

A dark blue vertical bar runs down the left side of the page. A blue arrow points to the right from this bar, containing the year '2021'. In the bottom left corner, there are several thin, curved lines in dark blue and light grey.

2021

# Project 1 scope

CMPG323 Discovery Project

## CONTENTS

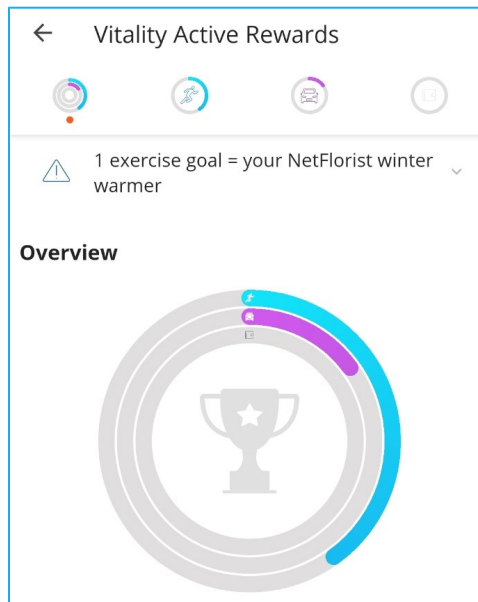
Background.....	2
Project background .....	2
Project: Discovery Account system .....	6
Technology stack .....	6
GIT .....	6
Java .....	6
IDE: IntelliJ IDEA .....	6
Build tool .....	7
Spring framework.....	7
Swagger .....	7
Docker (Optional).....	7
Diagrams.....	8
Logging.....	8
Code coverage.....	8
Prioritization of tasks .....	8

## BACKGROUND

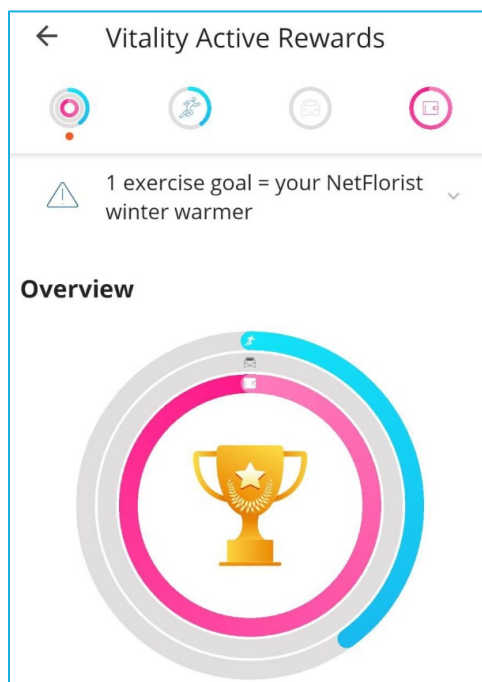
### PROJECT BACKGROUND

Discovery rewards its members for making healthy choices and living a healthy lifestyle through Vitality. Vitality's Active Rewards programme plays a key part in this.

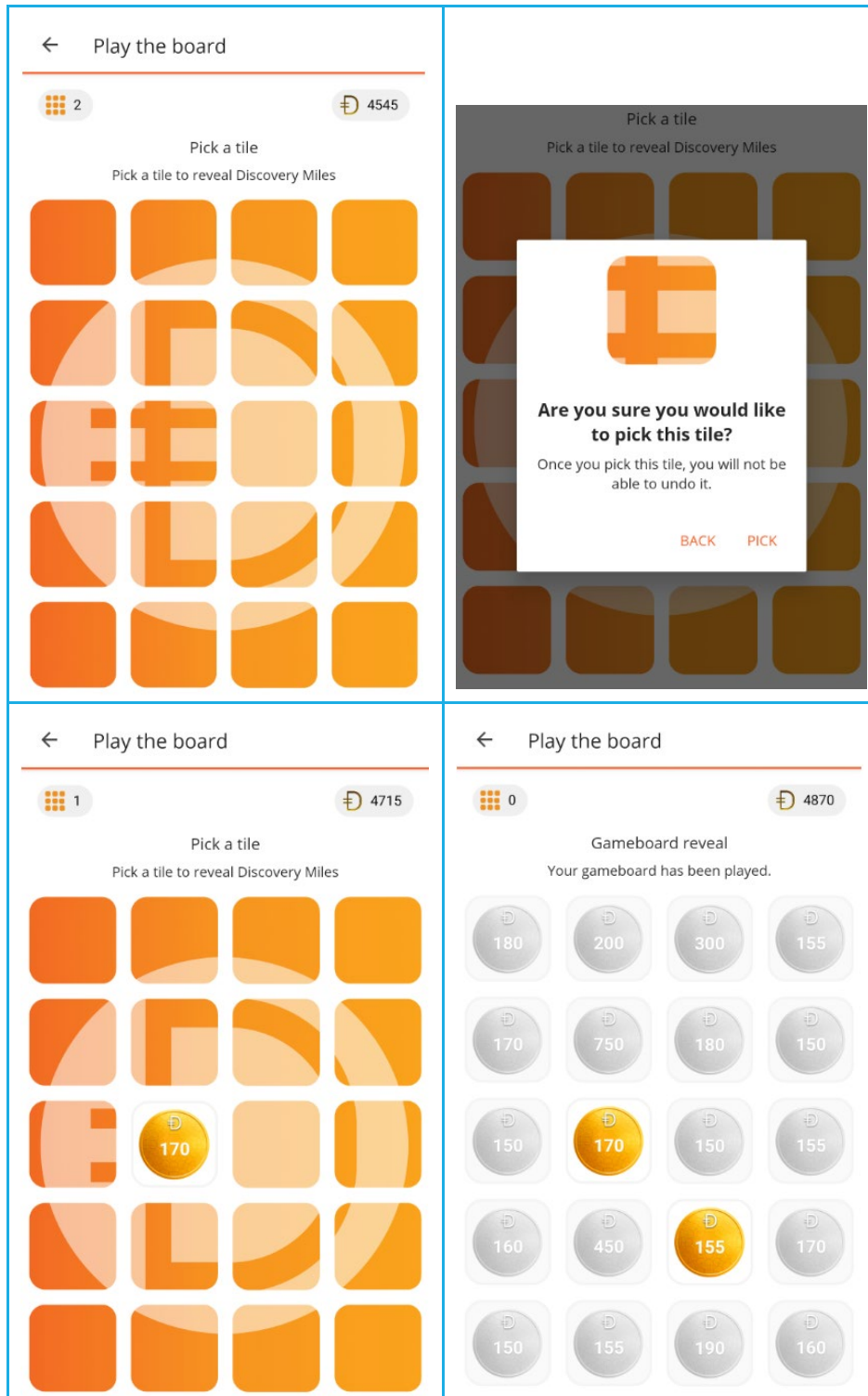
Active Rewards looks at members' Health and Fitness, Driving and Spending behavior to track towards each members' weekly goals.



Members who complete their weekly goals are awarded plays on the weekly gameboard.

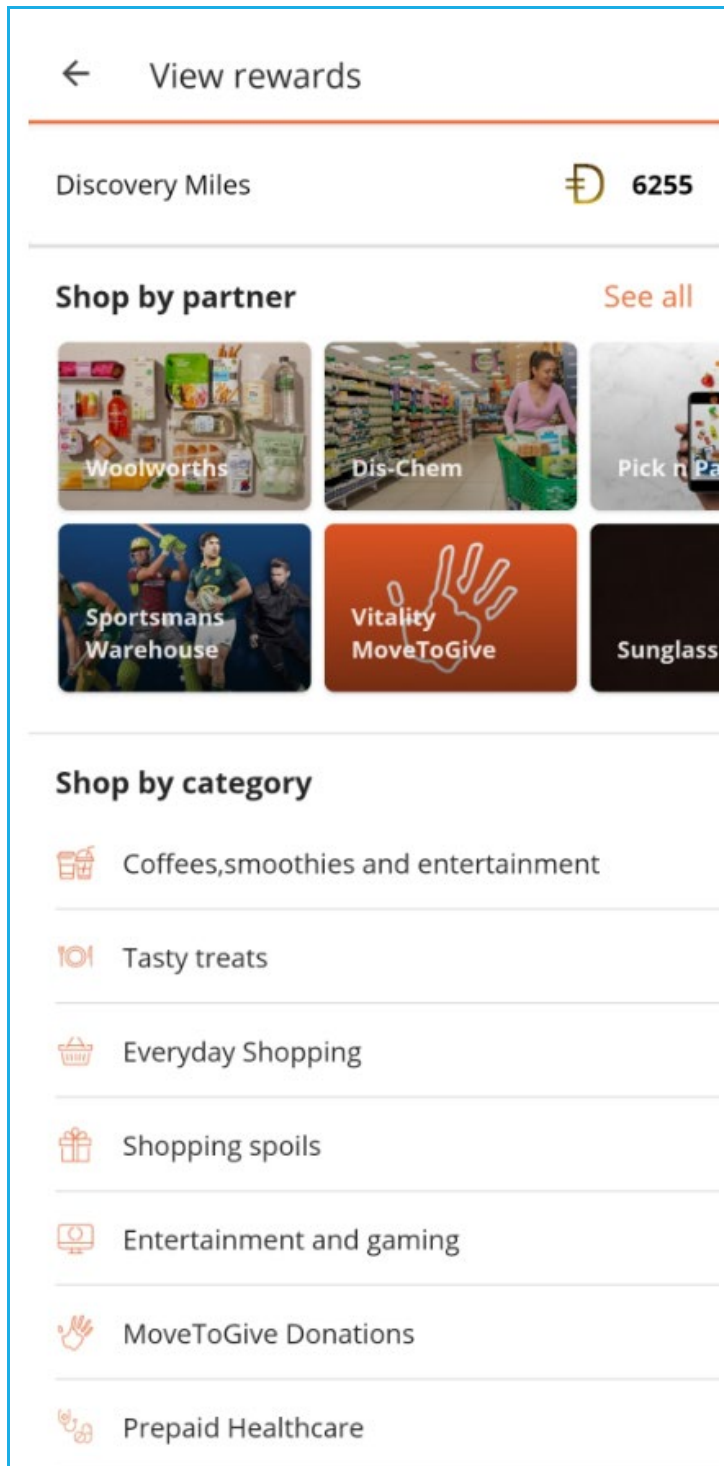


Every week there is a new gameboard and members use their plays earned in the previous week to play. The gameboard is filled with hidden tiles which contain Discovery Miles. The member uses the plays to pick tiles and earn the miles on the tiles. Once all the plays are used up, the entire game board is revealed.



Members accumulate Miles earned on the gameboard and can exchange their Miles for a reward voucher.

Active Rewards has many rewards partners and offer many different reward categories:



Each category lists many rewards that the member can choose when they have enough Miles:

### Entertainment and gaming

Choose a reward partner

**DStv BoxOffice**  
1 Movie Credit within 72hrs  
(Ensure you have a decoder that is compatible with DStv BoxOffice. In addition your ID/passport with Vitality must match your DStv account details.)  
R 315 (R31,50)  
~~R 350 (R35,00)~~

**Sony PlayStation**  
R50 credit to spend on PlayStation bundles  
R 450 (R45,00)  
~~R 500 (R50,00)~~  
Limited to 1 per member per day.

**Spotify**  
1-month subscription to Spotify  
R 540 (R54,00)  
~~R 600 (R60,00)~~

**Steam**  
R80 credit to spend on Steam  
R 720 (R72,00)  
~~R 800 (R80,00)~~  
Limited to 1 per member per day.

**Sony PlayStation**  
R100 credit to spend on PlayStation bundles  
R 900 (R90,00)  
~~R 1000 (R100,00)~~  
Limited to 1 per member per day.

**BT Games**  
R100 to spend online and in-store at BT Games  
R 900 (R90,00)  
~~R 1000 (R100,00)~~  
Limited to 1 per member per day.

**LEGO®**  
Spend R100 online at LEGO® Certified Stores  
R 900 (R90,00)  
~~R 1000 (R100,00)~~

**Spotify**  
3-month subscription to Spotify  
R 1620 (R162,00)  
~~R 1800 (R180,00)~~

**BT Games**  
R200 to spend online and in-store at BT Games  
R 1800 (R180,00)  
~~R 2000 (R200,00)~~  
Limited to 1 per member per day.

**LEGO®**  
Spend R250 online at LEGO® Certified Stores  
R 2250 (R225,00)  
~~R 2500 (R250,00)~~

### Shopping spoils

Choose a reward partner

**Cape Union Mart**  
R100 to spend in-store  
R 900 (R90,00)  
~~R 1000 (R100,00)~~

**Exclusive Books**  
R250 to spend at Exclusive Books in-store and online  
R 2250 (R225,00)  
~~R 2500 (R250,00)~~  
Limited to 1 per member per day.

**Poetry**  
R250 to spend in-store  
R 2250 (R225,00)  
~~R 2500 (R250,00)~~

**Cape Union Mart**  
R250 to spend in-store  
R 2250 (R225,00)  
~~R 2500 (R250,00)~~

**Keedo**  
R250 to spend in-store  
R 2250 (R225,00)  
~~R 2500 (R250,00)~~

**Loot**  
R250 to spend on www.loot.co.za  
R 2250 (R225,00)  
~~R 2500 (R250,00)~~  
Limited to 1 per member per day.

**Zando**  
R250 to spend on www.zando.co.za  
R 2250 (R225,00)  
~~R 2500 (R250,00)~~  
Limited to 1 per member per day.

**Strava**  
12 month Strava membership  
R 5399 (R539,90)  
~~R 5999 (R599,90)~~

**Sunglass Hut**  
R2000 to spend in-store  
R 18000 (R1 800,00)  
~~R 20000 (R2 000,00)~~  
Limited to 1 per member per day.

## PROJECT: DISCOVERY ACCOUNT SYSTEM

You are a backend developer for Discovery Vitality and need to write the Account system that manages the Active Rewards currency, Discovery Miles.

Here are your three user story requirements:

**As a Member**

**I want** to add Miles to my Miles account

**So that** I can earn and accumulate Miles when I reveal tiles on the gameboard.

**As a Member**

**I want** to view Miles in my Miles account

**So that** I can know whether I accumulated enough Miles for the reward I want.

**As a Member**

**I want** to subtract Miles to my Miles account

**So that** I can exchange my Miles for a reward voucher.

Use the Technology stack described to satisfy your three user story requirements. Delivering a project is about more than just writing the code. You need to do analysis up front and document your system. Your application needs to run within an integrated environment. For testing purposes add an optional field on the add service that will cause the service to throw an exception when indicated.

## TECHNOLOGY STACK

You will be expected to make use of the Technology stack listed below.

Discovery Vitality will provide online training classes on these technologies during the semester.

---

### GIT

GIT is used for version control.

You can download it here: <https://git-scm.com/downloads>

You can read more about it here: <https://git-scm.com/doc>

---

### JAVA

You will need to use Java 8 or higher.


You can download it here: <https://www.oracle.com/za/java/technologies/javase/javase-jdk8-downloads.html>

---


### IDE: INTELLIJ IDEA

You can download it here: <https://www.jetbrains.com/idea/download/>

Download the free Community version on the right:

 **IntelliJ IDEA**

Coming in 2021.2   What's New   Features   Resources



Version: 2021.1.2  
Build: 211.7442.40  
1 June 2021  
[Release notes](#)

## Download IntelliJ IDEA

[Windows](#)   [macOS](#)   [Linux](#)

### Ultimate

For web and enterprise development

[Download](#)   [.exe](#) ▼

Free 30-day trial

### Community

For JVM and Android development

[Download](#)   [.exe](#) ▼

Free, built on open source

---

## BUILD TOOL

You may choose whether you want to use either Maven or Gradle as a build tool. Only Maven will be covered in the training.

---

### MAVEN

You can download and read more about it here: <https://maven.apache.org/>

---

### GRADLE

You can download and read more about it here: <https://gradle.org/>

---

## SPRING FRAMEWORK

Spring is a widely used framework that does a lot of boiler plate code for you in the background. It provides a comprehensive programming and configuration model for modern Java-based enterprise applications. You do not need to install Spring as it will be pulled in as a dependency via your build tool.

You can read more about it here: <https://spring.io/>

---

## SWAGGER

Swagger is a quick way to generate web service documentation and provide you with an easy to use interface to call your services.

You can read more about it here: <https://swagger.io/>

---

## DOCKER (OPTIONAL)

You will need to run your application in some java container. Docker provides you with an environment to run your container in.

You can download it here: <https://docs.docker.com/get-docker/>



You can read more about it here: <https://docs.docker.com/>

---

## DIAGRAMS

You will be required to draw diagrams (ERD, Flow and Use Case diagrams) for your projects. You can make use of any tool of your own preference to do so.

If you are not sure what to use, you can make use of draw.io online via <https://app.diagrams.net/> for free.

Or install the free draw.io desktop app that you can download from: <https://github.com/jgraph/drawio-desktop/releases>

---

## LOGGING

Your application will need logging. Make use of a logging framework.

If you are not sure what to use, you can use Logback - <http://logback.qos.ch/>.

---

## CODE COVERAGE

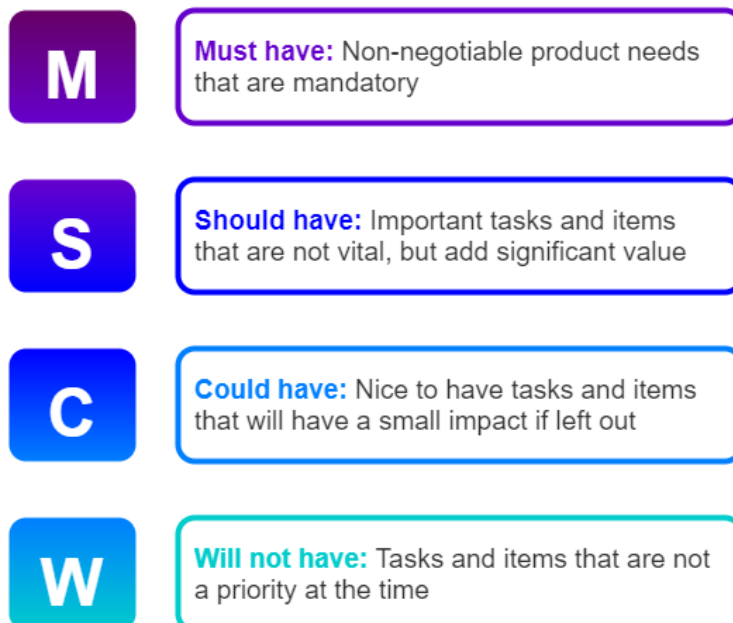
Use a tool, library, plugin or framework to check your code coverage.

If you are not sure what to use, you can use JaCoCo - <https://www.eclemma.org/jacoco/>.

## PRIORITIZATION OF TASKS

MoSCoW Prioritization will be used for your projects and you will be graded against this for the tasks and items that you included and have in a working state in your project.

### MoSCoW Prioritization



Your different tasks and deliverable items have been Prioritized using MoSCoW:

MoSCoW	Items and Tasks
Must have	<ul style="list-style-type: none"><li>ERD diagram. (Your table design.)</li><li>Use case diagram.</li><li>Flow diagrams for each service.</li><li>Use a build tool to manage all your dependencies and build your app.</li><li>Your application can connect to your own Data Base.</li><li>You have created your tables on the Data Base.</li><li>You have some unit tests.</li><li>You have a service that can add Miles for a member.</li><li>You have a service that can view Miles for a member.</li><li>You have a service that can subtract Miles for a member.</li><li>You can run your application standalone (via Spring boot).</li><li>You are using source control.</li></ul>
Should have	<ul style="list-style-type: none"><li>You can specify the start date on the service that adds Miles for a member.</li><li>You have proper error handling.</li><li>You have made use of proper application layering.</li><li>You have a Swagger API for your services.</li><li>You have a code coverage check.</li><li>You have an optional field on your add service that will cause the service to throw an exception.</li><li>Your transaction rolls back in the case that an exception occurs.</li><li>You have some logging.</li></ul>
Could have	<ul style="list-style-type: none"><li>You can run your application in Docker.</li><li>Your application can handle more than one Currency (i.e. not just Miles)</li><li>You can configure your different currencies via a service.</li><li>You have at least 80% unit tests code coverage. (excluding POJO's)</li></ul>
Will not have	<ul style="list-style-type: none"><li>A user frontend (Mobile, Webpage or Desktop application) that calls your services.</li></ul>