Part 1 – Basic TypeScript App (12 marks)

The aim is to develop a simple one-page TypeScript app to maintain a small database of ‘movie’ information while the browser window is open. You can initialise your app with hard coded data, or you can start with an empty data structure. Types are important in your code and you are expected to declare types for example, for variables, function parameters, and function return values. Marks may be deducted for too few type definitions, i.e. writing JavaScript instead of Typescript.

The movie data has the following fields:

 Name

 Director

 Year (i.e., release year)

 Genre (Comedy, Horror, Action, Drama, Thriller, Romantic)

 Notes (one line of comments that user can enter)

The movie data has the following requirements (1 mark):

• Each record must have a value for all fields, only the Notes field can be blank. There must be

at least 10 records.

• The Genre must be one of the options shown above.

• A Name can only be entered and saved once (it must be unique for each movie).

Your database would probably be a Typescript array of JavaScript objects, each of which has properties corresponding to the above fields. You can use another data structure if you wish but types will be necessary to ensure integrity of the data (1 mark).

Your app should implement:

• A way to edit the movie data (2 marks). Remember you do not have to permanently save the data between browser sessions, but you do have to save it while the app is active. HTML form widgets are the easiest here.

• A way to add a new movie data (2 marks). Consider re-using the edit form above, it will require careful thought so as not to confuse the user.

• A way to delete individual movie with a warning message saying “Do you really want to…” to reduce accidental deletions (2 marks).

• A way to search any movie by the Name field (2 marks). This will probably done with a HTML list but you may use other techniques (e.g. implement a search text field).

Your app should also:

• Provide an interface equally viewable by narrow and wider mobile screens (it will be tested).

Note that this is just a request to keep your app simple and displayable on narrow screen as well as possibly wider screens. You can put links in a long page to navigate your app on narrow screens. This will be important when we get to mobile device apps later in the unit (1 mark).

• Use interactive features (e.g. use innerHTML assignments) to improve user experience (not alert() calls!). You are expected to add error messages where necessary (1 mark).

Part 2 – Expand the App (13 marks)

Modify the app from Part 1 to become a multipage Angular app. For example, place the add record on a separate page to the search facility. You can use as many pages as you like but don’t forget to use an error page (for malformed URLs), an Information page e.g., about the app and have your name as app developer. A Help page showing how to use the app would be nice as well.

Marks are allocated as follows:

• Having a working Angular app with appropriate Angular coding (3 marks). Minimally the app will have one component and one module, but to implement the following requirements you will need more components.

• Using Angular forms (3 marks). This is separate to the multipage facility. If you just add Angular code to the one-page Typescript app from Part 1, you have not used Angular forms.

• The multipage facility using the Angular router (3 marks). This includes the navigation between pages, error indicators and no broken links.

• Add at least one help button on each page that shows appropriate help messages. You should make interaction with the help facility as easy as possible. Also add helpful information, including error messages, when data entry is incorrect (3 marks).

• Use of CSS styles (1 mark). Just use them, there are no marks for artistic design. We are looking for the ability to use styles. Feel free to use styles you find in the Angular documentation