Week 1 (w/c 10. June) -

* Introduction to Digital, getting setup on systems and tools.
* Research into Grafana, Kibana, Kubernetes, Elastic Stack, PromQL, Microservices and SP Digital
* Learning and practicing PromQL through Grafana to create queries.
* Assigned tickets to create an SAP FM Overview dashboard – this was an introduction to creating Grafana dashboards, deciding the correct data to show and the best method to show it. (Tables, single statistics, line graphs. Also doing calculations with data to find averages, max, min and other aggregate PromQL queries
* Introduction to DevOps team – investigated info about DevOps and further into each department in SP Digital what the role is, what they do:
  + DevOps are responsible for helping to integrate, maintain and troubleshoot and update software.
* Onboarding introduction to Services– info about the SP app, the team and how the Services team was structured and what you might do whilst on the team e.g., processes.

Week 2 (w/c 17. June) –

* I managed to create a first draft of the SAP FM Overview dashboard to, and then went about testing data and making changes to the dashboard so that only necessary info and readability were good for any users looking for an easy-to-read summary of the data.
* I sat in on a call regarding a ticket about User Management errors. Specifically, an issue where a customer was receiving password reset emails in their deceased husbands/wife name even after the details had been updated previously in SAP to title and first name, last name.
* I attempted to reverse engineer that ticket to see if I could see how my team had solved the problem from back to front, it turned into to be a duplicate data issue as any email addresses entered with different caps would create a new entry e.g. [abc@abc.com](mailto:abc@abc.com) / [ABC@abc.com](mailto:ABC@abc.com) the fix was to use lower() (or something similar to correct caps) to make sure everything was the same capitalization before checking for duplicate data.
* I investigated testing and debugging and did several Cucumber and JUNIT Testing example exercises and learned how to setup the dev environment for maven so that I could do Junit and Cucumber testing – this give me an insight as SP uses cucumber testing for everything with all test scenarios (Cucumber stories) accounted for.
* I also completed the Learn Java: Debugging course on codecademy this gave me an additional insight into different debugging methodologies and how I could implement them in future projects.
* I got assigned another ticket to complete the Weekly Reporting API/FM Dashboard – this was for different calls on API’s and FMs for various features showing different data such as total calls, average response time, successful calls, and various other more specific statistics on each call so that it was easy to tell at which point things were going well and what might need looked at.
* Another dashboard I was given to create was the API/FM Login Performance Dashboard, similar to the weekly reporting dashboard this one broke down all the calls and average times it took for each part in the Login Journey – the Overall Performance data for every call and then split into the 2 types of customers Smart Customers and Credit Customers – it showed the individual total times for API and FM and then the overall time for the individual customer.

Week 3 (w/c 24. June) –

* I was assigned another ticket for Ops Performance and Stability Improvements to create a dashboard on the API/FM calls for App/Web Features

Week 4 (w/c 15. July | Started Wednesday after holiday) –

* Completed the Codecademy Skill Path for Creating REST APIs with Spring and Java, the final part was an introduction to Sprint Context – learning about and using spring boots and beans. Spring Beans referring to an object that is managed by the Spring IoC Container and Spring Boots referring to the simplification of development by using Spring Boot to extend the Spring framework by enabling the autoconfiguration of Spring beans.
* The next section of this was a topic on Data with JPA (Java Persistence Application) this involved creating CRUD applications (Create, Read, Update, Delete) this used a lot of my previously gained knowledge such as REST and allowed me to gain knowledge on JPA which has many benefits such as improved performance, Simplified Data access and Portability.

Week 5 (w/c 22. July) –