Faculty of Engineering, Built Environment and Information Technology

Fakulteit Ingenieurswese, Bou-omgewing en Inligtingtegnologie / Lefapha la Boetšenere, Tikologo ya Kago le Theknolotši ya Tshedimošo

UNIVERSITEIT VAN PRETORIA
UNIVERSITY OF PRETORIA
YUNIBESITHI YA PRETORIA

DEPARTMENT OF INFORMATICS

INF 354

SEMESTER TEST (1) **DATE: 2021-05-31**

Examiners	:	Dr. JP van Deventer Mr. R Hanslo Mr. B Rampete	**Time**	:	180 min
Moderator / External Examiner	:	Dr Johan Breytenbach University of the Western Cape	**Marks**	:	30

Student Number								Surname	Initials

Question Section	Module outcomes (as in Study Guide)							Marks allocated	Maximum mark
	MO1	MO2	MO3	MO4	MO5	MO6	MO7		
Section A			X	X					**10**
Section B					X				**10**
Section C		X		X					**10**
Total									**30**

Instructions

1. This paper consists of 3 sections with several main questions (sub-sets of instructions) each.
2. Each section relates to a small semi-complete program that needs to be updated or finalised.
3. Each question relates to a file in one of the programs that you need to update or finalise.
4. Each sub-sets of instructions relates to activities and tasks in one of the files.
5. Each main question relates to a file that needs to be edited in one of the three programs you have been provided with.
6. Answer all the questions – there are no optional questions.
7. Please read all questions, instructions and sub-sets of tasks very carefully.
8. After completing work on a relevant question, please upload ONLY the edited file to the correct upload area.

The University of Pretoria commits itself to produce academic work of integrity. I affirm that I am aware of and have read the Rules and Policies of the University, more specifically the Disciplinary Procedure and the Tests and Examinations Rules, which prohibit any unethical, dishonest or improper conduct during tests, assignments, examinations and/or any other forms of assessment. I am aware that no student or any other person may assist or attempt to assist another student, or obtain help, or attempt to obtain help from another student or any other person during tests, assessments, assignments, examinations and/or any other forms of assessment.

SECTION A – WEB API AND ANGULAR (10)

For Section A, you need to complete two questions; a Web API endpoint question and an Angular page question. Both questions do not require access to an existing database file to be completed. In other words, repositories are generated through the application. All the source files are in the following zipped files:

- **INF354SemTest_SectionA(1a).zip – SECTION A - QUESTION 1 (WEB API)**
- **INF354SemTest_SectionA(1b).zip – SECTION A - QUESTION 2 (ANGULAR)**

Once you have completed the 2 questions in this section, upload the **CourseController.cs** file and the **courses-list.component.ts** file to the Section A upload slot.

Your task is to complete the code base in order to get these applications to function as described below. The steps required to achieve this have been broken down into separate questions. For each question, please do the following:

- Read the instructions carefully.
- Look for comments (hints) in the provided sample solution. The comments will guide you in terms of where the code modifications are required.
- Apply the necessary code changes to the specified file in the required application.
- Only upload the modified file associated with the question to ClickUP.

(IMPORTANT: DO NOT UPLOAD ZIP, TAR, RAR or SLN files)

SECTION A - QUESTION 1 (WEB API) (5 MARKS)

Only upload the **CourseController.cs** file after completing this question.

- For this question, you must get the "GetCoursesByDurationType" function in the "CourseController.cs" file to work. The "GetCoursesByDurationType" function must display all the **Year** courses from a list of Year and Semester courses as a Web API endpoint.
- You need to call the "GetAllCoursesByDurationType" function from the CourseRepository inside the "GetCoursesByDurationType" function to retrieve the courses from the repository.
- Further, you must place the call in a try-catch block, and check for the 404 (Not Found) and 500 (Internal Server Error) status code errors.
- Thereafter, you should be able to call the updated function as a Web API endpoint from your browser.
- **NB: The application was created using Visual Studio 2019. The full list of additional configuration steps and instructions are in the "GetCoursesByDurationType" function in the "CourseController.cs" file, as comments. Please read through them.**

Q1 Expected Output

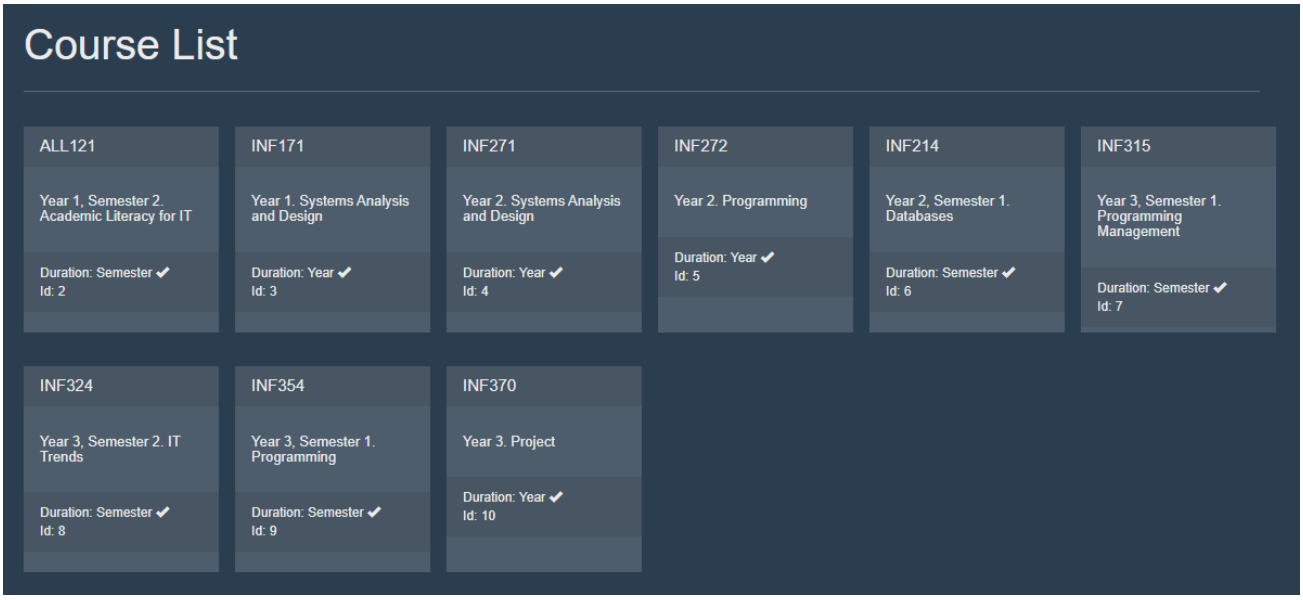
[{"courseId":3,"courseCode":"INF171","isYear":true,"duration":"Year","description":"Year 1. Systems Analysis and Design"},{"courseId":4,"courseCode":"INF271","isYear":true,"duration":"Year","description":"Year 2. Systems Analysis and Design"},{"courseId":5,"courseCode":"INF272","isYear":true,"duration":"Year","description":"Year 2. Programming"},{"courseId":10,"courseCode":"INF370","isYear":true,"duration":"Year","description":"Year 3. Project"}]

SECTION A - QUESTION 2 (ANGULAR) (5 MARKS)

Only upload the **courses-list.component.ts** file after completing this question.

- For this question, you need to allow **all the Courses** to be displayed as bootstrap panels in the "Home" link after looping through them. The data to loop through to display the Courses is in the "CoursesListComponent" class in the "courses-list.component.ts" file.
- The code for the panel/card is already created in the "course-template.component.ts" file. You will only be updating the "courses-list.component.ts" file. In other words, all the code for all the other functionality is already created.
- You just need to loop through all the courses and assign them to the "course-template". This will be done in the template section of the "courses-list.component.ts" file.
- **NB: The application was created using Visual Studio Code. To re-install the node_modules folder and its packages, you must run "npm install" from your command line. Thereafter, you can run the Angular application with the "npm start" command. The full list of additional configuration steps and instructions are in the "instructions.component.html", as comments. Please read through them.**

Q1 Expected Output



SECTION A TOTAL **10**

SECTION B – IONIC / ELECTRON (10)

For Section B, you need to complete three Ionic questions. All the source files are in the following zipped files:

- **INF354SemTest_SectionB(1).zip**

This section consists of one Ionic mobile application. This application works as a to do list, it reads its initial to-dos from a JSON file in its assets folder.


```
[
  {
    "id": 1,
    "name": "INF 354",
    "isOpen": false,
    "children": [
      {
        "id": 1,
        "name": "Do homework assignment 02",
        "isDone": false
      },
      {
        "id": 2,
        "name": "Do homework assignment 03",
        "isDone": false
      },
      {
        "id": 3,
        "name": "Do homework assignment 04",
        "isDone": false
      }
    ]
  },
  {
    "id": 2,
    "name": "INF 370",
    "isOpen": false,
    "children": [
      {
        "id": 1,
        "name": "Do deliverable assignment 00",
        "isDone": false
      },
      {
        "id": 2,
        "name": "Do deliverable assignment 01",
        "isDone": false
      },
      {
        "id": 3,
        "name": "Do deliverable assignment 02",
        "isDone": false
      }
    ]
  }
]
```


JSON structure for the to-dos to be listed by category.(assets/todos.json)

The to-dos are then displayed in an accordion list in the application. As seen in the JSON structure below, the to-dos should be listed in the application by category. The questions in this section will pertain to fixing / filling in the missing code to make sure that the application works as expected. The JSON structure can be found in the assets folder under the name todos.json

Please note: The adding or updating of either categories or to-dos should not write to the JSON file. Appending and editing should only affect the to-dos array in the tab1.page.ts. We only read from the JSON file initially and never use the JSON file again.

Your task is to complete the code base in order to get this application to function as described below. The steps required to achieve this have been broken down into separate questions. For each question, please do the following:

- Read the instructions carefully.
- Look for comments (hints) in the provided sample solution. The comments will guide you in terms of where the code modifications are required.
- Apply the necessary code changes to the specified file in the required application.
- Only upload the modified file associated with the question to ClickUP.

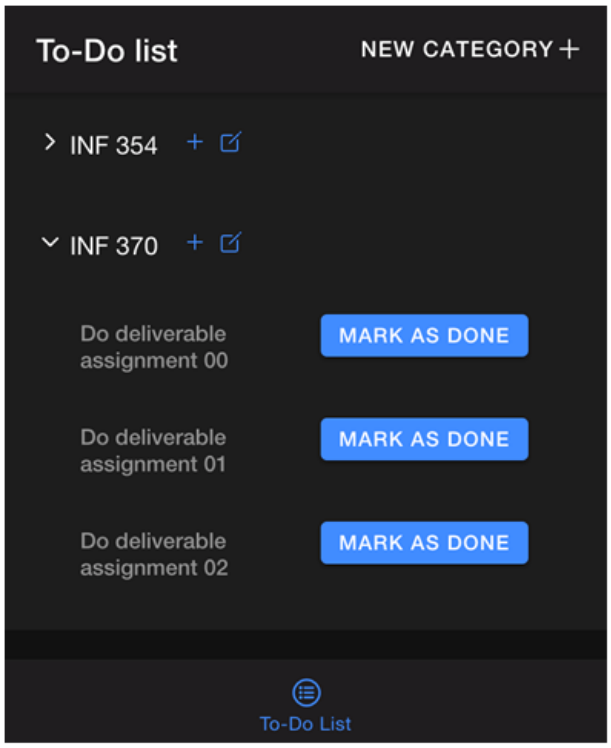
(IMPORTANT: DO NOT UPLOAD ZIP, TAR, RAR or SLN files)

SECTION B - QUESTION 1	**(3 MARKS)**

Only upload the **main.service.ts** file after completing this question.

- In this question, you will be required to write the code for a function to read the JSON data from the assets folder, **assert** that the data fits the structure of the Category[] interface that has been declared for the data coming in.
- The name of the function should be **getTodos()** .In order to read this data from the assets folder you will be **required** to use an HTTP request, please note that the HTTPClient class has already been injected for you.
- The function you write must have an explicit return type of Observable<Category[]>. If the function is completed correctly, the to-dos should be displayed in their categories as seen in the image below.

Q1 Expected Output

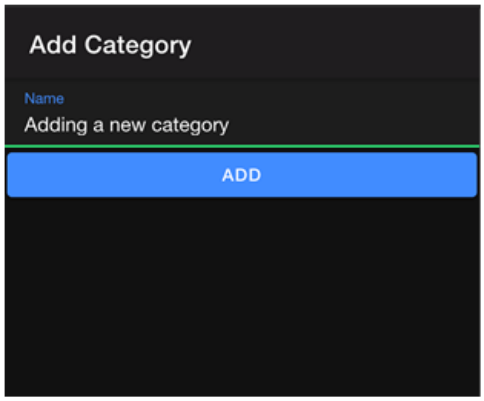


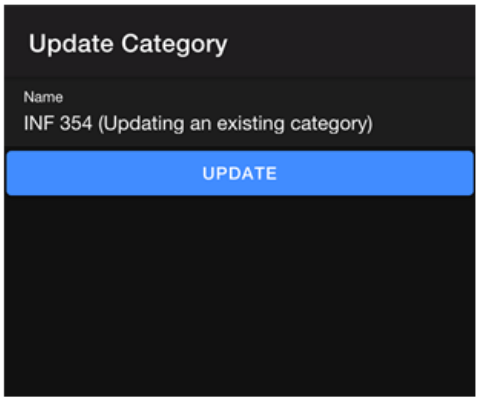
SECTION B - QUESTION 2 (3 MARKS)

Only upload the **add-edit-category.page.ts** file after completing this question.

- In this question, you will be required to configure the form group to be able to enter or edit the name of a Category. You will be required to complete the **ngOnInit** lifecycle hook to be able to configure this form correctly. This form should use the Reactive Forms approach to forms.
- This form group only has one form control being the **'name'** control, this control should be set as **required** for the form to be valid. Please note that this component should be able to cater for adding and editing a category if needed. The properties isAddMode , category (for the incoming data to the component), and form have already been **declared** for you.
- The function to submit the form and pass the name of the category back to the tab1.page.ts component has been written for you. The HTML form have been written for you as well. If you have completed this task correctly, the form should display with no issues when adding and updating a category.

Q2 Expected Output





SECTION B - QUESTION 3 (4 MARKS)

Only upload the **tab1.page.ts** file after completing this question.

In this question, you will be required to finish off three functions that perform similar actions. These functions are:

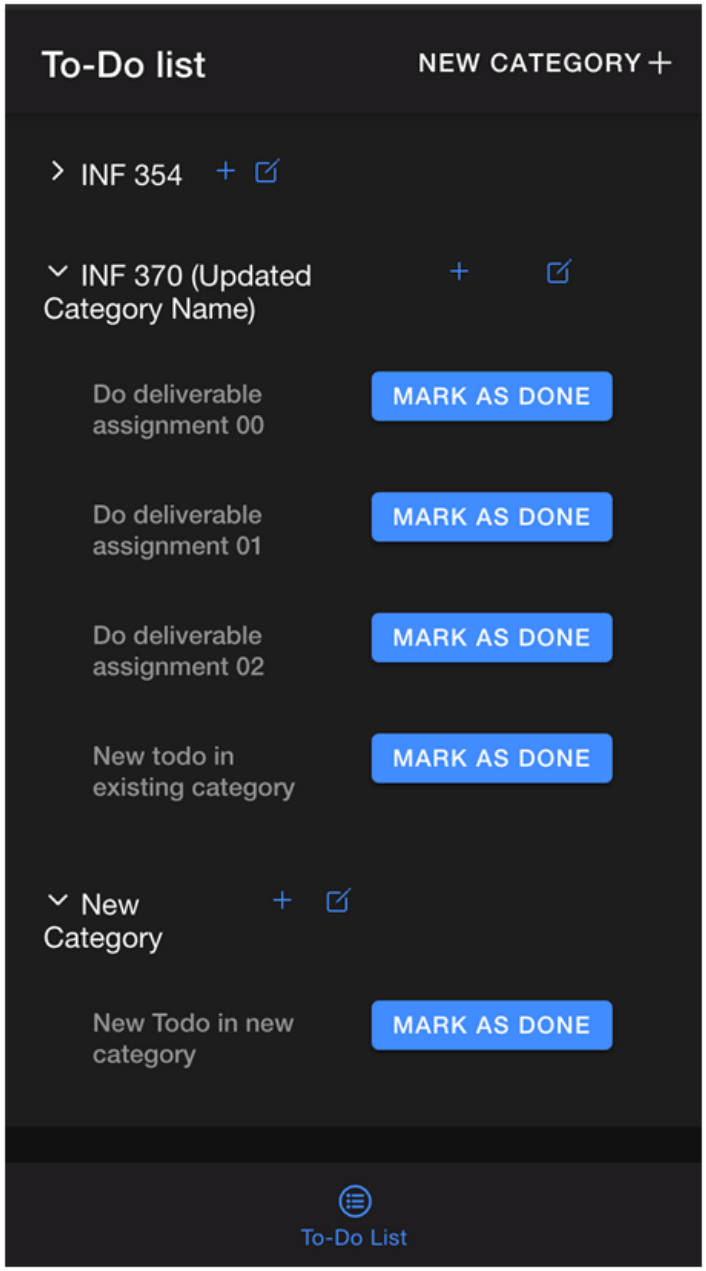
- The addCategory function
  - This function should initialise a **new, empty** and **closed** category with the **name** of the category being provided from the form in the add-edit-category.page.ts. You should obtain the name submitted from the modal and use it to append to the list todos in the tab1.page.ts.
  - The code to open the modal for adding a category has been provided to you. You will be required to write the code to obtain the name submitted from the add-edit-category.page.ts and initialise a new category and append it to the todos array.

- The updateCategory function
  - This function should update the **name** of the category. The updated name should be provided from the form in the add-edit-category.page.ts. You should obtain the name submitted from the modal and use it to update to the selected category to update in the todos list in the tab1.page.ts.
  - The code to open the modal for updating a category has been provided to you. You will be required to write the code to obtain the name submitted from the add-edit-category.page.ts and update the right category in the todos array.

- The addTodo function
  - This function should initialise a **new** to-do that will be appended to the list of children of the category that was selected. The **name** of the to-do being provided from the form in the add-edit-todo.page.ts. The new to-do should be initialised as **not** done. You should obtain the name submitted from the modal and use it to append to the correct category's children.
  - The code to open the modal for adding a todo has been provided to you. You will be required to write the code to obtain the name submitted from the add-edit-todo.page.ts and initialise a new todo and append it to the right category's children array.

- **Please note:** The adding or updating of either categories or to-dos should not write to the JSON file. Appending and editing should only affect the todos array in the tab1.page.ts. We only read from the JSON file initially and never use the JSON file again.
- If you have completed this question and the questions above properly, you should be able to fully append to and edit the to-do list. An example of what the functioning application would look like can be seen in the image below.

Q3 Expected Output



SECTION B TOTAL **10**

SECTION C – SASS & REPORTING (10)

For Section C, you need to complete two questions. All the source files are in the following zipped files:

- **INF354SemTest_SectionC(1).zip**

This application is a simple application that includes details on SASS / SCSS as well as ng2-charts and jsPDF. To ensure that this application runs and works you will be required to install the following:

- npm install
- npm install ng2-charts --save
- npm install chart.js --save
- npm install jspdf --save
- npm install html2canvas --save

Your task is to complete the code base in order to get this application to function as described below. The steps required to achieve this have been broken down into separate questions. For each question, please do the following:

- Read the instructions carefully.
- Look for comments (hints) in the provided sample solution. The comments will guide you in terms of where the code modifications are required.
- Apply the necessary code changes to the specified file in the required application.
- Only upload the modified file associated with the question to ClickUP.

(IMPORTANT: DO NOT UPLOAD ZIP, TAR, RAR or SLN files)

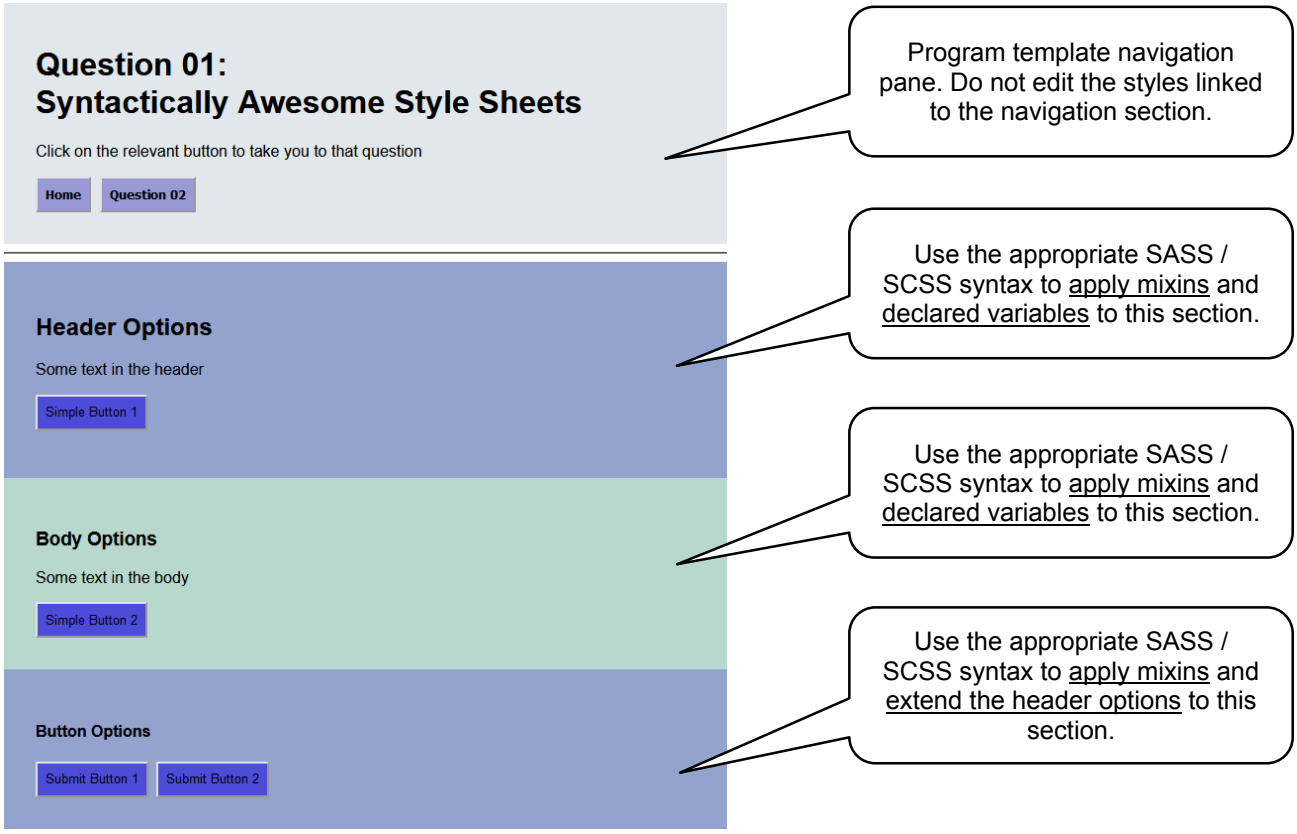
SECTION C - QUESTION 1 (3 MARKS)

Only upload the **question01.component.scss** file after completing this question.

Carefully consider the SASS variables and the SASS mixin's. Complete the SASS stylesheet. Only make use of SASS syntax. No traditional CSS stylesheet syntax will be considered.

- Replicate all the mixin's to the correct space in the stylesheet to ensure that the styles are applied as found in the example on the PDF in the question paper (2).
- Extend the header style to the submit section by making use of the appropriate style tag (1).

Q1 Expected Output



SECTION C - QUESTION 2 (7 MARKS)

Only upload the **question02.component.ts** file after completing this question.

Add the following data to a ng2-chart **bar chart**.

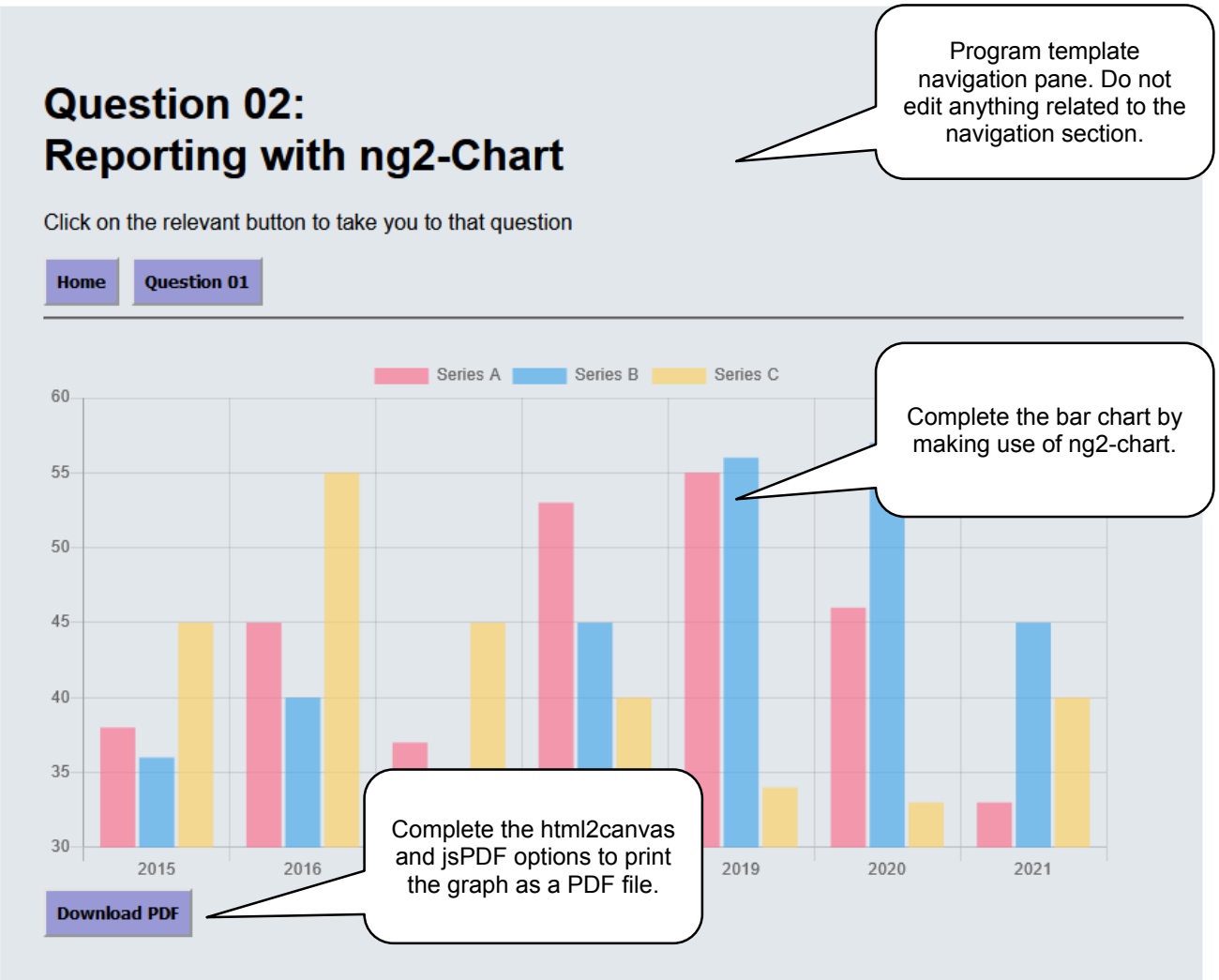
LABELS	2015	2016	2017	2018	2019	2020	2021
Series A	38	45	37	53	55	46	33
Series B	36	40	34	45	56	57	45
Series C	45	55	45	40	34	33	40

After the completion of the chart, make use of jsPDF to pint the bar chart to PDF.

To complete the aforementioned one would have to do the following:

- Modify and correct the '@angular/core' import (1/2)
- Add the appropriate imports to ensure that the chart functions (1/2)
- Add the appropriate chart options (1)
- Add the aforementioned chart data (1)
- Add the PDF options to get the appropriate elements (1)
- Add the appropriate html2canvas options (1)
- Add the final PDF page options (2)

Q2 Expected Output



SECTION C TOTAL **10**