Your client Komodo Insurance has a number of software development teams and needs you to write a Console App to help their Product Managers manage them all.

**Here is the GitHub Repository where we have gotten your environment started for you: [DevTeamsRepo (Links to an external site.)](https://github.com/WilsoC14/DevTeamsProject" \t "_blank)**

**You will need to download the Repository from that link, extract it, and then navigate to the extracted file to open the solution.**

(We will link the video from class where Michael walks you through downloading the repository)

**We have created 4 classes for you.**  
**- Two POCOS:**Developer, DevTeam  
**- Two Repositories:**DeveloperRepo, DevTeamRepo

**Info from Instructors:**  
You will need to give the POCOs properties and and Repos need CRUD methods as appropriate. You will then create a User Interface (Console Application) to build the functionality Komodo Insurance requires utilizing these classes. We have provided you with a few fields in the appropriate classes to help you in writing the CRUD Methods for the Developer and DevTeam POCOs.

You need to adhere to the guidelines from Komodo Insurance while writing their application to meet their needs in managing the DevTeams and Developers. Clients will typically not give you all the info you need, so you may need to use your brain and peers to fill in the blanks!

**Info from Komodo Insurance:**  
Developers have names and ID numbers; we need to be able to identify one developer without mistaking them for another. We also need to know whether or not they have access to the online learning tool: Pluralsight.

**Challenge**: Our HR Department uses the software monthly to get a list of all our Developers that need a Pluralsight license. Create this functionality in the Console Application

Our managers need to be able to add and remove members to/from a team by their unique identifier. They should be able to see a list of existing developers to choose from and add to existing teams. Odds are, the manager will create a team, and then add Developers individually from the Developer Directory to that team.

**Challenge:** Some of our managers are nitpicky and would like the functionality to add multiple Developers to a team. Integrate this into your application.

Your client Komodo Insurance has a number of software development teams and needs you to write a Console App to help their Product Managers manage them all.

**Here is the GitHub Repository where we have gotten your environment started for you: [DevTeamsRepo (Links to an external site.)](https://github.com/WilsoC14/DevTeamsProject" \t "_blank)**

**You will need to download the Repository from that link, extract it, and then navigate to the extracted file to open the solution.**

Link to Michaels Walkthrough of downloading Repository: [Link (Links to an external site.)](https://zoom.us/rec/share/XqZugWKLpW2yjDRxFdRfBKu8bolSLU82qJ9Qx8kLnfI-nHfsv5amLJFPq7x7d4MD.TLyazaGzSnAF9H7h)  
Passcode: 66FQvWi$

**We have created 4 classes for you.**  
**- Two POCOS:**Developer, DevTeam  
**- Two Repositories:**DeveloperRepo, DevTeamRepo

**Info from Instructors:**  
You will need to give the POCOs properties and and Repos need CRUD methods as appropriate. You will then create a User Interface (Console Application) to build the functionality Komodo Insurance requires utilizing these classes. We have provided you with a few fields in the appropriate classes to help you in writing the CRUD Methods for the Developer and DevTeam POCOs.

You need to adhere to the guidelines from Komodo Insurance while writing their application to meet their needs in managing the DevTeams and Developers. Clients will typically not give you all the info you need, so you may need to use your brain and peers to fill in the blanks!

**Info from Komodo Insurance:**  
Developers have names and ID numbers; we need to be able to identify one developer without mistaking them for another. We also need to know whether or not they have access to the online learning tool: Pluralsight.

**Challenge**: Our HR Department uses the software monthly to get a list of all our Developers that need a Pluralsight license. Create this functionality in the Console Application

Our managers need to be able to add and remove members to/from a team by their unique identifier. They should be able to see a list of existing developers to choose from and add to existing teams. Odds are, the manager will create a team, and then add Developers individually from the Developer Directory to that team.

**Challenge:** Some of our managers are nitpicky and would like the functionality to add multiple Developers to a team at once, rather than one by one. Integrate this into your application.  
  
  
\*\* Attention: Memo from the boss \*\*

Hi Team,  
Your Senior Dev made a mistake when scaffolding out the DevTeamRepo.cs file. This person even left a message in the comments trying to guide you down a path that will ultimately lead to many headaches. This happens sometimes, and you'll need to be reactive to these situations. He left explicit instructions for you to remedy this error so you can move on with your life. I've copied those instructions below:

**Attention all Developers working on the Komodo Insurance Developer Team Management Application:**  
You must remove the field: \_developerRepo from the DevTeamRepo class.

You will need to access your pool of Developers from outside of the DevTeamRepo class; utilize parameters to pass in Developer objects when adding or removing Developers from a Team.

## Rubric

**Developer Team Application**

| Developer Team Application | | |
| --- | --- | --- |
| **Criteria** | **Ratings** | **Pts** |
| This criterion is linked to a Learning OutcomeSolution Structure | |  |  |  | | --- | --- | --- | | **5.0 to >3.33 pts**  **Complete**  The solution structure met requirements along with having organized assemblies that lent to the project having an overall cohesiveness. | **3.33 to >0.0 pts**  **Basic**  The solution structure met requirements, though with unorganized assemblies. | **0.0 pts**  **Incomplete**  The solution structure did not meet the requirements. | | 5.0 pts |
| This criterion is linked to a Learning OutcomeFunctional Code  Project Compiles/Runs when cloned or downloaded. | |  |  |  | | --- | --- | --- | | **5.0 to >2.5 pts**  **Proficient**  Code runs when cloned with no changes required. | **2.5 to >0.0 pts**  **Basic**  Code runs when cloned after a few minor changes. | **0.0 pts**  **Incomplete**  Code does not run when cloned/ needs to many changes to run after being cloned | | 5.0 pts |
| This criterion is linked to a Learning OutcomeRepository  Repository Functionality (Back End). This includes creation and implementation of required functionality. | |  |  |  |  | | --- | --- | --- | --- | | **20.0 to >16.0 pts**  **Proficient**  All CRUD Methods for Developer functional and written in DeveloperRepo.cs file. All CRUD Methods for DevTeam functional and written in DevTeamRepo.cs file. | **16.0 to >8.0 pts**  **Lacking Functionality**  All CRUD Methods for Developer functional and written in Developer Repo.cs file. Some CRUD Methods for DevTeam functional and written in DevTeamRepo.cs file. | **8.0 to >4.0 pts**  **Not Functional: Developers but no DevTeams**  All CRUD Methods for Developer functional and written in DeveloperRepo.cs file. | **4.0 to >0 pts**  **Incomplete**  Student added properties to Developer and DevTeam, but did not successfully write CRUD methods for either object. | | 20.0 pts |
| This criterion is linked to a Learning OutcomeConsole Application  Console Application User Interface (UI) Functionality (Front End). This includes creation and implementation of required functionality. | |  |  |  |  | | --- | --- | --- | --- | | **15.0 to >12.0 pts**  **Proficient**  Student completed the required functionality for the UI portion. Application runs with no issues. Business Requirements are successfully met. UI makes sense, and navigation for intended user is professional. Users do not know the innerworkings of the application: Application prompts the users for proper input. | **12.0 to >6.0 pts**  **Lacking User Experience**  Student completed most of the required functionality for the UI portion and/or the application runs with a few issues. App is functional, but users cannot easily navigate through it to meet their needs. UI lacks informative prompts and occasionally has a blank screen or wrong information preventing the user from fully utilizing the application. | **6.0 to >0.0 pts**  **In Progress**  Student completed half or less of the required functionality for the UI portion and/or the application has clear bugs/issues running. Application is in no way ready for the client. | **0.0 pts**  **Incomplete**  Student did not complete the required functionality for the UI portion and/or the application does not run. | | 15.0 pts |
| This criterion is linked to a Learning OutcomeCode Cleanliness  Naming Conventions, File Structure, White Space | |  |  |  | | --- | --- | --- | | **5.0 to >3.33 pts**  **Proficient**  Student left no extra white space as well as maintained appropriate naming conventions for variables, methods and classes. | **3.33 to >1.67 pts**  **Basic**  Student left little extra white space and/or maintained mostly consistent naming conventions for variables, methods and classes. | **1.67 to >0 pts**  **Incomplete**  Student left a lot of extra white space and/or had inconsistent naming conventions for variables, methods and classes. | | 5.0 pts |
| This criterion is linked to a Learning OutcomeExtra Credit  Place for any possible Extra Credit, if applicable. | |  |  | | --- | --- | | **15.0 to >0.0 pts**  **Extra Credit**  Challenges from prompt implemented in project. Extraordinarily clean, modular, and functional code with extensive validation. UI is clean and allows for great User Experience. All in all, if your app is stellar, you'll get 15 extra credit points. | **0.0 pts**  **No Marks**  No Extra Credit. | | 15.0 pts |
| Total Points: 65.0 | | |