Test Cases

Black Box Testing **1**
Equivalence Class and Boundary Value Testing 1
White Box Testing **3**
Update User Details 3
Control Flow Diagram 3
Basis Path Testing 4
Fetching Carpark Availability Data 5
Control Flow Diagram 5
Basis Path Testing 6

Black Box Testing

Equivalence Class and Boundary Value Testing

For Black Box Testing, we will be testing the SignUpController control class. SignUpController is responsible for receiving new user details from the SignUpForm, then creating a user and saving it in the database. It checks that the username and email submitted are not already associated with any account in the database. It also ensures that the email is in the correct format, and the password must be at least 8 characters long with a combination of uppercase letters, lowercase letters, numbers and symbols. Email, Username and Password are all discrete values, so they have 1 valid and 1 invalid equivalence class (EC).

Email:
Valid EC: Email with correct format and not already in use.
Invalid EC: Email with incorrect format and/or already in use.

Username:
Valid EC: Username is not already in use.
Invalid EC: Username is already in use.

Password:
Requirements list (tick if fulfilled, cross if not):

1. Password must be at least 8 characters long.
2. Password must include at least one special character.
3. Password must contain at least one number.
4. Password must have at least one uppercase letter.
5. Password must have at least one lowercase letter.

Valid EC: At least 8 characters long with a combination of uppercase letters, lowercase letters, numbers and symbols (requirements 1-5 fulfilled).
Invalid EC: Less than 8 characters long and/or does not contain at least 1 uppercase letter, 1 lowercase letter, 1 number and 1 symbol (at least one of the requirements is not fulfilled).

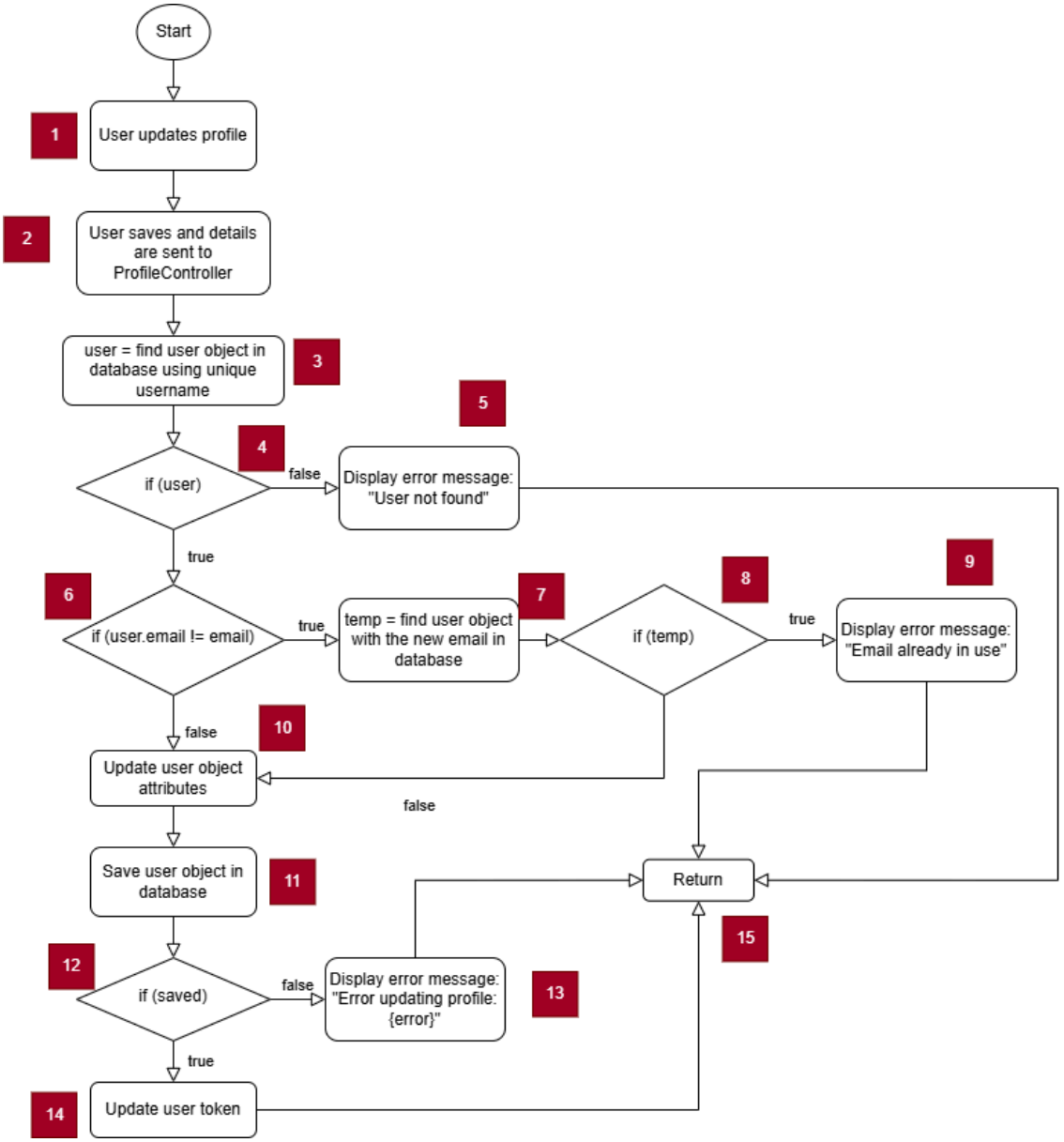
Test Input			Expected Output	Actual Output
Email	Username	Password		
newemail@gmail.com	newuser	Testing@123	No error, redirect to homepage	No error, redirect to homepage
newemail	newuser	Testing@123	Error message: “Please include an ‘@’ in the email address.’newuser’ is missing an ‘@’.”	Error message: “Please include an ‘@’ in the email address.’newuser’ is missing an ‘@’.”
existingemail@gmail.com	newuser	Testing@123	Error message: “Email already in use”	Error message: “Email already in use”
newemail@gmail.com	existinguser	Testing@123	Error message: “Username already exists”	Error message: “Username already exists”
newemail@gmail.com	newuser	Tes@123	Only requirement 1 (8 characters) is not fulfilled. Error message: “Please match the requested format.”	Only requirement 1 (8 characters) is not fulfilled. Error message: “Please match the requested format.”
newemail@gmail.com	newuser	Testing123	Only requirement 2 (1 special character) is not fulfilled. Error message: “Please match the requested format.”	Only requirement 2 (1 special character) is not fulfilled. Error message: “Please match the requested format.”
newemail@gmail.com	newuser	Testing@	Only requirement 3 (1 number) is not fulfilled. Error message: “Please match the requested format.”	Only requirement 3 (1 number) is not fulfilled. Error message: “Please match the requested format.”
newemail@gmail.com	newuser	testing@123	Only requirement 4 (1 uppercase letter) is not fulfilled. Error message: “Please match the requested format.”	Only requirement 4 (1 uppercase letter) is not fulfilled. Error message: “Please match the requested format.”
newemail@gmail.com	newuser	TESTING@123	Only requirement 5 (1 lowercase letter) is not fulfilled. Error message: “Please match the requested format.”	Only requirement 5 (1 lowercase letter) is not fulfilled. Error message: “Please match the requested format.”

White Box Testing

Update User Details

Since the methods for updating user details in the controller and view are separate, we'll only do the control flow for the controller class. Users are allowed to modify their name, email and car plate number but not their username since the username is a unique key used to identify users in the database. Error checking is done in the view class and the inputs received are in the correct format.

Control Flow Diagram



Basis Path Testing

Cyclomatic Complexity
= |decisionpoint| + 1 = 4 + 1 = 5 (since all decision points are binary)

Basis Paths

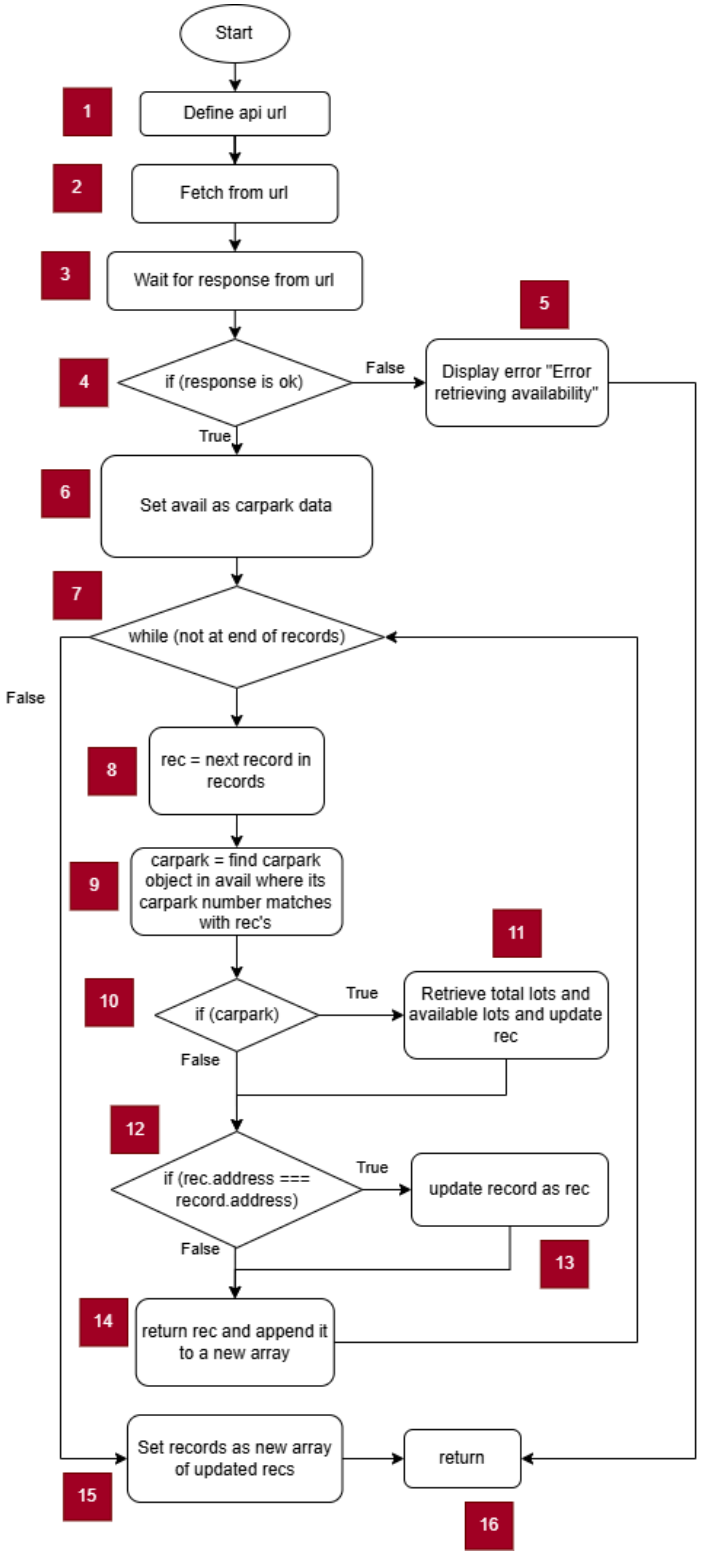
Basis Path no.	Basis Path	Description
1 (Baseline)	1, 2, 3, 4, 6, 10, 11, 12, 14, 15	Normal update flow (no email change, save successful)
2	1, 2, 3, 4, 5, 15	User not found in Database
3	1, 2, 3, 4, 6, 7, 8, 9, 15	New email already taken
4	1, 2, 3, 4, 6, 7, 8, 10, 11, 12, 14, 15	Email change successful
5	1, 2, 3, 4, 6, 10, 11, 12, 13, 15	Save failed due to DB error

Basis path	Test Input	Expected Output	Actual Output
1	Email: sameemail@gmail.com Name: New Name Car Plate No.: ABC-1234	User updated	User updated
2	Email: sameemail@gmail.com Name: New Name Car Plate No.: ABC-1234 Setup: User record deleted from DB before test. Input fields remain the same.	“User not found”.	“User not found”.
3	Email: exisitngemaill@gmail.com Name: New Name Car Plate No.: ABC-1234	“Email already in use”	“Email already in use”
4	Email: newemaill@gmail.com Name: New Name Car Plate No.: ABC-1234	User updated	User updated
5	Email: sameemail@gmail.com Name: New Name Car Plate No.: ABC-1234 Setup: Turn database offline at node 6	“Error updating profile: {error}”	“Error updating profile: {error}”

Fetching Carpark Availability Data

“records” is the array of carpark objects from HDB carpark information api. “record” is the carpark the user is currently viewing. “avail” is the carpark availability data fetched from the external api.

Control Flow Diagram



Basis Path Testing

Cyclomatic Complexity
= |decisionpoint| + 1 = 4 + 1 = 5 (since all decision points are binary)
Basis Paths

Basis Path no.	Basis Path	Description
1 (Baseline)	1, 2, 3, 4, 6, 7, 8, 9, 10, 11, 12, 13, 14, 7, 15, 16	Normal update flow (successful fetch, match found, updates made to records array and current record being accessed)
2	1, 2, 3, 4, 5, 16	Error receiving response from API.
3	1, 2, 3, 4, 6, 7, 15, 16	The records array is empty.
4	1, 2, 3, 4, 6, 7, 8, 9, 10, 12, 13, 14, 7, 15, 16	Carpark not found in availability data.
5	1, 2, 3, 4, 6, 7, 8, 9, 10, 11, 12, 14, 15, 16	The address of the record being updated (rec in records) is not the same as the current record being accessed / viewed by user (record).

Basis path	Test Input	Expected Output	Actual Output
1	url = api_url records = [carpark1] record = carpark1	carpark1 in records and record has total lots and available lots information added	carpark1 in records and record has total lots and available lots information added
2	url = “” records = [carpark1] record = carpark1	“Error retrieving availability”	“Error retrieving availability”
3	url = api_url records = [] record = carpark1	“No records to update”	“No records to update”
4	url = custom_url where response = [] records = [carpark1] record = carpark1	carpark1 not updated	carpark1 not updated
5	url = api_url records = [carpark1] record = carpark2	carpark1 in records updated, record (carpark2) not updated	carpark1 in records updated, record (carpark2) not updated