Web Application Project – Semester 2, 2021

Due: 5pm Wednesday 3rd November 2021

Worth: 35% of COMP636 marks

Scenario

A small community rugby club has approached you to develop a member management system for them.

They have the following functional requirements:

There must be two versions of the interface:

1. The first, for a club member to use, should allow members to check and update their personal details, and see upcoming fixtures (games) that they will be playing in. It should also show the latest news announcements posted by the club.
2. The second interface, for club administrators, allows them to edit member details, assign members to teams, assign teams to games, and post news announcements for club members.

In this version of the system, there is no requirements for users to log-in using username/password. They can simply select their name from a drop-down list.

Using the first interface (members) active club members should be able to:

- See the latest three news announcements from the club
- Check and update their contact details
- See the list of upcoming fixtures for their team

Using the second interface (admin) club administrators should be able to:

- Add news items
- Add a new member to the club
- Edit the details of a member
- Change the membership status of a member (Active/Inactive)
- See a list of all active members
- Add a new team to the club
- Add a new (opposition) team to another club
- Assign a member to a team (a member can belong to only one team)
- Assign a team to a fixture against another team
- Print a report showing Grade eligibility (described below)

General requirements

The user login interface should be displayed when the default route is accessed (/)

Users (members) should then select their name from a drop-down box and be re-directed to the main interface. Only active members are listed in the drop-down box.

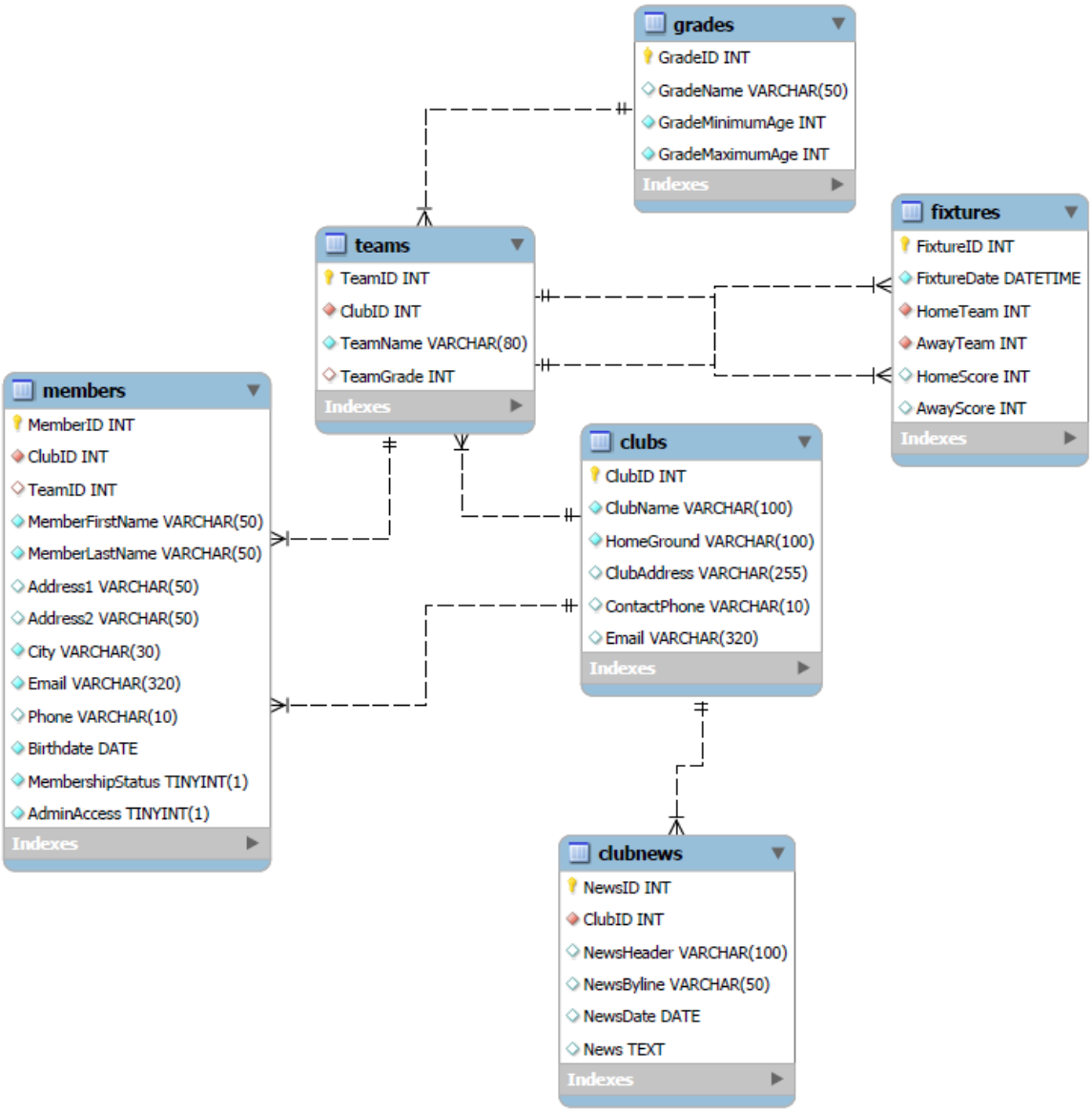
Users should be shown menu items that relate to their role (i.e. admin or standard member).

You are provided with tables (rugby.sql) that implement the following data model.

You are also provided with some example data (rugbydata.sql) that you should load.

Be sure to read the Project Hints at the end of this document.

Data Model



This diagram is also provided as a PDF in the project files.

The two key tables are members and clubs. The club has an ID and various information about the club including the Home Ground where games are played and contact details. The table also stores data about other clubs, to allow fixtures to be shown.

Members stores information about each member of a club. They have a unique ID (MemberID) and belong to a club and optionally, a team (note, team must be from the same club they are a member of!). Members do not have to belong to a team but must belong to a club.

There are various contact details for the member along with their birthdate, membership status (1 is active, 0 is inactive), and if they are an Admin (AdminAccess, 1 for yes, 0 for no (default)).

Each Team belongs to a Grade, and a Club. Each Grade has a minimum Age and a maximum Age for players in it. This is used for calculating Player Eligibility (see below).

Fixtures stores information on the teams that are playing. Only teams that are of the same grade can play against each other.

Clubnews stores news articles for the club.

There are two relationships between teams and fixtures, one for a team's home games and one for its away games.

Calculating Eligibility Requirements

Each Grade has a minimumAge and a maximumAge specified. Players eligible to play in that grade are those whose age on the date entered are between these values (inclusive).

E.g. Senior Grade – minimumAge = 18, maximumAge = 99

Members

James Smith Birthdate 27/July/1990

Paul Edwards Birthdate 3/January/2003

Eligibility Date 1/January/2021

(Eligibility date is entered into the system when the Eligibility Report run, it can be any date)

Result

James Smith

Explanation

Paul Edwards was 17 and 363 days old on 1st January 2021, so was not yet 18 and therefore is too young to play in the senior grade this year.

Project Approach Requirements

- Use the provided rugby.sql file to create the database within your MySQL database on AWS & pythonAnywhere. Also run the rugbydata.sql file to load some initial data.
- Create a Flask Webapp that:
    - meets the functional requirements
    - is appropriately commented
    - connects to your database
    - provides appropriate routes for the different functions
    - provides templates & incorporates HTML forms to take input data
    - uses Bootstrap CSS to provide styling and formatting
- Create a GitHub repository to store your code
- Host your Webapp on pythonAnywhere
- Include a report document outlining the structure of your solution (routes & functions) and detailing any assumptions and design decisions that you have made. This report must be created using GitHub Markdown and saved in the README.md file of your GitHub repository.

Submission

You must

a) Create a private github repository that contains
    a. All python, HTML, image and any other required files for the web app
    b. A requirements.txt file showing the required pip packages.
    c. An extract of your database (schema and data)
    d. Your project report
    e. Your repository should have a .gitignore file and therefore not have a copy of your virtual environment.
b) Add lincolnmac (computing@lincoln.ac.nz) as a collaborator to your private github repository.
c) Create your report document as a the README.md document as part of your github repository
d) Host your system (including database) using pythonAnywhere and add lincolnmac as your "teacher" via the site configuration.
e) Submit your pythonAnywhere URL via the link on the Learn COMP636 page.

Marking

Your project is expected to reflect the work of a professional. This includes colour and styling choices, and the level of commenting present in your code.

General Project Aspects:

Project Element	Marks Available
Project Report: • outlining the structure of your solution (routes & functions) • detailing any assumptions and design decisions that you have made.	20 marks • Note: If Assumptions and Design Decisions are not included, then maximum mark available is 8.

• report created using GitHub Markdown and saved in the README.md file of your GitHub repository	• Up to 3 marks of these 20 are awarded for spelling, punctuation, grammar and presentation
GitHub Repository Setup, and shared pythonAnywhere Hosting, and WebApp correctly configured, including database setup	5 marks
Consistent 'Look and Feel' (Bootstrap styling & templates)	5 marks

Functional Project Aspects

Within each of these functional areas we are looking at and for:

- The functionality specified works
- Well commented HTML, SQL, and Python. Well formatted code throughout.
- Data Validation on Forms.
- Functionally correct python.
- Well-structured SQL Queries.
- Appropriate naming, both of variables and labels.
- Appropriate and well formatted content

Functionality	Expectations	Marks Available
Login	Active member selects their name from the drop-down list. Administrators see admin functionality in their menus, normal users do not	6
Member Dashboard (screen just after login)	Shows latest three news announcements. Shows the upcoming fixtures for the team the member plays for.	6
Member Update Details	Member can update their details but can not modify primary keys	5
Admin News	Administrator can add a new news item to the system and see all news items	6
Add/Edit Member	A new member can be added by administrator. Existing members details can be edited. (Appropriate interface to find members to edit)	10
Member Status	De-activate existing members Re-activate inactive members (Appropriate interface to find members to re-activate/de-activate)	3
Report of all active members	A list of all active members in a user-friendly format suitable for printing	5
Add a new team to the club	Add a new team to the club	3
Add a new opposition team	New opposition team can be added to the system	3
Team assignment	Join a member to a team (they can be a member of only one	3

	team, a team from their own club)	
Fixtures	Create a home fixture where a team from the club is the home team. Create an away fixture where another team is the home team. Teams must be of the same grade.	10
Eligibility Report	A list of all active members, in a format suitable for printing, who are eligible to play in a grade as at an entered date.	10

Functionality not required

You are not required to

- Add or modify club information in the system
- Add scores for fixtures
- Edit existing team information in the system
- Delete any records from the database.

These features may be desirable in a full system, but are outside the scope of this assessment.

Project Hints

Create your GitHub repository first and create all your required code/files in your local folder.

Regularly commit and push changes to your GitHub repository.

pythonAnywhere is case sensitive so test your app early – we will mark the pythonAnywhere version of your app.

Spend some time sketching the structure of your application before you start developing. Think about which features could share the same (or nearly the same) templates. Remember you can nest templates.