* ~~Athlete sends join request to team admin~~

Athlete Service:

public void askJoin (Athlete athlete,String teamName){

TeamAdmin teamAdmin=teamAdminRepository.findTeamAdminByteamName(teamName);

if (teamAdmin == null){

throw new ApiException("Invalid Team Name");

}

athlete.setStatus("Pending");

athlete.setTeamAdmin(teamAdmin);

athleteRepository.save(athlete);

}

Athlete Controller:

@PostMapping("/join/{teamName}")

public ResponseEntity askJoin(@RequestBody @Valid Athlete athlete,@PathVariable String teamName){

athleteService.askJoin(athlete,teamName);

return ResponseEntity.*status*(200).body(new ApiResponse("send your request successful"));

}

* ~~Athletes views requests - ADD TO FIGMA~~

Class Service: Athlete

public Athlete viewRequest(Integer athleteId){

Athlete athlete=athleteRepository.findAthleteByAthleteId(athleteId);

return athlete;

}

Class Controller: Athlete

@GetMapping("/view-request/{athleteId}")

public ResponseEntity viewDetiels(@PathVariable Integer athleteId){

return ResponseEntity.*status*(200).body(athleteService.viewRequest(athleteId));

}

* ~~Athletes views all teams~~

Class Service : FanService , AthleteService , Team Admin:

public List<String> getAllTeams(){

List<TeamAdmin>teamAdmins=teamAdminRepository.findAll();

List<String>teams=new ArrayList<>();

for(TeamAdmin team:teamAdmins){

teams.add("Name: "+team.getTeamName()+" Number of Athletes: "+team.getAthleteSet().size());

}

return teams;

}  
  
Class Controller:  
 Fan:  
public ResponseEntity getAllTeams(){

return ResponseEntity.*status*(200).body("The Teams is:"+fanService.getAllTeams());

}

Athlete:

@GetMapping("/view-teems")

public ResponseEntity getAllTeams(){

return ResponseEntity.*status*(200).body("The Teams is:"+athleteService.getAllTeams());

}

TeamAdmin:

@GetMapping("/view-teems")

public ResponseEntity getAllTeams(){

return ResponseEntity.*status*(200).body("The Teams is:"+teamAdminService

.getAllTeams());

}

* ~~Search athlete by name~~

Athlete Service:

public String SearchAtheletbyName(String AtherteName){

Athlete athlete=athleteRepository.findAthleteByName(AtherteName);

return " Name: "+athlete.getName()+" Sport: "+athlete.getSport()+" Weight: "+athlete.getWeight()+" Height: "+athlete.getHeight()+" Category: "+athlete.getCategory()+" "+athlete.getSport()+" Achievements: "+athlete.getAchievements()+" Gender: "+athlete.getGender()+

" Age: "+athlete.getAge()+" LicenseNumber: "+athlete.getLicenseNumber();

}

Athlete Controller:

@GetMapping("/by-name/{name}")

public ResponseEntity SearchAthletetbyName(@PathVariable String name){

return ResponseEntity.*status*(200).body(athleteService.SearchAtheletbyName(name));

}

* ~~Get all achievements~~

Athlete Service:

public Integer getAtchevments(Integer athleteId){

Athlete athlete=athleteRepository.findAthleteByAthleteId(athleteId);

return athlete.getAchievements();

}

Athlete Controller:

@GetMapping("/get-achievements/{athleteId}")

public ResponseEntity getAtchevments(@PathVariable Integer athleteId){

return ResponseEntity.*status*(200).body("Achievements is: "+athleteService.getAtchevments(athleteId));

}

* ~~Team admin searches all sports in the team~~

Team Admin Service:  
 public List searchAllSportsInTeam(Integer teamAdminId){

TeamAdmin teamAdmin=teamAdminRepository.findTeamAdminByTeamId(teamAdminId);

List<SportAdmin> sport=sportAdminRepository.findAll();

List<String> sports=new ArrayList<>();

if(teamAdmin==null){

throw new ApiException("Invalid");

}else

for(SportAdmin sport1:sport){

sports.add(sport1.getSportType());

}

return sports;

}

Team Admin Controller:

@GetMapping("/search-all-sports/{teamId}")

public ResponseEntity searchAllSportsInTeam(@PathVariable Integer teamId){

return ResponseEntity.*status*(200).body("Sports: "+teamAdminService.searchAllSportsInTeam(teamId));

}

* ~~Team admin adds athlete achievements~~

Athlete Model:

// @NotNull(message = "License Number should be not empty")

@Column(columnDefinition = "int not null unique ")

private Integer licenseNumber;

Team Admin Service:

public void addAtchevments(Integer liceneOfAthlete,Integer teamAdminId){

Athlete athlete=athleteRepository.findAthleteByLicenseNumber(liceneOfAthlete);

TeamAdmin teamAdmin=teamAdminRepository.findTeamAdminByTeamId(athlete.getTeamAdmin().getTeamId());

if(teamAdminId!=teamAdmin.getTeamId()){

throw new ApiException("Invalid");

}else

athlete.setAchievements(athlete.getAchievements()+1);

athleteRepository.save(athlete);

}

Team Admin Controller:  
@PutMapping("/add-Atchevments/{liceneOfAthlete}/{teamAdminId}")

public ResponseEntity addAtchevments(@PathVariable Integer liceneOfAthlete,@PathVariable Integer teamAdminId){

teamAdminService.addAtchevments(liceneOfAthlete,teamAdminId);

return ResponseEntity.*status*(200).body(new ApiResponse("Atchevments Successfully"));

}

* ~~Sport admin view requests- Sport Admin~~

SportAdmin Service:  
public List<String> vieweRequest(Integer sportAdminId){

SportAdmin sportAdmin=sportAdminRepository.findSportAdminByAdminId(sportAdminId);

if(sportAdmin == null) {

throw new ApiException("Invalid SportAdmin ID");

}

List<TeamAdmin>teamAdmin=teamAdminRepository.findTeamAdminByStatus("Pending");

List<String>teamAdmins=new ArrayList<>();

for(TeamAdmin teamAdmin1: teamAdmin) {

teamAdmins.add("Name: "+

teamAdmin1.getTeamAdminName()+

" Status: "+teamAdmin1.getStatus()+

" Team Name: "+teamAdmin1.getTeamName()+

" Email: "+teamAdmin1.getTeamEmail()+

" Phone Number: "+teamAdmin1.getPhoneNumber());

}

return teamAdmins;

}

Sport Admin Controller:

@GetMapping("/view-requests/{sportAdminId}")

public ResponseEntity vieweRequest(@PathVariable Integer sportAdminId){

return ResponseEntity.*status*(200).body(sportAdminService.vieweRequest(sportAdminId));

}

* ~~Sport admin approve requests- Sport Admin~~

SportAdmin Service:

public void ApproveRequests(Integer sportid,String sportType ,String teamName) {

Requests requests = requestsRepository.findRequestsBySportType(sportType);

SportAdmin sportAdmin = sportAdminRepository.findSportAdminByAdminId(sportid);

TeamAdmin teamAdmin=teamAdminRepository.findTeamAdminByteamName(teamName);

if (requests.getSportType().equals(sportAdmin.getSportType())){

requests.setStatus("Accepted");

sportAdmin.getTeamAdminSet().add(teamAdmin);

requestsRepository.save(requests);

sportAdminRepository.save(sportAdmin);

teamAdminRepository.save(teamAdmin);

}

}

Sport Admin Controller:

@PutMapping("/approve-request/{sportAdminId}/{teamName}/{sportType}")

public ResponseEntity ApproveRequests(@PathVariable Integer sportAdminId,@PathVariable String teamName ,@PathVariable String sportType){

sportAdminService.ApproveRequests(sportAdminId,sportType,teamName);

return ResponseEntity.*status*(200).body("approve Requests");

}