**Rich Web Lab 3**

**Question 1:** *Explain what is meant by the stream abstraction. What is the relationship between streams and the observer pattern?  What are streams useful for modelling and when might you use them in Rich Web development?*

Streams can be defined as a sequence of data made available over time. A stream can be thought of as a time-ordered list of data – which may be present now or may arrive some time in the future - which can be read and transformed into other streams or values in a specific order.

Streams implement the observer pattern – a design pattern in which a one-to-many dependency between objects is defined so that when one object changes state, its dependents are notified and updated accordingly. In RxJs, an Observable object emits a sequence of values occurring in the stream. This object can then be subscribed to, and do something with the data emitted.

Streams are useful for modelling asynchronous data sources, and processing data when you either don’t know its potential size or when it may arrive into an application. In terms of rich web development, streams can be used as powerful event processing mechanism. Using streams, asynchronous events such as mouse clicks, keyboard input, timers and DOM state changes can all be processed in the same logical structure.

**Question 2:** *Assume that you are building an interface to an API in your Rich Web App. Describe in detail how you could use the RxJS library to handle asynchronous network responses to API requests. In your opinion, what are the benefits to using a streams library for networking over, say, promises? And what do you think are the downsides?*

To handle asynchronous network responses to API requests using RxJs, I would first create an observable to hand asynchronous user input events, such a mouse click on a search button. I would then subscribe to this observable, use it to handle any user input needed for the request, and then send a http request to the API. The subscribe operator can be used to handle any possible errors that may be returned from the http request, and similarly to display data successfully returned from the API.

In my opinion, the advantage of using streams over promises for networking is that the streams library allows the networking of your application to used in tandem with asynchronous events such as user input.

One disadvantage of using streams that comes to mind is that all logic to do with handling requests and responses must be encapsulated within an observable object.