Section 1: Theory Questions

1.1 Explain the difference between a primary key and a foreign key

A primary key is a unique identifier in its table whereas the foreign key is a reference to a primary key in another table, establishing relationships between tables.

Core differences:

- Each table can only have one primary key but a table can have multiple foreign keys
- Primary keys cannot be null but foreign keys can be since not every record will link to another table
- Primary keys are a unique identifier for records in that table but foreign keys establish relationships between tables

1.2 Explain the difference between undefined and null in JavaScript

Null is used when a variable has no value, it indicates the absence of a value and is typically intentionally used by the user. It's considered an object type. On the other hand, undefined happens by default when you create a variable but you haven't assigned a value to it. Undefined is then assigned to that variable and is its own type.

When something is *null*:

```
let user = null;
console.log(typeof user);     //would output object
```

When something is *undefined*:

```
let user;                     // no value assigned
console.log(typeof user);     //would output undefined
```

When the user doesn't assign a value, we say the variable is not *initialised*.

Core differences:

- Null is intentionally used by the user whereas undefined occurs by default
- Null represents the absence of a value whereas undefined occurs when a variable has not been initialised
- Null is an object type whereas undefined is an undefined object type

1.3 Explain the difference between a commit and a push in Git

When a user does *git commit* it saves staged changes that weren't tracked to the local git repository. To do so, you must stage your changes using *git add [file] or [. for all untracked files]*.

Whereas a *git push* sends your local commits to a remote repository, synchronising your local repository with your remote one. As such, changes made in your local repository can now be seen in your remote repository!

Core differences:

- Commit saves changes to your local repository whereas push sends committed changes to your remote repository
- Commit only affects the local repository whereas push involves both the local and remote repository
- Commit is a local command and things remain private after your commit whereas changes can be made available to others after a push, provided they have access to your remote repository
- You typically commit before making a push

It should be noted that you can do 'remote' commits via interfaces like GitHub where you can access remote repositories. Similarly, these commits only apply to the remote repository, the local repository is unaffected. To update your local repository with these changes, you have to do a *git pull* which involves both the local and remote repositories.

1.4 Provide examples of two web APIs and describe their functionalities

A web API is an application programming interface (API) for the web. It acts as a bridge between a client and a server. It enables communication and the exchange of data over the web, using HTTP methods to access or send data.

Two examples of a web API are:

1) Amazon Product Advertising (APA) API

   Purpose: the APA API allows access to data such as items for sale its associated information such as pricing, reviews, and images. It's used in web applications or

websites that show Amazon products, enabling users to search, browse and retrieve product data. It's useful for comparison websites or affiliate marketers.

Features Included:

- Searching products available on Amazon and its associated information like price, availability, and reviews
- Retrieving product information in your language of preference
- Retrieve data about ongoing deals and promotions that are currently available on Amazon
- Retrieve data regarding which products are on Prime
- Fetch real-time prime pricing for events like Prime Day.

2) Facebook Graph API

Purpose: Allows access to Facebook's social graph data, such as posts, comments, events, and pages. It was named as such because it represents the concept of graphs as data structures where nodes would be Facebook pages and edges are connections between them. You can retrieve all data regarding users, messages, and pages. It would be useful for data analytics or allowing a third-party app to log into Facebook.

Features Included:

- Retrieve information about users such as profile details, friend lists, photos, posts etc
- Retrieve data regarding connections and relationships between users, groups and pages
- Read or publish content on behalf of users
- Retrieve data for analytical purposes for instances like marketing campaigns
- Retrieve data regarding events, with further functionality to manage these events
- Send and receive messages using Facebook messenger

1.5 Describe four tasks in the role of the Product Owner in Agile development

The product owner manages and plans the project. This includes ensuring the team is working to their best ability and managing product backlog.

Four key tasks the role of Product Owner is responsible for in an agile work environment:

1) Sprint Planning - Given the Product Owner has a high-level overview of the vision, strategy and priorities behind the product, they are a key part of sprint planning. It's their responsibility to propose improvements and goals based on stakeholder feedback. In this way, they work with developers to define sprint goals, determine what backlog items to include in the sprint and decide how the team will achieve this work.

2) Sprint Review - Another task the Product Owner carries out is demonstrating the work that has been done in a sprint and obtaining feedback from stakeholders. In this way, it is the job of the Product Owner to liaise with stakeholders and act as a primary communicator between stakeholders and teams. They should be people who can understand and anticipate client's needs.

3) Sprint Retrospective - reflecting on the sprint and identifying points of improvement in terms of the approach to the sprint and process is another responsibility of the Product Owner. They participate in discussions on how to improve the process and offer feedback as to how the team can better address business needs.

4) Backlog Refinement - the Product Owner manages and maintains the product backlog. As such, it is their task to continuously refine and reprioritise the backlog per changing needs. The product backlog is a dynamic list of all features, enhancements and fixes for a product that is ordered according to priority. The product backlog is a live document that is continuously updated as the product evolves. Because of this, the Product Owner is responsible for making the list accessible and available to all stakeholders such as scrum team members. This enables future work to be aligned with business priorities.

1.6 Name two types of SQL joins and provide an example scenario for each

There are multiple types of joins such as INNER, LEFT OUTER, RIGHT OUTER, FULL. We'll look at INNER and LEFT OUTER in more detail!

Inner joins return rows that have matching values in both tables. If a row in table 1 does not have a matching value in table 2 then that row is excluded.

For example, say we have two tables in a database called *Hogwarts*: *Students* and *Classes*:

Students

StudentID	StudentName
1	Harry Potter
2	Hermione Granger
3	Ron Weasley
4	Draco Malfoy

Classes

ClassID	StudentID	ClassName
101	1	Defence Against the Dark Arts
102	2	Potions
103	3	Transfiguration

Say we want to find the students who are actually attending classes/ are enrolled, we can do an INNER join on the tables in a query like:

```
SELECT Students.StudentName, Classes.ClassName
FROM Students
INNER JOIN Classes
ON Students.StudentID = Classes.StudentID;
```

This would return something like:

StudentName	ClassName
Harry Potter	Defence Against the Dark Arts
Hermione Granger	Potions

Ron Weasley	Transfiguration

Only rows referring to students present in both tables are returned. As such, Draco Malfoy's record is excluded.

A LEFT OUTER JOIN returns all rows from the 'left; table, the first table in the query, and the matched rows from the 'right' table (the second table in the query). If there is no match for one of the left table rows, null values are returned for selected columns from the right table.

For example, say we have an additional table in the Hogwarts database that details information regarding students and quidditch:

Quidditch

TeamID	StudentID	TeamName
101	1	Gryffindor
101	3	Gryffindor
102	4	Slytherin

Say we want a high-level overview of students, including their extracurriculars, maybe we want to know who to target to get into an extracurricular club. We could do a query like:

```
SELECT Students.StudentName, Quidditch.TeamName
FROM Students
LEFT JOIN Quidditch
ON Students.StudentID = Quidditch.StudentID;
```

This would return something like:

StudentName	TeamName
Harry Potter	Gryffindor
Hermione Granger	NULL
Ron Weasley	Gryffindor

Draco Malfoy	Slytherin

We now know Hermione Granger does not participate in Quidditch and so has no TeamName. Students is the ‘left’ table since it is the first table mentioned and Quidditch is the second table as it is the second table mentioned.

1.7 What is the difference between synchronous and asynchronous functions?

Asynchronous functions allow tasks to be executed in the background. This avoids blocking the main thread of execution so other code can run while the task is completed. This is useful for tasks that take time to complete, like when you’re fetching/reading files in an API!

Synchronous functions are tasks that are sequentially executed one after another. Typically this is implemented when the execution of the following code depends on completing the task before it. They are blocking unlike asynchronous functions because the rest of the program execution stops until the task is completed.

1.8 Explain both Agile and Waterfall approaches to Software Development and at least 2 differences between them.

Agile methodology is a flexible form of project management. It was created by a group of software developers in 2001 and enables continuous development through an iterative and cyclic approach. Each cycle of development walks through requirements, design, development, testing, deployment, and review.

It involves working in short sprints, covering all of the above development stages each time, to deliver a project in small increments. Therefore, it inherently promotes continuous testing and frequent releases. There are other frameworks built on top of the methodology like scrum and kanban.

Whereas the waterfall approach is a linear and traditionally sequential approach to software development. Once a phase is finished, it is typically not revisited and the project continues onward (like a waterfall!).

Although there is some overlap with the phases there are some that stand out. The phases that are included in the waterfall methodology are requirements, analysis, design, coding, testing and operations. The phases are such that they don’t need to be revisited and are completed in their entirety before moving on to the next phase. Thus testing and feedback only occur once so frequent releases and continuous testing aren’t as big of a thing.

Both approaches have their use cases and advantages but the core differences are:

- Flexibility and room for change: Agile is a highly flexible methodology which allows for changes or adjustments at any stage. Whereas, waterfall methodology is a rigid and structured approach with less of a scope for changes. You might employ an agile methodology where there is a lot of customer involvement and you get their input regularly. Because of this flexibility, this approach can manage risks better and identify problems early.

- Overall development process: Agile methodology is iterative and incremental, delivering bits of work in rapid intervals (sprints). After each sprint, it's reviewed and feedback is incorporated. Waterfall methodology is linear and sequential, testing and feedback only occur after all the development is complete. Because of this, waterfall approaches rely on extensive upfront planning where all the requirements are gathered beforehand.

- Testing: Testing and feedback occur only once in waterfall approaches as opposed to continuously in agile approaches.

- Responsibility: In agile methodologies, the entire team usually takes ownership of the project whereas in waterfall methodologies, sometimes phases have different teams.