2021

Project 1 scope

CMPG323 Discovery Project

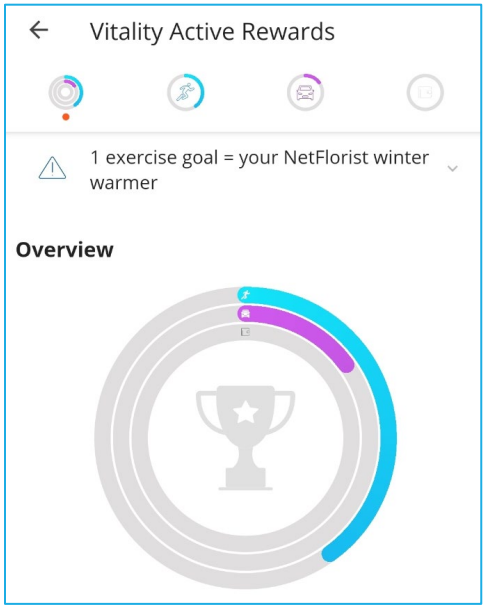
CONTENTS

BACKGROUND

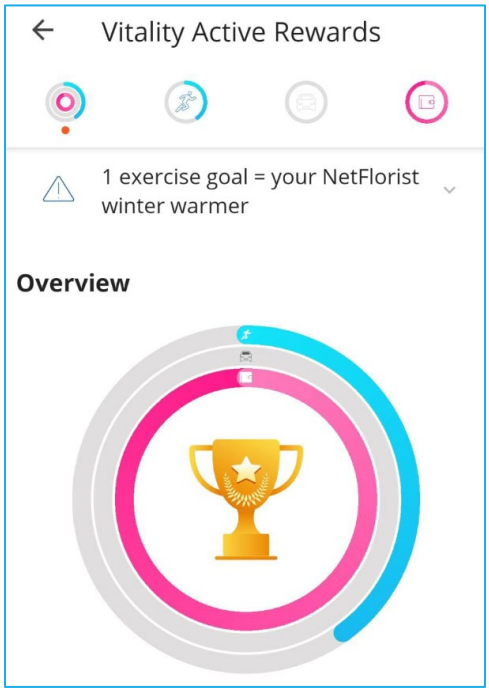
PROJECT BACKGROUND

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

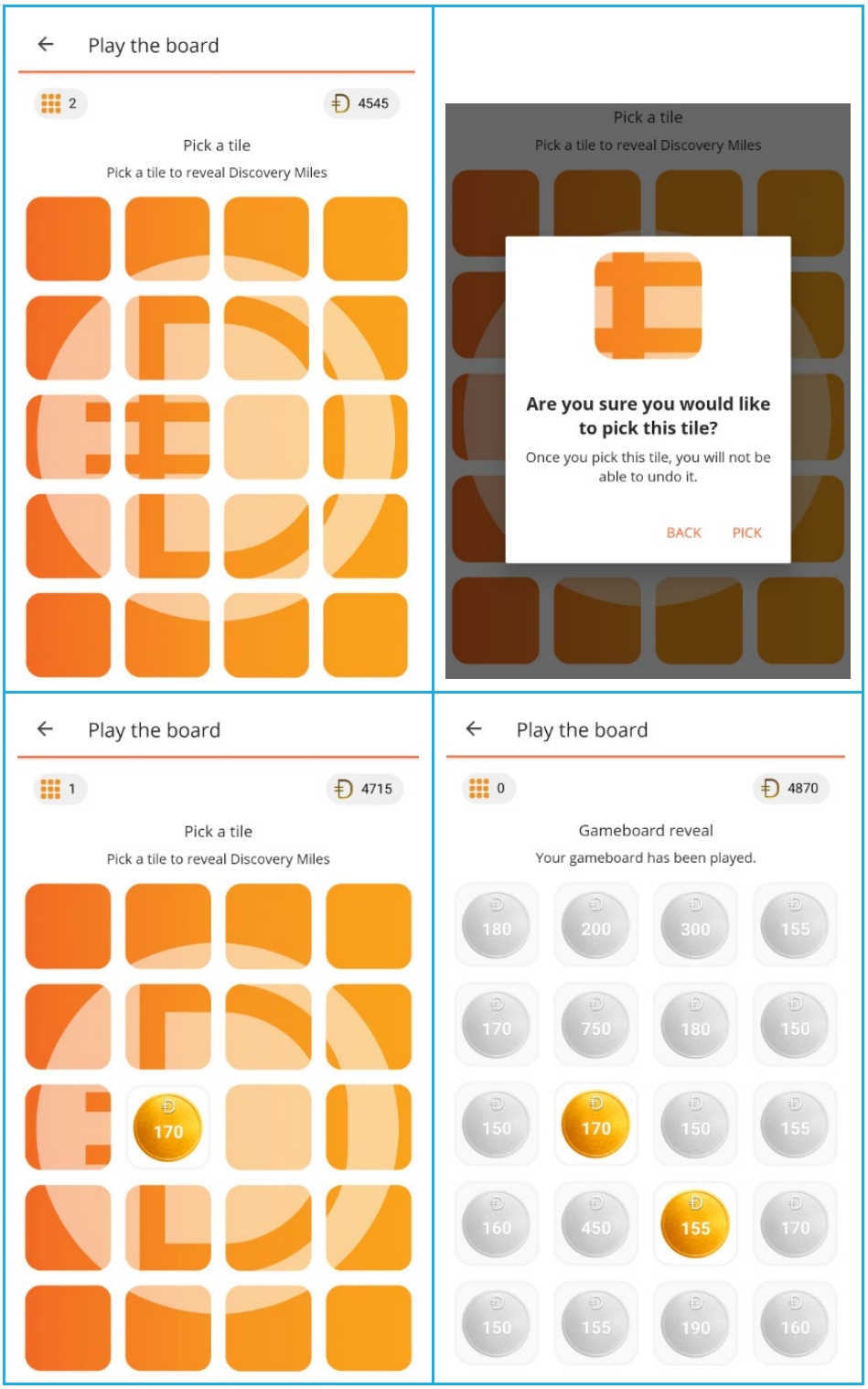
Active Rewards looks at members' Health and Fitness, Driving and Spending behavior to track towards each members' weekly goals.



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

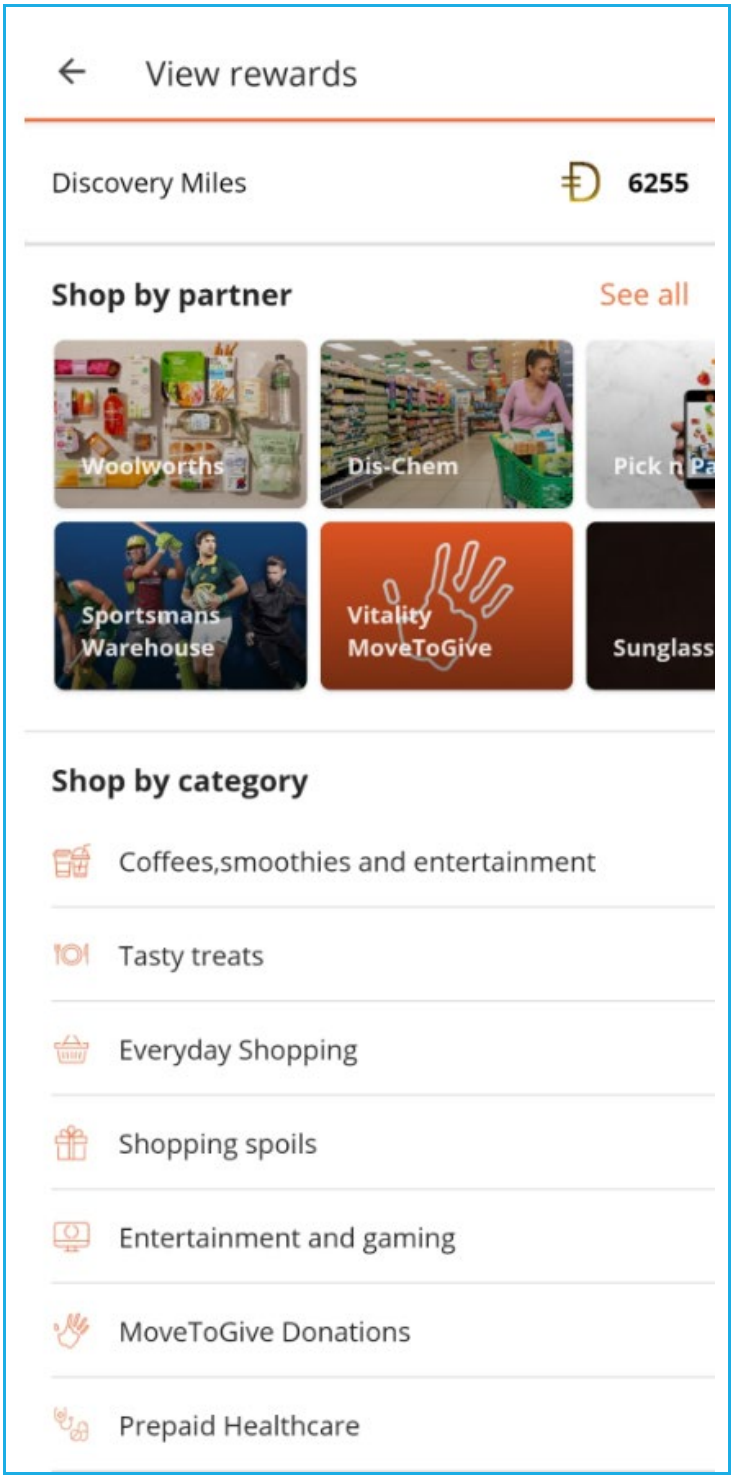


Every week there is a new gameboard and members use their plays earned in the previous week to play. The gameboard is filled with hidden tiles which contain Discovery Miles. The member uses the plays to pick tiles and earn the miles on the tiles. Once all the plays are used up, the entire game board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

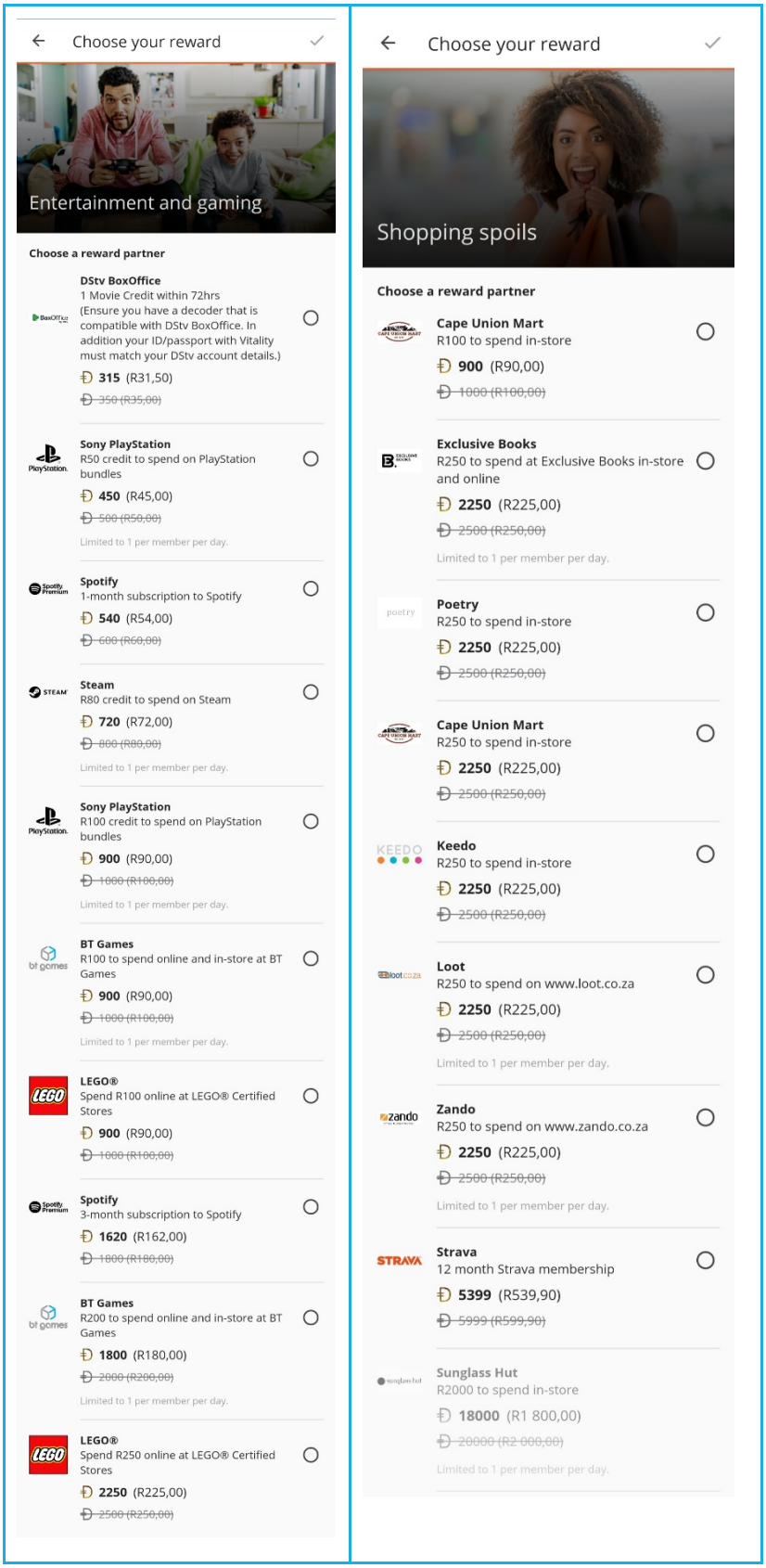


Members accumulate Miles earned on the gameboard and can exchange their Miles for a reward voucher.

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:



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

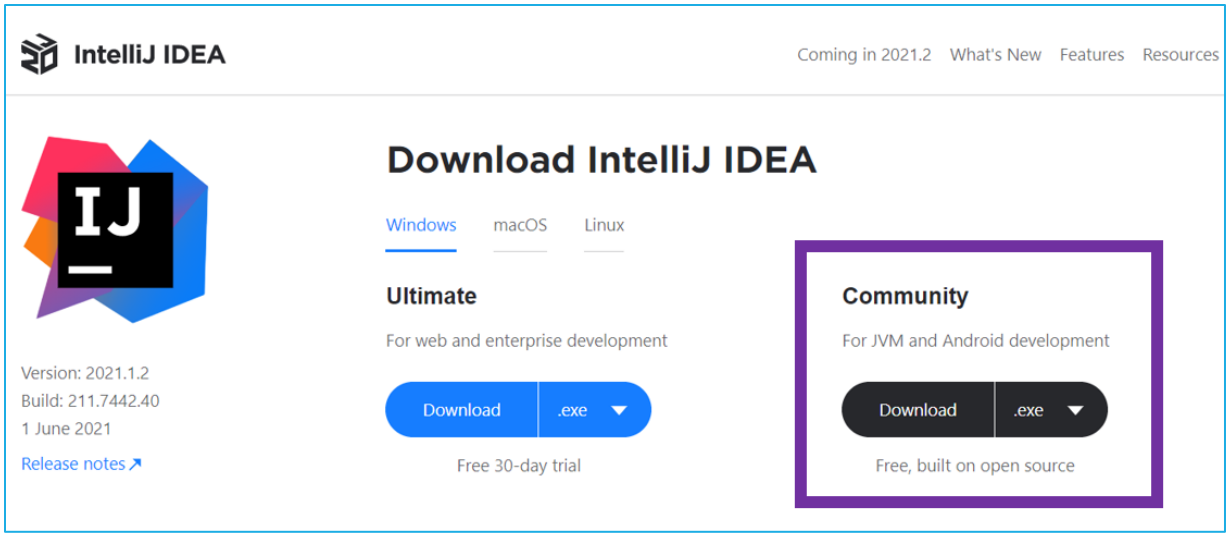
You will need to use Java 8 or higher.

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:



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.

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

MoSCoW	Items and Tasks
Must have	ERD diagram. (Your table design.) Use case diagram. Flow diagrams for each service. Use a build tool to manage all your dependencies and build your app. Your application can connect to your own Data Base. You have created your tables on the Data Base. You have some unit tests. You have a service that can add Miles for a member. You have a service that can view Miles for a member. You have a service that can subtract Miles for a member. You can run your application standalone (via Spring boot). You are using source control.
Should have	You can specify the start date on the service that adds Miles for a member. You have proper error handling. You have made use of proper application layering. You have a Swagger API for your services. You have a code coverage check. You have an optional field on your add service that will cause the service to throw an exception. Your transaction rolls back in the case that an exception occurs. You have some logging.
Could have	You can run your application in Docker. You application can handle more than one Currency (i.e. not just Miles) You can configure your different currencies via a service. You have at least 80% unit tests code coverage. (excluding POJO's)
Will not have	A user frontend (Mobile, Webpage or Desktop application) that calls your services.