() {

this.conn = DBConnection.createNewDBconnection();

}

public Connection getConnection() {

return conn;

}

}