SIM GLOBAL EDUCATION

UNIVERSITY OF WOLLONGONG AUSTRALIA

CSCI321 – Project

TECHNICAL REPORT FOR FYP-23-S1-14

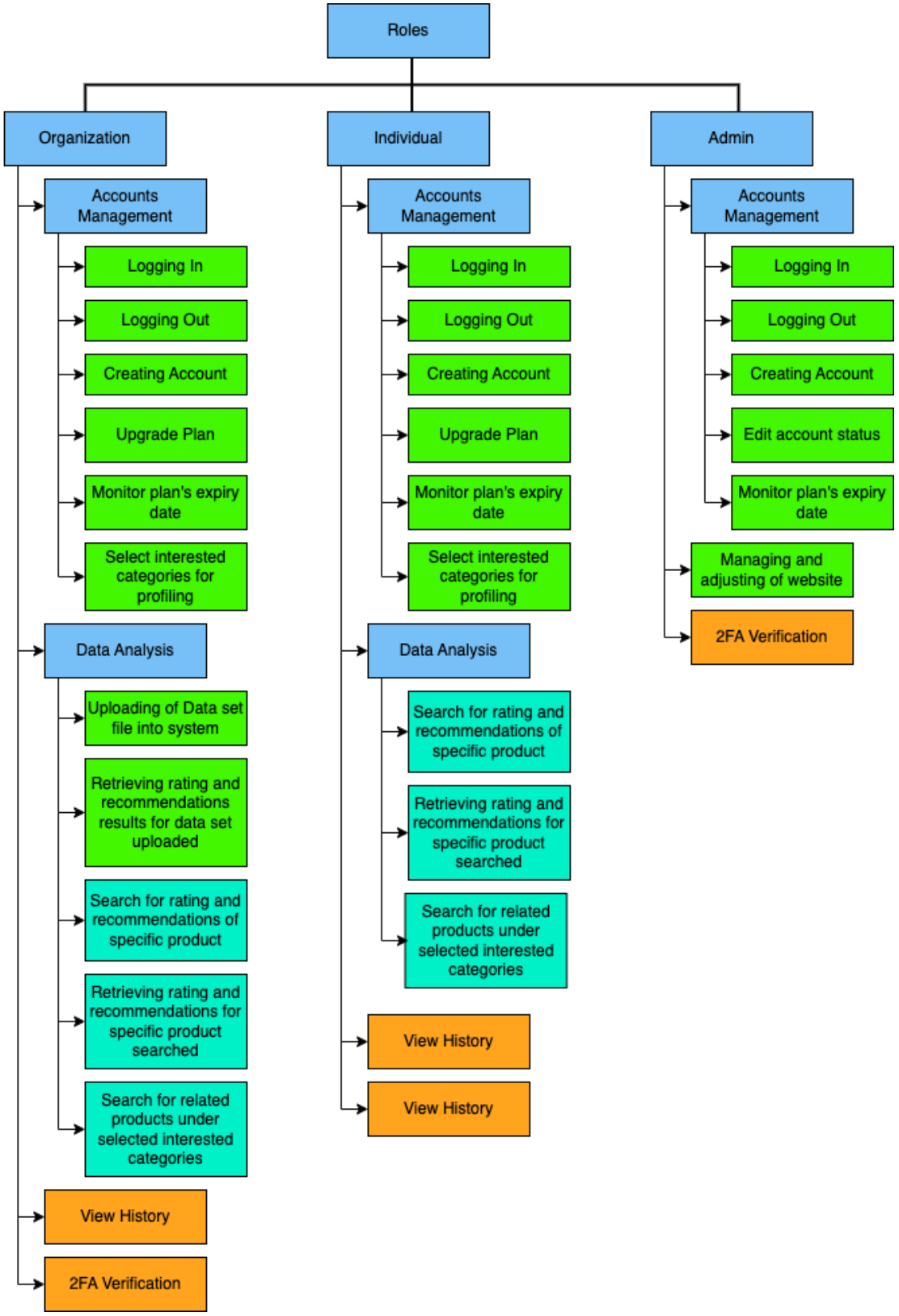
Project Team

Group: FYP-23-S1-14		
Student ID:	Name:	Email:
7222828	Loo Joon Wee	jwloo002@mymail.sim.edu.sg
7372772	Low Rui Hao	rhlrlow001@mymail.sim.edu.sg
7436208	Muhamad Amir Akmal BMS	mohammad103@mymail.sim.edu.sg
7224527	Tham Jia Xuan Travis	jxttham001@mymail.sim.edu.sg
7222439	Wee Zee En Leonard	zelwee001@mymail.sim.edu.sg
Supervisor:	Mr Terence Chew	tchew@uow.edu.au
Accessor:	Dr Prem	premrajan_p@yahoo.com

Table Of Contents

Requirement Specification

Product Features Mind Map



Interface Requirements

User Interface

Our recommender system comes with a simple design and acts as a web dashboard for which the user can use to view the predicted ratings of products and get recommendations for products they can promote to their customers. When they first land on the webpage they will see two textboxes for them to enter username and password, or for new clients/users they can create a new account by clicking sign up in the top right corner of the landing page. The web dashboard they access after login will have a menu accessible at the top of the website to allow for ease of navigation to other pages. The web dashboard will also allow the client/user to easily add a new data set for the recommender system to use as training data with the aid of a simple add button easily located at the top of the web page on the right corner.

Software Interface

Operating System: Windows 10, macOS, Linux

Database: MySQL

Front End: PHP

WebCrawler: Scrapy, Python
Back End: Python

Communications Interface

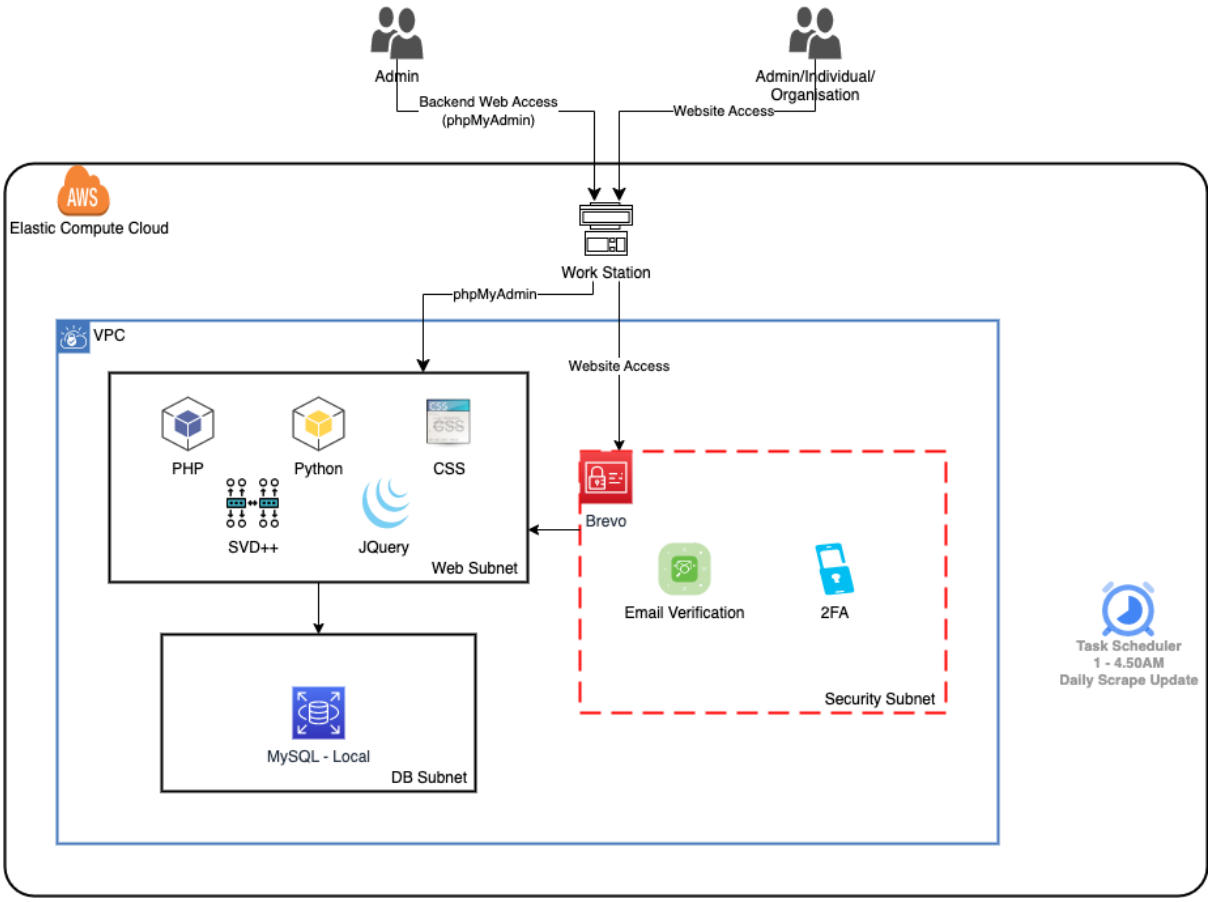
The website dashboard will be accessible to all browsers that support PHP. HTTP/HTTPS

Hardware Interface

The website dashboard does not require any additional or specific hardware as it is able to run on computers as they are.

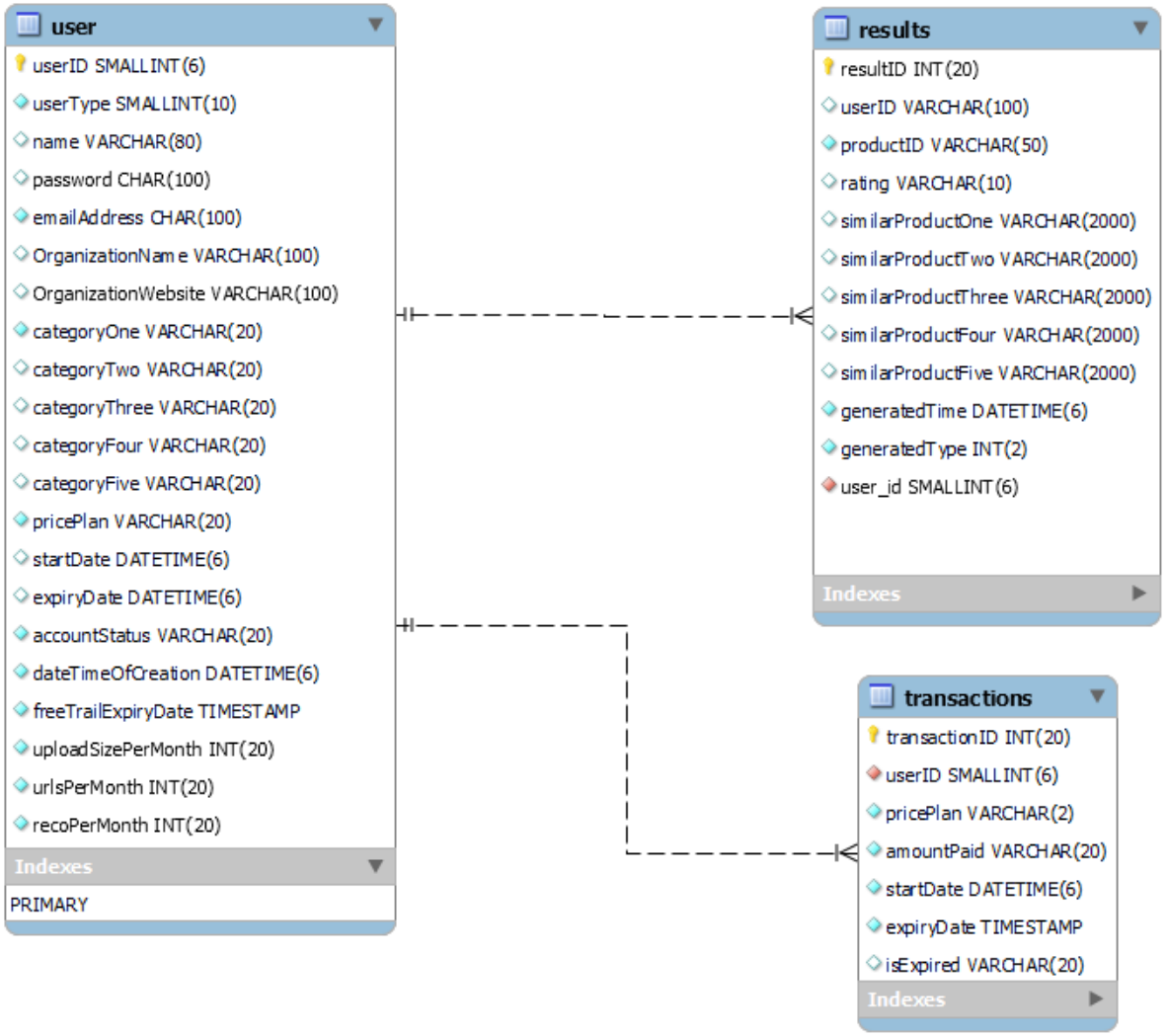
Design Specification

Architecture Design



Database Design

We have chosen MySQL as our main database for storing user data. Below is our Entity Relation Diagram.



In our database design, we have a total of 3 tables. user table which stores all the user information, results table which stores all of the results that were generated and lastly the transactions table where all the transactions are stored.
The tables are in a relationship of 1 user to many results and 1 user to many transactions. The primary keys are userID for user table, resultID for results table and transactionID for transactions table. Foreign key for the results table is user_id which is linked to userID in the user table. Foreign key for the transactions table is userID which is linked to userID in the user table.

Software Design

The creation of the product is heavily based on the integration between PHP, CSS, JavaScript, Python and MySQL.

PHP was chosen for the frontend as it was extremely scalable, helps create dynamic web pages and its learning curve was not steep. CSS is a given as it was needed to customize the web page, and MySQL is one of the most popular database software. Python is one of the most popular web crawling or web scraping languages, and it also supports the prediction model that we are planning to use. JQuery, a framework of JavaScript works well with PHP to help integrate with Python. It also enabled asynchronous calls, meaning that we can run multiple Python scripts from PHP via JQuery without it lagging as much.

Initially, the team was thinking about using HTML and Python, but once we figured out that there was an open source cross-platform program out there known as XAMPP, which offered solutions to displaying PHP, as well as management of MySQL, it made us go that direction instead.

Elaborating further on the web crawling part, we utilized a web crawling framework written in Python, known as Scrapy. Scrapy was a great choice due how scalable it was, and its ability in being able to handle big data at great speed. There was a lot of trial and error involved, and most of these errors were mostly getting blocked by the website that we wanted to crawl, which in this case, was Amazon, or rather Amazon.com.au. To overcome that, we implemented a random proxy selector for each scrape which will take a random IP Address from valid_proxylist.txt, which contains all the valid proxies that we have found. And this indeed did help us to avoid getting blocked by Amazon.com.au. Another consideration we took was finally deciding not to scrape Amazon US, as they have very strict bot detections and force us to "sign-in" and serve us captcha even if we used a headless browser to log in.

How it works is that we have either a text or a csv input script that will extract the links from their original URL and will be checked via regex to get the domain and prodID; they will be added to separate domain_ls, prodID_ls. The crawler will combine the links back into amazon_reviews_url = f'https://{amazon_site}/product-reviews/{asin}/'. scrapy.Request is then called and then it will send the amazon_reviews_url into the parse_review function and pass the metadata of 'asin' and the domain url to be saved into the csv file later.

After that, the crawler extracts the ("div.product-variation-strip") and runs it through a for loop to get all the text from the span values. This will be used later to compare against the review format strip. We then added f'https://{url_amazon}' in front of the website url and extract the image url link and .replace("_AC_US60_", "_AC_US240_") if it is available in the url so that we could have a higher resolution picture. The program will then get the name of the product and extract the reviews from ("#cm_cr-review_list div.review"), where the reviews information will be extracted from a for loop. Furthermore, the crawler will get the user links from a review, check against product-variation-strip and if it is the same, scrape the info and user_link being available, and if product variation strip is None and user_link being available, we will scrape the data which consists of userID, prodID, rating, domain, image_url and prodname. Lastly, the crawler will check whether there is any next page url that we are able to scrape. If there is "/ap/signin", we will set the next_page_short_url as none. If both next_page_short_url and user_link is not None, the crawler will do scrapy.Request on parse_review function to extract the previous information again.

For the category scraping, we use links that are from the subcategories of the category that we chose, which are computers, electronics, pets, toys and videogames. We will make use of a modify_settings script to modify the settings.py of scrapy. This is to ensure the host and referrer in the DEFAULT_REQUEST_HEADERS are similar to the request headers that any web browser uses. We will use scrapy to extract the product links from those subcategories that we found that allow us to extract links from, with a maximum limit of 100, However, as the pages are loaded in either 16 or 24 links, they will exceed limit, so the maximum number of links scraped will be at 120. There will be a delete script to do some data cleaning so that we

will not have an overlapping of data. We will run a specialcharfilter.py to remove unwanted special characters from the prodname column, as it is difficult for php to display unwanted special characters. We thought about the website running in the background, so we decided to scrape these into a processing folder after that we will merge the data together with the original data without duplicates.

Svdpp is the prediction model we are using, and it runs using Python as well. It takes into account implicit ratings, and is similar to singular value decomposition (SVD) in terms of usage of stochastic gradient descent (SGD). The program utilizes the regularized squared error objective to train the parameters through SGD. There will be more info on how the program works in detail below.

Moving on from the web crawling and prediction model to the website implementation, first of all, during the development of various aspects of the website, PHP and CSS are the primary languages utilized for the majority of the website's components. Some Javascript was also used to help facilitate dynamic requests, like the changing of screen upon clicking a button without the need to make a server request. Other than those, every design user can view, be in the login, sign up page, home or main page, they are all built using PHP and CSS.

The main idea here is to build a functional website that looks visually appealing while carrying out its core functions at a good to high level. We understood the requirements of our product and identified the key functionalities that we had to implement.

PHP worked hand in hand with MySQL to get user information to display on screen. For instance, for management of users, the admin user will get to view all the users which have signed up on the website, which allows for easier management. Things like the creating, editing, deleting of users are also enabled with the assistance of MySQL. And all these are actually part of the core functionalities of our product. Taking that into consideration, all the core functionalities are actually grouped under a class in PHP called 'User'. In this case, there would be an edit user, create user, get user and delete user function. This benefits the programming process as we can just call the function if we need it again.

Once a user has signed up, logged in, they will be able to access the workspace, which is the place where users can add files, crawl information, generate recommendations and ratings prediction. To be able to get the results after putting the user and product information, the PHP would have to use a shell command to run the Python script, before gathering the information and then displaying as results to the user. This Python includes both the crawling script as well as the prediction model script.

Initially, the website was lagging every time we tried to execute the shell command. After some research, it was discovered that this happens because the shell command was executing on the same php page that the user is currently on. With this revelation in mind, the team decided that JQuery, which is a Javascript framework, and AJAX, which stands for Asynchronous JavaScript and XML, would help to minimize this issue. And as such that is implemented in such a that when user clicks generate, the information will be saved as JQuery variables, and JQuery uses AJAX to send this information to another PHP script which

will execute the python shell command in the background without interfering with what's happening in the current page. Once the PHP script executes whatever that's sent over from JQuery, it sends back the information and JQuery will display the information to the user.

When users see this information, they will only be able to see the frontend portion of it, which will be the product names and images. They won't even catch any glimpse of anything that runs in the backend. After all, we prioritize user experience, which includes easy-to-use services, alongside good to great visuals.

SVDpp_input_csv program:

All the checking will be encompassed under the start() function, until we pass variables into svdppcv3_gs function.

We will start by calling the python script with a directory of the csv(s) as the first argument. If length of sys.argument is more than 3, and if the second argument is “product”, it will assign the 3rd variable as prodID. If the length of the argument is more than 5, and if the fourth argument is “user”, it will assign the fifth variable as userID. If the userID is less than 13, it will output to tell the user to “User ID usage incorrect, please use another user ID” and exit the program. If length of argument is less than 6, it will assign the fourth variable to arg_no_of_reco, required number to display for similar recommendation and set arg_userID = ‘’. If the prodID length is not between 7 and 12, the program will say Product ID usage incorrect, please use another product ID, and exit the program.
Skip SVDpp_input_csv_5 program portion, go to Continue from here.

SVDpp_input_csv_5 program:

There will start by calling the python script with the directories of the csv(s) as the five arguments being passed. If length of sys.argument is more than 7, and if the sixth argument is “product”, it will assign the 7th variable as prodID. If the length of the argument is more than 9, and if the eighth argument is “user”, it will assign the ninth variable as userID and arg_no_of_reco = 0. If the userID is less than 13, it will output to tell the user to “User ID usage incorrect, please use another user ID” and exit the program. If length of argument is less than 10, it will assign the eighth variable to no_of_reco, required number to display for similar recommendation and set arg_userID = ‘’. If the prodID length is not between 7 and 12, the program will say Product ID usage incorrect, please use another product ID, and exit the program.

Continue from here:
The program will find file(s) ending with .csv in the directory argument, and append it to a list called csv_files. Next, it will access the data in those .csv file(s), if it is not empty, and store the data in merged_df.

We will use df as the main calling point rather than merged_df, “df = merged_df”.

Extract_user_id(href) function will be used on userID by using regex. Using the commands, if re.search(r'amzn1\.account\.(\w+)', href), has a variable, they will return re.search(r'amzn1\.account\.(\w+)', href).group(1).

df.dropna() will be used to drop all the null values.
df.drop_duplicates() will be used to remove duplicates.

We will check for the label headers for the first 3 column data, if the first column of the dataframe is not equal to 'userID' and the second column is not equals to 'prodID' and third column is not equals to 'rating'. We will use df.rename(columns=rename_ID), rename_ID function will check on the length >13, and return ‘userID’, it will check whether the length is between 7 to 12 and return ‘prodID’, it will check whether the x == ‘5.0’ or x=='4.0' or x=='3.0' or x=='2.0' or x=='1.0' and return ‘rating’. It will just return x, as it does not fulfil these criterias.

Get the df_shape, and if the df_shape[1] is more than 3, it will slice the df, to only have the first 3 columns.

if (df_shape[0]>10000) then we will sample it to 10000, df = df.sample(10000), else if (df_shape[0]>1000) then we will sample it to 1000, df = df.sample(1000)

Set reader to the default settings.

Pass df, reader, arg_userID, arg_prodID, arg_no_of_reco into svdppcv3_gs function.

In svdppcv3_gs, we will use Dataset.load_from_df and extract 'userID', 'prodID', 'rating' from df, and pass the reader into it, and equate it to data.

We will use build_full_trainset on data, and equate it to train_set.

We will use the train_set and build_testset on it to check the RMSE values and call it data_testset.

We create a basic svdpp() model that is called svdppalgo.

We will use svdppalgo to fit the train_set.

We use svdppalgo.test(data_testset) and call it test_Pred,

Using test_Pred on accuracy.rmse value to compute an RMSE value, and name it rmseVal_ori for the basic svdpp.

We will go on to use param_grid to tune the gridsearchCV.
We will use this param_grid based on previous runtime speed and RMSE outputs for the tuned values.

```
# hyperparameters
param_grid = {"n_factors": [10, 15],
              "n_epochs": [30, 60],
              "lr_all": [0.002, 0.004],
              "reg_all": [0.04, 0.07]
              }
```

We will use gridsearch at cv=3, n_jobs=-1 will use all cores in the system and put the param_grid and SVDpp in the GridSearchCV
gs_cv3 = GridSearchCV(SVDpp, param_grid, measures=["rmse", "mae"], cv=3, n_jobs=-1)

We will fit the data into the gs_cv3.

We will use best_estimator for rmse to get the lowest value out of the param grid that was in the gridsearch, svd_pp_cv3 = gs_cv3.best_estimator["rmse"]

We will fit the train_set into svd_pp_cv3.

We use svd_pp_cv3.test(data_testset), to compute an accuracy.rmse value, and name it bestpred_cv3 for the tuned svdpp.

Using bestpred_cv3 on accuracy.rmse to compute an RMSE value, and name it as rmseVal_cv.

If rmseVal_cv is more than rmseVal_ori, we will use the original base SVD++ parameters and run the prediction.

If the length of sys.argv is more than 5, we will print out the RMSE value for rmseVal_ori.

And run the svdppalgo.predict on uid=arg_userID, iid=arg_prodID, and r_ui = None with .est and call it norm_pred.

We will print out the rating prediction (f"Rating prediction: {round(norm_pred)}")

If the length of sys.argv is less than 6, we will pass arg_prodID, df, train_set, svdppalgo, int(no_of_reco) into similar_products function

In similar_products, we will get the list of all item IDs in the dataset.

Run it through the algo to get the predicted ratings for each item for the userID = 0.

Sort the predictions by rating and extract the top n recommended item IDs and exclude the prodID that is being recommended on.

Get the names of the recommended items and drop duplicates.

Create a list of dictionaries containing the recommended item IDs and names

Return the list to topReco_prod

Print out the top recommended items from topReco_prod.

start() at the end to start the program.

Testing Documentation

Test Plan

Introduction
This document is a test plan whereby it is used to communicate the test approach to team members. It includes the objectives, scope, schedule, risks, and approach. This document will clearly identify what the test deliverables will be and what is deemed in and out of scope.

Objectives
The test team is responsible for testing the website service and ensuring it meets the requirements and is of high quality. The test team will be acting as organization, individuals as well as the admin in this project.

The team will create and conduct test cases on every functionality of the website service. Must have functionality is considered more important than the delivery date in this project.

Quality Assurance Team Members

Name	Role
Loo Joon Wee	QA Team Lead, QA Analyst, QA Tester
Muhamad Amir Akmal BMS	QA Analyst, QA Tester

Scope
Testing for all 'must have' requirements are set as priority. These and any other requirements that get included must all be tested. At each test sprint, the tester must:

1. Create test cases that caters to individual functionality with as detailed steps as possible.
2. Conduct the test according to the test case's steps.
3. Record down the results and enter appropriate comments.
4. Document test report
5. Ensure requirements are met.

Functional Testing:
- Verify that all user stories and requirements are met.
- Test all functional requirements to ensure that they meet the expected behavior and functionality.

- Verify that the software is compatible and functional in all intended environments, platforms, and browsers.
- Verify that the software handles errors and exceptions correctly.
- Verify that the software performs the required outcome after integration with third-party systems, databases, and services.

Non-Functional Testing:

- Conduct usability testing to ensure that the website service is easy to use and understand.
- Conduct security testing to ensure that the website service is secure from unauthorized access or users with unauthentic profile information (Non-existent organization or email of individual)
- Conduct compatibility testing to ensure that the website service works correctly with all intended operating systems and hardware.

Regression Testing:

- To ensure that previously tested functionalities continue to function correctly after changes, improvements or debugging have been made to the website and software.

System Testing:

- To ensure that the website service works as a whole after all different modules and functions are being integrated together.
- To ensure that the website service as a whole meets all the specified requirements and performs as expected.

Acceptance Testing:

- To ensure that the website service meets the requirements of the client and end-user.
- To assure that the service is ready for publish.

Deliverables:

- Test plan outlining scope, testing approach, risk assessment, and schedule
- Test cases
- Test results and reports

Roles and Responsibilities:

- The quality assurance team is responsible for developing and executing test cases, identifying, and reporting bugs, as well as providing updates.
- The development team is responsible for fixing bugs, addressing and solving issues identified during testing.
- The product owner is responsible for verifying that the website service meets all acceptance criteria.

Assumptions / Risks

Assumptions

This section lists assumptions that are made specific to this project.

1. Functions that meant to be tested, are implemented, and published on the website.
2. All the content on the website and test cases are in English.
3. Testers and users understand English.
4. Website and services accessible to any OS platform and browsers.
5. Testers and users have access to internet.
6. Test data will be available and accurate.
7. The software will be stable and free of major bugs.

Risks

The following risks have been identified and the appropriate action identified to mitigate their impact on the project. The impact of the risk is based on how the project would be affected if the risk was triggered. The consequence is what event would cause the risk to become an issue to be dealt with.

No	Risk	Impact	Consequence	Mitigation Plan
1	Delay in the delivery of functional software features	Medium	Affect testing schedule	Skip to next available user story functionality and conduct respective tests on it and return to the planned function testing when the development team is ready.
2	The software may have critical bugs that requires long debugging duration	Medium	Affect testing schedule	Same as previous mitigation plan, move on to subsequent functional testing first or documentation updates while waiting for development team.
3	Team members may face unexpected personal issues that delays their work production	High	Affect testing schedule and development schedule	Other team members help to cover up the required work production while waiting for the team member to be back. So, the whole team doesn't fall behind too much.
4	Third party integrations may not work as expected or functionalities might not work as expected	High	Affect testing schedule and development schedule	If confirmed on third party integrations being unable to work as planned, for example hosting platforms, immediately switch to

				next better option that was researched in the documentations.

Test Approach

- Test often throughout the development cycle to ensure that issues are identified and addressed as early as possible.
- Test in parallel with development to ensure that issues are identified and addressed in a timely manner.
- Focus on critical functionality first, followed by testing of remaining functionalities or good to have functionalities.

Manual testing

Automated tests are part of the development process, but no automated functional tests are planned for our projects, we will only be conducting manual testing.

Our quality assurance team will be manually testing all the test cases, record down the results and feedback accordingly.

Metrics to measure usability testing:

To define measurement of whether usability testing pass/fail:

- Success Rate: Percentage of testers and users being able to complete certain task without assistance or guidance from developer team

- Efficiency Rate: Percentage of testers and users that agrees that they are able to complete task smoothly without having to hesitate or trial and error around the navigation of the whole service

- User Experience: Level of difficulty to navigate through the website, level of difficulty to adapt and understand the usage of the software, level of satisfaction after using the software, level of engagement and interest the users have with the software

Test Environment

The service will be integrated in the website, hosting on AWS platform. So, the testing will be done through website, with the presence of internet. However, early testing will be conducted through local host, as the hosting on AWS will be integrated at a later stage.

After hosting on AWS platform, we will be running integration testing, regression testing, system testing, acceptance testing again. It will be conducted using desktop/laptop, with the presence of internet, on both windows and IOS

Testing website link:

- fyprecs.com

Hardware and software requirements:

- Desktop/Laptop (MacOS, Windows)
- Browsers (Google Chrome, Internet Explorer, Firefox, Safari)

Dependencies:

- MySQL database
- Connected working internet access

Preconditions:

- Ensure the hardware, software and dependencies are all installed and present

Timeline

Test Schedule

- **Sprint 6:**
    - **Week 2:**
        - Create test plan
        - Create test cases for user stories on all aspects of account management for all types of users
        - Begin parts of functional testing for user stories on account management
        - Begin regression testing on previously tested user stories

- **Sprint 7:**
    - **Week 1:**
        - Continue and finish up functional testing on remaining user stories on account management
        - Continue and finish up regression testing on previously tested user stories
        - Conduct and finish integration testing for previous tested user stories that are integrated together
        - Create test cases for user stories on security features aspects, as well as rating and prediction function aspects of all users
        - Conduct and finish functional testing for user stories on above mentioned aspects

        - Conduct and finish regression testing on previously above tested user stories
        - Conduct and finish integration testing for previous tested user stories

    - **Week 2:**
        - Create test cases on user stories of recommendation and web scraping based on category
        - Conduct and finish functional testing for the above mentioned user stories
        - Conduct and finish integration testing of certain above mentioned user stories
        - Conduct and finish regression testing on previously tested user stories and ensure every user stories tested and integrated so far are working together as a whole
        - Create test cases on user stories of security functions
        - Begin functional testing of security function test cases
        - Begin integration testing of above tested security test cases

- **Sprint 8:**
    - **Week 1:**
        - Continue and finish functional testing of security function test cases
        - Continue and finish integration testing on above test cases
        - Conduct and finish final debugging, implementations, build testing and polishing
        - Conduct final regression testing for all above tested user stories
        - Conduct system testing as a whole to ensure the software is ready for publishment
        - Conduct acceptance testing to ensure the software meets the criteria

Deliverables

Deliverable	**By**
Test Plan	Joon Wee
Test Cases	Joon Wee, Amir

Test Results	Joon Wee, Amir
Test Status report	Joon Wee, Amir

Test Cases

Account Management:

ID	Case Category	Sub Case Description	Responsible tester	Status(Pass/Fail/not executed)
1	Account Management		Joon Wee	Pass
1.0.0		Account Registration for Individual (Correct Credentials)	Joon Wee	Pass
1.0.1		Account Registration for Individual (Wrong Credentials, used email and invalid matching password)	Joon Wee	Pass
1.0.2		Account Registration for Organisation (Correct credentials)	Joon Wee	Pass
1.0.3		Account Registration for Organisation (Wrong credentials)	Joon Wee	Pass
1.1.0		Account Login for Individual (Correct Credentials)	Joon Wee	Pass
1.1.1		Account Login for Individual (Wrong Credentials)	Joon Wee	Pass
1.1.2		Account Login for Organisation (Correct Credentials)	Joon Wee	Pass
1.1.3		Account Login for Organisation (Wrong Credentials)	Joon Wee	Pass
1.1.4		Account Login for Admin (Correct Credentials)	Joon Wee	Pass
1.1.5		Account Login for Admin (Wrong Credentials)	Joon Wee	Pass
1.2.0		Change interest categories for Individual (User Profile after registration)	Joon Wee	Pass
1.2.1		Change categories for products sold for Organisation (User Profile after registration)	Joon Wee	Pass
1.3.0		Delete accounts (Admin)	Joon Wee	Pass
1.3.1		Edit account's plan (Admin)	Joon Wee	Pass
1.3.2		Edit account's status (Admin)	Joon Wee	Pass
1.3.3		Edit account's email (Admin)	Joon Wee	Pass
1.3.4		Edit account's password (Admin)	Joon Wee	Pass
1.3.5		Edit account's name (Admin)	Joon Wee	Pass
1.3.6		Edit account's user type (Admin)	Joon Wee	Pass
1.3.7		Edit account's interest categories (Admin)	Joon Wee	Pass
1.3.8		Create Accounts (Admin)	Joon Wee	Pass
1.3.9		Search accounts by user type or user ID (Admin)	Joon Wee	Pass
1.4.0		Monitor plan's expiry date (Individual)	Joon Wee	Pass
1.4.1		Monitor plan's expiry date (Organisation)	Joon Wee	Pass
1.4.2		Monitor plan's quota (Individual)	Joon Wee	Pass
1.4.3		Monitor plan's quota (Organisation)	Joon Wee	Pass
1.4.4		Monitor account's transaction history (Individual)	Joon Wee	Pass
1.4.5		Monitor account's transaction history (Organisation)	Joon Wee	Pass
1.5.0		Changing account's name (Admin)	Joon Wee	Pass
1.5.1		Changing account's password (Admin)	Joon Wee	Pass
1.5.2		Changing account's name (Individual)	Joon Wee	Pass
1.5.3		Changing account's password (Individual)	Joon Wee	Pass
1.5.4		Changing account's name (Organisation)	Joon Wee	Pass
1.5.5		Changing account's password (Organisation)	Joon Wee	Pass
1.5.6		Changing account's organisation name (Organisation)	Joon Wee	Pass
1.5.7		Changing account's organisation website (Organisation)	Joon Wee	Pass
1.6.0		Upgrading of plan (Individual)	Joon Wee	Pass
1.6.1		Upgrading of plan (Organisation)	Joon Wee	Pass
1.7.0		Logging Out (Admin)	Joon Wee	Pass
1.7.1		Logging Out (Individual)	Joon Wee	Pass
1.7.2		Logging Out (Organisation)	Joon Wee	Pass

Security:

ID	Case Category	Sub Case Description	Responsible tester	Status(Pass/Fail/not executed)
2	Security		Amir	Pass
2.1.1		Test Login button on the top right (next to sign up)	Amir	Pass
2.1.2		Test Login button in the middle left (next to sign up now)	Amir	Pass
2.2.1		Login using correct email and password for Individual	Amir	Pass
2.2.2		Login using incorrect email and correct password for Individual	Amir	Pass
2.2.3		Login using correct email and incorrect password for Individual	Amir	Pass
2.2.4		Try Individual login in with the correct password but the alphabets in different Caps	Amir	Pass
2.2.5		Try Individual login with email in caps	Amir	Pass
2.2.6		Try Individual login in with empty email field and correct password	Amir	Pass
2.2.7		Try Individual login in with correct email and empty password field	Amir	Pass
2.2.8		Try to bypass 2FA verification for Individual login by trying to enter without the verification code	Amir	Pass
2.2.9		Enter correct 2FA code sent by RECS site for Individual login	Amir	Pass
2.2.10		Enter incorrect 2FA Code for Individual login	Amir	Pass
2.2.11		Enter partial 2FA Code for Individual login	Amir	Pass
2.2.12		Click on "RECS" on top left of the 2FA verification page for Individual login	Amir	Pass
2.3.1		Login using correct email and password for Organization	Amir	Pass
2.3.2		Login using incorrect email and correct password for Organization	Amir	Pass
2.3.3		Login using correct email and incorrect password for Organization	Amir	Pass
2.3.4		Try Organization login in with the correct password but the alphabets in different Caps	Amir	Pass
2.3.5		Try Organization login with email in caps	Amir	Pass
2.3.6		Try Organization login in with empty email field and correct password	Amir	Pass
2.3.7		Try Organization login in with correct email and empty password field	Amir	Pass
2.3.8		Try to bypass 2FA verification for Organization login by trying to enter without the verification code	Amir	Pass
2.3.9		Enter correct 2FA code sent by RECS site for Organization login	Amir	Pass
2.3.10		Enter incorrect 2FA Code for Organization login	Amir	Pass
2.3.11		Enter partial 2FA Code for Organization login	Amir	Pass
2.3.12		Click on "RECS" on top left of the 2FA verification page for Organization login	Amir	Pass
2.4.1		Login using correct email and password for Admin	Amir	Pass
2.4.2		Login using completely incorrect email for Admin	Amir	Pass
2.4.3		Login using completely incorrect password for Admin	Amir	Pass
2.4.4		Try Admin login in with the correct password but the alphabets in different Caps	Amir	Pass
2.4.5		Try Admin login with email in caps	Amir	Pass
2.4.6		Try Admin login in with empty email field and correct password	Amir	Pass
2.4.7		Try Admin login in with correct email and empty password field	Amir	Pass
2.4.8		Try to bypass 2FA verification for Admin login by trying to enter without the verification code	Amir	Pass
2.4.9		Enter correct 2FA code sent by RECS site for Admin login	Amir	Pass
2.4.10		Enter incorrect 2FA Code for Admin login	Amir	Pass
2.4.11		Enter partial 2FA Code Admin login	Amir	Pass
2.4.12		Click on "RECS" on top left of the 2FA verification page for Admin login	Amir	Pass
2.5.1		Upon landing on the login page, multiple fast clicks on Sign In button	Amir	Pass
2.5.2		Enter a valid email and password, then click "sign in" button multiple times	Amir	Pass
2.5.3		Click on "Forgot Password?"	Amir	Pass
2.5.4		On "Forget Password" page, click verify with empty email field	Amir	Pass
2.5.5		On "Forget Password" page, enter incorrect email	Amir	Pass
2.5.6		On "Forget Password" page, enter correct email	Amir	Pass
2.5.7		On verify email page, enter the verification code received	Amir	Pass
2.5.8		On "Forget Password" page, leave password field blank and try to update	Amir	Pass
2.5.9		On "Forget Password" page, enter different passwords into the two fields	Amir	Pass
2.5.10		On "Forget Password" page, entersame passwords but not in suggested format	Amir	Pass
2.5.11		On "Forget Password" page, enter correct new passwords	Amir	Pass

Trial Feature:

ID	Case Category	Sub Case Description	Responsible tester	Status(Pass/Fail/not executed)
3	Trial Feature		Joon Wee	Pass
3.0.0		Choosing of category for recommendation service (Free Trial)	Joon Wee	Pass
3.0.1		Generating recommendations from product chosen in the sample (Free Trial)	Joon Wee	Pass
3.0.2		Viewing of interested product from recommendation (Free Trial)	Joon Wee	Pass

Website Introduction:

ID	Case Category	Sub Case Description	Responsible tester	Status(Pass/Fail/not executed)
4	Website Introduction		Joon Wee	Pass
4.0.0		Viewing of features of our service	Joon Wee	Pass
4.0.1		Checking how the website and our service works	Joon Wee	Pass
4.0.2		Checking what are the features and how it works for individual user	Joon Wee	Pass
4.0.3		Checking what are the features and how it works for organisation user	Joon Wee	Pass
4.1.0		Reach the documentation page where it shows the detailed explanations of each features and price plan	Joon Wee	Pass
4.1.1		Reach the page for detailed explanation for how it works	Joon Wee	Pass
4.1.2		Reach the page for detailed explanation for price plans	Joon Wee	Pass
4.1.3		Reach the FAQ page	Joon Wee	Pass
4.1.4		Reach the About Us page	Joon Wee	Pass
4.1.5		Reach the workspace page (Individual)	Joon Wee	Pass
4.1.6		Reach the instruction page for workspace (Individual)	Joon Wee	Pass
4.1.7		Reach the workspace page (Organisation)	Joon Wee	Pass
4.1.8		Reach the instruction page for workspace (Organisation)	Joon Wee	Pass

Web scraping:

ID	Case Category	Sub Case Description	Responsible tester	Status(Pass/Fail/not executed)
5	Webscraping		Joon Wee	Pass
5.0.0		Add list of URL into service to scrape (Individual)	Joon Wee	Pass
5.0.1		Add list of URL into service to scrape (Individual) V2	Joon Wee	Pass
5.0.2		Adding same/existing list of url (Individual)	Joon Wee	Pass
5.0.3		Upload empty file (Individual)	Joon Wee	Pass
5.0.4		Check uploaded file list (Individual)	Joon Wee	Pass
5.0.5		Check uploaded file list (Individual) V2	Joon Wee	Pass
5.0.6		Delete uploaded file list (Individual)	Joon Wee	Pass
5.0.7		Exceed uploaded file list (Individual)	Joon Wee	Pass
5.1.0		Add list of URL into service to scrape (Organisation)	Joon Wee	Pass
5.1.1		Add list of URL into service to scrape (Organisation) V2	Joon Wee	Pass
5.1.2		Adding same/existing list of url (Organisation)	Joon Wee	Pass
5.1.3		Upload empty file (Organisation)	Joon Wee	Pass
5.1.4		Check uploaded file list (Organisation)	Joon Wee	Pass
5.1.5		Check uploaded file list (Organisation) V2	Joon Wee	Pass
5.1.6		Delete uploaded file list (Organisation)	Joon Wee	Pass
5.1.7		Exceed uploaded file list (Organisation)	Joon Wee	Pass
5.2.0		Crawl the links in the file uploaded (Individual)	Joon Wee	Pass
5.2.1		Crawl the links in the file uploaded (Organisation)	Joon Wee	Pass
5.3.0		Retrieve the product ID of product to rate and recommend (Individual)	Joon Wee	Pass
5.3.1		Retrieve the product ID of product to rate and recommend (Individual) V2	Joon Wee	Pass
5.3.2		Retrieve the product ID of product to rate and recommend (Organisation)	Joon Wee	Pass
5.3.3		Retrieve the product ID of product to rate and recommend (Organisation) V2	Joon Wee	Pass
5.3.4		Retrieve the user ID of own account or other user (Individual)	Joon Wee	Pass
5.3.5		Retrieve the user ID of own account or other user (Individual) V2	Joon Wee	Pass
5.3.6		Retrieve the user ID of own account or other user (Organisation)	Joon Wee	Pass
5.3.7		Retrieve the user ID of own account or other user (Organisation) V2	Joon Wee	Pass
5.4.0		Generate recommendations (Individual)	Joon Wee	Pass
5.4.1		Generate recommendations (Organisation)	Joon Wee	Pass
5.4.2		Generate recommendations with wrong Product ID (Individual)	Joon Wee	Pass
5.4.3		Generate recommendations with wrong Product ID (Organisation)	Joon Wee	Pass
5.5.0		Generate ratings prediction (Individual)	Joon Wee	Pass
5.5.1		Generate ratings prediction (Organisation)	Joon Wee	Pass
5.5.2		Generate ratings prediction with wrong User ID (Individual)	Joon Wee	Pass
5.5.3		Generate ratings prediction with wrong Product ID (Individual)	Joon Wee	Pass
5.5.4		Generate ratings prediction with wrong User ID (Organisation)	Joon Wee	Pass
5.5.5		Generate ratings prediction with wrong Product ID (Organisation)	Joon Wee	Pass
5.6.0		View history results of recommendations (Individual)	Joon Wee	Pass
5.6.1		View history results of recommendations (Organisation)	Joon Wee	Pass
5.6.2		View history results of rating prediction (Individual)	Joon Wee	Pass
5.6.3		View history results of rating prediction (Organisation)	Joon Wee	Pass

Data Set:

ID	Case Category	Sub Case Description	Responsible tester	Status(Pass/Fail/not executed)
6	Data Set		Amir	Pass
6.1.1		Test "Add Data Set" button on the Discover page	Amir	Pass
6.1.2		Test "Add Data Set" on the Side navigation menu	Amir	Pass
6.1.3		On "Add Data Set" page press "upload" button to upload my own data file	Amir	Pass
6.1.4		On "Add Data Set" page press "Choose file" link to upload my own data file	Amir	Pass
6.1.5		On "Add Data Set" try to drag and drop my own data file	Amir	Pass
6.1.6		Click in the upload box	Amir	Pass
6.1.7		Try upload my own data set in different file format from the recommended	Amir	Pass
6.1.8		Try upload multiple files	Amir	Pass
6.1.9		Try upload same file again while the file is still stored in RECS	Amir	Pass
6.1.10		After adding a file instead of uploading, add another file	Amir	Pass
6.2.1		Test "Uploaded Data Set" button on the Discover page	Amir	Pass
6.2.2		Test "Uploaded Data Set" button on the Side navigation menu	Amir	Pass
6.2.3		Search for a particualr file	Amir	Pass
6.2.4		clear search bar	Amir	Pass
6.2.5		Sort by File Name	Amir	Pass
6.2.6		Sort by File Size	Amir	Pass
6.2.7		Sort by Date Uploaded	Amir	Pass
6.2.8		press delete option	Amir	Pass

Ratings Prediction:

ID	Case Category	Sub Case Description	Responsible tester	Status(Pass/Fail/not executed)
7	Ratings Prediction		Amir	Pass
7.1.1		Test to see if the the Generate Ratings tab (on Discover page) is working for the Individual user account	Amir	Pass
7.1.2		Test to see if the the Generate Ratings (RECS' Data) tab (in Side Navigation menu) under "Our Data" section is working	Amir	Pass
7.1.3		Test to see if the the Generate Ratings (Uploaded URL) tab (in Side Navigation menu) under "Our Data" section is working	Amir	Pass
7.1.4		Check if the implemented tooltips work in Generate Ratings (RECS' Data)	Amir	Pass
7.1.5		Check if hyperlinks for how to get User ID and Product ID worksGenerate Ratings (RECS' Data)	Amir	Pass
7.1.6		Check if the implemented tooltips work in Generate Ratings (Uploaded URL)	Amir	Pass
7.1.7		Check if hyperlinks for how to get User ID and Product ID worksGenerate Ratings (Uploaded URL)	Amir	Pass
7.1.8		Leaving the User ID and Product ID field empty, try generating Rating Prediction (RECS' Data)	Amir	Pass
7.1.9		while both User ID and Product ID fields are empty, spam multiple fast clicks on "Generate" button (RECS' Data)	Amir	Pass
7.1.10		Leave the User ID empty but fill the Product ID field then try generating Rating Prediction (RECS' Data)	Amir	Pass
7.1.11		Fill in the User ID and leave the Product ID field empty, then try to generate Rating Prediction (RECS' Data)	Amir	Pass
7.1.12		Enter both User ID and Product ID to generate Rating prediction (RECS' Data)	Amir	Pass
7.1.13		Try using User ID from other websites (RECS' Data)	Amir	Pass
7.1.14		Try Product ID from other websites (RECS' Data)	Amir	Pass
7.1.15		Leaving the User ID and Product ID field empty, try generating Rating Prediction (Uploaded URL)	Amir	Pass
7.1.16		Leave the User ID empty but fill the Product ID field then try generating Rating Prediction (Uploaded URL)	Amir	Pass
7.1.17		Fill in the User ID and leave the Product ID field empty, then try to generate Rating Prediction (Uploaded URL)	Amir	Pass
7.1.18		Enter both User ID and Product ID to generate Rating prediction(Uploaded URL)	Amir	Pass
7.1.19		After generating ratings go to the "Discover" page, checking to see "Recent Searches"	Amir	Pass
7.1.20		On "Discover" Page, click "See All Results"	Amir	Pass
7.1.21		Go to "History" in side navigation menu and click on "Results" tab	Amir	Pass
7.1.22		Search for a particular result	Amir	Pass
7.1.23		In "Results" click sort by Result ID	Amir	Pass
7.1.24		In "Results" click sort by sort by Date Generated	Amir	Pass
7.1.25		In "Results" click sort by sort by Generated Type	Amir	Pass
7.1.26		In "Results" click results option	Amir	Pass
7.1.27		In "Results" click filter button "Ratings Prediction (Your Data)	Amir	Pass
7.1.28		In "Results" click "All" button	Amir	Pass
7.1.29		In "Results" click filter button "Ratings Prediction (RECS' Data)	Amir	Pass
7.2.1		Test to see if the the Generate Ratings tab (on Discover page -> "Your Data") is working for the Organization user account	Amir	Pass
7.2.2		Test to see if the the Generate Ratings tab (on Discover page -> "Our Data") is working for the Organization user account	Amir	Pass
7.2.3		Test to see if the the Generate Ratings tab (in Side Navigation menu) under "Your Data" section is working	Amir	Pass
7.2.4		Test to see if the the Generate Ratings tab (in Side Navigation menu) under "Our Data" is working	Amir	Pass
7.2.5		Test to see if the the Generate Ratings (Uploaded URL) tab (in Side Navigation menu) under "Our Data" section is working	Amir	Pass
7.2.6		Check if the implemented tooltips work in Organization user account	Amir	Pass
7.2.7		Check if hyperlinks for how to get User ID and Product ID works Organization user account	Amir	Pass
7.2.8		Leaving the User ID and Product ID field empty, try generating Rating Prediction	Amir	Pass
7.2.9		while both User ID and Product ID fields are empty, spam multiple fast clicks on "Generate" button	Amir	Pass
7.2.10		Leaving the User ID empty but Product ID field is filled, try generating Rating Prediction	Amir	Pass
7.2.11		Fill in the User ID and leave the Product ID field empty, then try to generate Rating Prediction	Amir	Pass
7.2.12		Enter both User ID and Product ID to generate Rating prediction	Amir	Pass
7.2.13		Try using User ID from other website (shopee)	Amir	Pass
7.2.14		Try Product ID from other websites (lazada)	Amir	Pass
7.2.15		Leaving the User ID and Product ID field empty, try generating Rating Prediction (Uploaded URL)	Amir	Pass
7.2.16		Leave the User ID empty but fill the Product ID field then try generating Rating Prediction (Uploaded URL)	Amir	Pass
7.2.17		Fill in the User ID and leave the Product ID field empty, then try to generate Rating Prediction (Uploaded URL)	Amir	Pass
7.2.18		Enter both User ID and Product ID to generate Rating prediction(Uploaded URL)	Amir	Pass

7.2.19		Enter User ID and Product ID then try generating predicted rating before uploading a data set into RECS	Amir	Pass
7.2.20		Leaving the User ID and Product ID field empty, try generating Rating Prediction (Your Data)	Amir	Pass
7.2.21		Leave the User ID empty but fill the Product ID field then try generating Rating Prediction (Your Data)	Amir	Pass
7.2.22		Fill in the User ID and leave the Product ID field empty, then try to generate Rating Prediction (Your Data)	Amir	Pass
7.2.23		Enter both User ID and Product ID to generate Rating prediction (Your Data)	Amir	Pass
7.2.24		After generating ratings go to the "Discover" page, checking to see "Recent Searches"	Amir	Pass
7.2.25		Go to "History" in side navigation menu and click on "Results" tab	Amir	Pass
7.2.26		Search for a particular result	Amir	Pass
7.2.27		In "Results" click sort by Result ID	Amir	Pass
7.2.28		In "Results" click sort by sort by Date Generated	Amir	Pass
7.2.29		In "Results" click sort by sort by Generated Type	Amir	Pass
7.2.30		In "Results" click results option	Amir	Pass
7.2.31		In "Results" click filter button "Ratings Prediction (Your Data)	Amir	Pass
7.2.32		In "Results" click "All" button	Amir	Pass
7.2.33		In "Results" click filter button "Ratings Prediction (RECS' Data)	Amir	Pass
7.2.34		On "Discover" Page, press "See All Results"	Amir	Pass

Recommendations:

ID	Case Category	Sub Case Description	Responsible tester	Status(Pass/Fail/not executed)
8	Recommendations		Amir	Pass
8.1.1		Test to see if the the Generate Recommendations tab (on Discover page) is working	Amir	Pass
8.1.2		Test to see if the the Generate Recommendations (RECS' Data) tab (in Side Navigation menu) under "Our Data" section is working	Amir	Pass
8.1.3		Test to see if the the Generate Recommendations (Uploaded URL) tab (in Side Navigation menu) under "Our Data" section is working	Amir	Pass
8.1.4		Check if the implemented tooltips work in Generate Recommendations (RECS' Data)	Amir	Pass
8.1.5		Check if hyperlinks for how to get Product ID works Generate Recommendations (RECS' Data)	Amir	Pass
8.1.6		Check if the implemented tooltips work in Generate Recommendations (Uploaded URL)	Amir	Pass
8.1.7		Check if hyperlinks for how to get Product ID works Generate Recommendations (Uploaded URL)	Amir	Pass
8.1.8		Leaving the Product ID field empty, try generating Recommendation (RECS' Data)	Amir	Pass
8.1.9		With Product ID field empty, spam multiple fast clicks on "Generate" button (RECS' Data)	Amir	Pass
8.1.10		Enter Product ID to generate Recommendations (RECS' Data)	Amir	Pass
8.1.11		Try Product ID from other websites (RECS' Data)	Amir	Pass
8.1.12		Leaving the Product ID field empty, try generating Recommendation (Uploaded URL)	Amir	Pass
8.1.13		With Product ID field empty, spam multiple fast clicks on "Generate" button (Uploaded URL)	Amir	Pass
8.1.14		Enter Product ID to generate Recommendations (Uploaded URL)	Amir	Pass
8.1.15		Try Product ID from other websites (Uploaded URL)	Amir	Pass
8.1.16		After generating Recommendations go to the "Discover" page, checking to see "Recent Searches"	Amir	Pass
8.1.17		On "Discover" Page, click "See All Results"	Amir	Pass
8.1.18		Go to "History" in side navigation menu and click on "Results" tab	Amir	Pass
8.1.19		Search for a particular result	Amir	Pass
8.1.20		In "Results" click sort by Result ID	Amir	Pass
8.1.21		In "Results" click sort by sort by Date Generated	Amir	Pass
8.1.22		In "Results" click sort by sort by Generated Type	Amir	Pass
8.1.23		In "Results" click results option	Amir	Pass
8.1.24		In "Results" click filter button " Recommendations Prediction (Your Data)	Amir	Pass
8.1.25		In "Results" click filter button " Recommendations Prediction (RECS' Data)	Amir	Pass
8.2.1		Test to see if the the Generate Recommendations tab (on Discover page -> "Your Data")	Amir	Pass
8.2.2		Test to see if the the Generate Recommendations tab (on Discover page -> "Our Data")	Amir	Pass
8.2.3		Test to see if the the Generate Recommendations tab (in Side Navigation menu) under "Your Data" section is working	Amir	Pass
8.2.4		Test to see if the the Generate Recommendations tab (in Side Navigation menu) under "Our Data" is working	Amir	Pass
8.2.5		Test to see if the the Generate Recommendations (Uploaded URL) tab (in Side Navigation menu) under "Our Data" section is working	Amir	Pass
8.2.6		Check if the implemented tooltips work in Organization user account (RECS' Data)	Amir	Pass
8.2.7		Check if the implemented tooltips work in Organization user account (Uploaded URL)	Amir	Pass
8.2.8		Check if the implemented tooltips work in Organization user account (Your Data)	Amir	Pass
8.2.9		Leaving the Product ID field empty, try generating Recommendation (RECS' Data)	Amir	Pass
8.2.10		With Product ID field empty, spam multiple fast clicks on "Generate" button	Amir	Pass
8.2.11		Enter Product ID to generate Recommendations	Amir	Pass
8.2.12		Try Product ID from other websites	Amir	Pass
8.2.13		Leaving the Product ID field empty, try generating Recommendation (Uploaded URL)	Amir	Pass
8.2.14		With Product ID field empty, spam multiple fast clicks on "Generate" button (Uploaded URL)	Amir	Pass
8.2.15		Enter Product ID to generate Recommendations (Uploaded URL)	Amir	Pass
8.2.16		Try Product ID from other websites (Uploaded URL)	Amir	Pass
8.2.17		Leaving the Product ID field empty, try generating Recommendation (Your Data)	Amir	Pass
8.2.18		With Product ID field empty, spam multiple fast clicks on "Generate" button (Your Data)	Amir	Pass
8.2.19		Enter Product ID to generate Recommendations (Your Data)	Amir	Pass
8.2.20		Try Product ID from other websites (Your Data)	Amir	Pass
8.2.21		After generating Recommendations go to the "Discover" page, checking to see "Recent Searches"	Amir	Pass
8.2.22		Go to "History" in side navigation menu and click on "Results" tab	Amir	Pass
8.2.23		Search for a particular result	Amir	Pass
8.2.24		In "Results" click sort by Result ID	Amir	Pass
8.2.25		In "Results" click sort by sort by Date Generated	Amir	Pass
8.2.26		In "Results" click sort by sort by Generated Type	Amir	Pass
8.2.27		In "Results" click results option	Amir	Pass
8.2.28		In "Results" click filter button " Recommendations Prediction (Your Data)	Amir	Pass
8.2.29		In "Results" click filter button " Recommendations Prediction (RECS' Data)	Amir	Pass

Test Status Report

ID	Case Category	Responsible tester	Tested in Sprint	Status(Pass/Fail/not executed)
1	Account Management	Joon Wee	6	Pass
2	Security	Amir	7	Pass
3	Trial Feature	Joon Wee	7	Pass
4	Website Introduction	Joon Wee	7	Pass
5	Webscraping	Joon Wee	8	Pass
6	Data Set	Amir	7	Pass
7	Ratings Prediction	Amir	7	Pass
8	Recommendations	Amir	8	Pass

Sprint 6	14/04/2023 Start
Overall progress of the QA cycle(Ontime, delayed, Stopped)	**On time**
Total number of test cases	41
Number of testers	1
Test cycle duration	5 Days
Status for	
Number of test cases	41
Number of test cases executed	41
Number of failures encountered so far	5
Number of critical defects- still open	0
Overall status	
Number of test cases	41
Number of test cases executed	41
Number of test cases failed	0
Pass Percentage of the failure	< 5%
Failure Percentage	0.00%
Critical defects percentage	0%
Sprint 7	**19/04/2023 Start**
Overall progress of the QA cycle(Ontime, delayed, Stopped)	**Delayed**
Total number of test cases	144
Number of testers	2
Test cycle duration	15 Days
Status for	
Number of test cases	144
Number of test cases executed	144
Number of failures encountered so far	0
Number of critical defects- still open	0
Overall status	
Number of test cases	185
Number of test cases executed	185
Number of test cases failed	0
Pass Percentage of the failure	< 5%
Failure Percentage	0.00%
Critical defects percentage	0%

Sprint 8	03/05/2023 Start
Overall progress of the QA cycle(Ontime, delayed, Stopped)	On time
Total number of test cases	94
Number of testers	2
Test cycle duration	9 Days
Status for	
Number of test cases	93
Number of test cases executed	93
Number of failures encountered so far	0
Number of critical defects- still open	0
Overall status	
Number of test cases	279
Number of test cases executed	279
Number of test cases failed	0
Pass Percentage of the failure	< 5%
Failure Percentage	0.00%
Critical defects percentage	0%
Overall Summary	
All Test Cases Passed	