Software Requirement Specification

Group members	Dylan Burton
	George Williams
	Bobby Khleang
	Sarah Ross
	Rin Pereira
	Omaid Ghafoori

1. Project Description

A healthcare tracker where the user provides information like weight, height, and age in order to create a health plan with features like sleep tracker, weight tracker, medication/supplement reminders.

2. Functional Requirements

FR01	The software shall allow the user to register with an account (email, username, password)
FR02	The software must enable user to input and store medical information (weight, height, age)
FR03	The software must enable user to input health goals (weight loss/gain, sleep, steps)
FR04	The software shall provide users with reminders for scheduled medication.
FR05	The software could track users' sleep patterns using manual input or synced device data.
FR06	The software shall allow users to update their personal and health information
FR07	The software shall store and visualize user progress over time
FR08	The software shall validate user inputs to ensure all data is reasonable
FR09	
FRN	

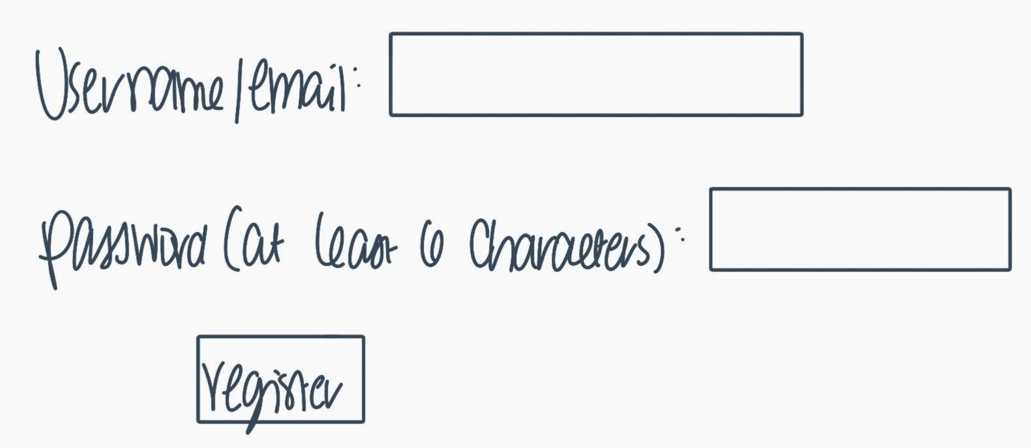
3. Non-Functional Requirements

NFR01	The software's response time to user actions must be within 5 seconds [performance]
NFR02	The software has security and encrypts users' information. [security]
NFR03	The software can be ported to different devices, specifically Android and web browser usage. (portability)
NFR04	If the software is run online, the application has to be available 90% of the time [availability].
NFR05	The system shall handle at least 50 concurrent users without performance degradation
NFR06	The system shall have a clean interface that is easy to navigate
FRN	

4. Use Case Specification

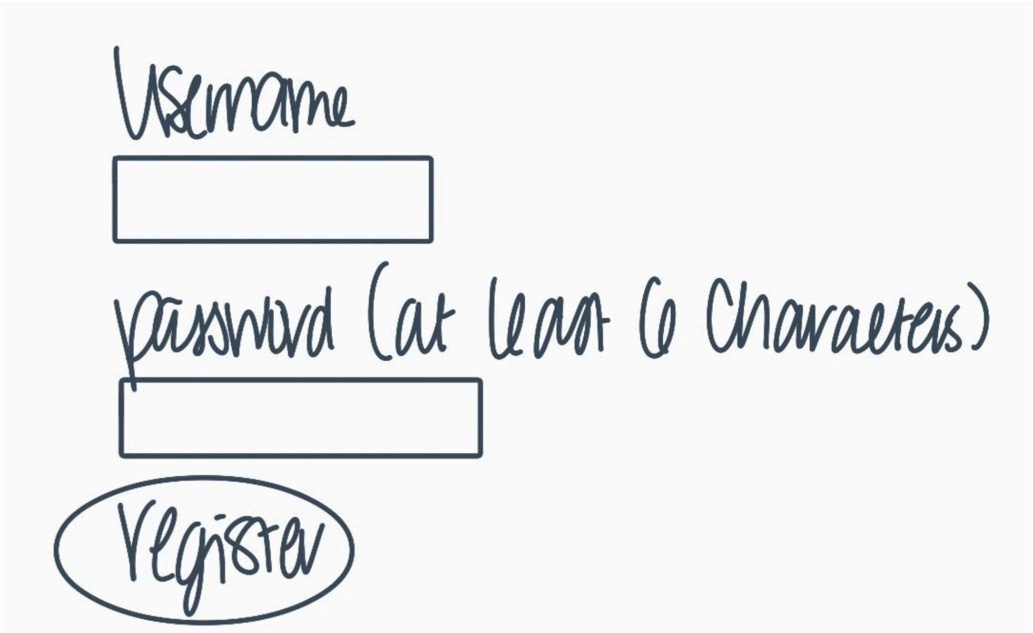
*<< Select **three** functional requirements and describe them in detail using use cases.>>*

UC01 Name:	Register a user
Description:	Allows a user to register using an email, username, and password.
Actor:	New user
Entry condition:	Registration page
Basic path:	1. The system presents a screen containing a. ‘Register’ button 2. The application shows a registration form screen containing a. Email/username (editable) b. Password (editable) 3. Actor clicks ‘Register’ button **[E01]** 4. The application verifies user data **[BR01][BR02][A01][A02][A03]** 5. The application stores username/email, password. 6. The application displays text confirming registration.
Alternative paths:	**[A01]** 1. The user enters an email/username that already exists. 2. The application displays “This username/email is already in use.” 3. The use case returns to step 3 of the basic path. **[A02]** 1. The user enters an email that is invalid. 2. The application displays “This email doesn’t work! Try another one.” 3. The use case returns to step 3 of the basic path. **[A03]** 1. The user enters a password that isn’t long enough. 2. Application displays ‘Passwords must be between # and # characters.’

<table>
<tr><td></td><td>3. The use case returns to step 3 of the basic path.</td></tr>
<tr><td><b>Exception paths:</b></td><td><b>[E01]</b>
1. Server connection fails → displays ‘Registration failed. Try again.’
2. The use case returns to step 3 of the basic path.</td></tr>
<tr><td><b>Business Rules:</b></td><td><b>[BR01]</b> . Password must be at least 6 characters
<b>[BR02]</b> . Email must be [email]@[extension].com/org</td></tr>
<tr><td><b>Data description</b></td><td>
<table>
<tr><th>Name</th><th>Type</th><th>Length</th><th>Mask</th><th></th></tr>
<tr><td>email</td><td>string</td><td>min: 7,
max:50</td><td>-</td><td></td></tr>
<tr><td>username</td><td>string</td><td>min: 4
max: 16</td><td>-</td><td></td></tr>
<tr><td>password</td><td>string</td><td>min: 6
max: 30</td><td>xxxxxx</td><td></td></tr>
<tr><td></td><td></td><td></td><td></td><td></td></tr>
</table>
</td></tr>
<tr><td><b>Prototype:</b></td><td><b>[PRO01]</b>


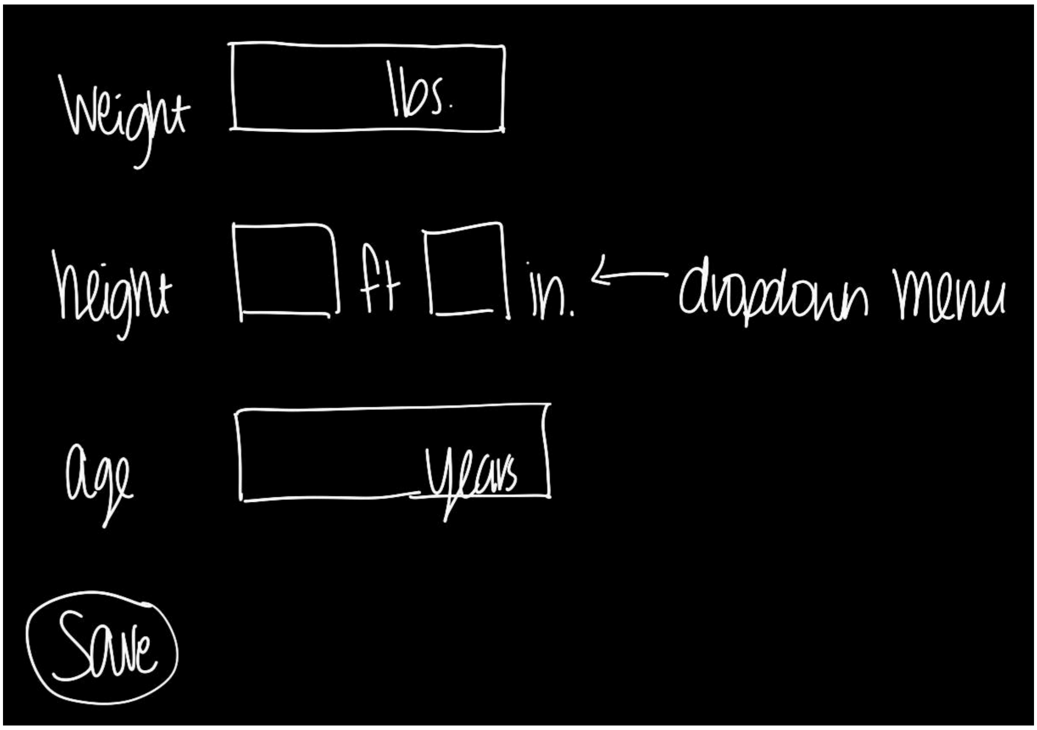
Username/email:
Password (at least 6 characters):
register

<b>[PRO02]</b></td></tr>
</table>

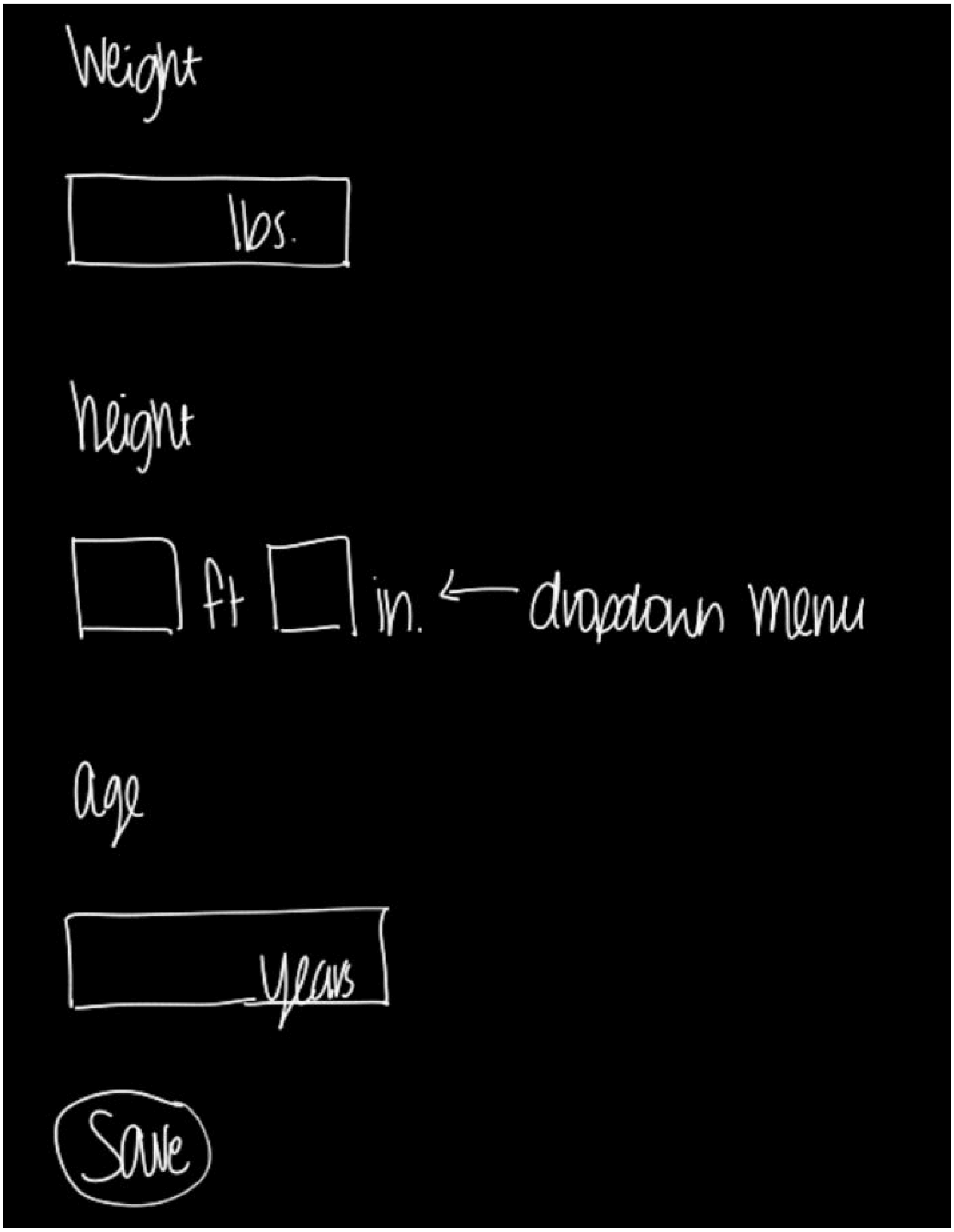
	 Username password (at least 6 Characters) Register

UC02 Name:	Input medical information
Description:	The software must enable user to input and store medical information (weight, height, age)
Actor:	Registered user
Entry condition:	User information page
Basic path:	1. The system displays the "Medical Information" screen after the registration page 2. Application shows fields with weight, height, and age forms 3. User can enter weight, height, and age 4. User clicks 'save' button **[E01]** 5. Application verifies user data **[A01][A02][A03][BR01][BR02]** 6. Application stores weight, height, and age 7. Applications displays text confirming saved user data.
Alternative paths:	**[A01]** 1. The user doesn't fill out one of the fields. 2. The application displays '[Weight, Height, Age] cannot be blank!' 3. The use case returns to step 2 of the basic path. **[A02]** 1. The user enters an invalid weight - ie. basically just 0 2. The application displays 'Invalid weight.' 3. The use case returns to step 2 of the basic path. **[A03]** 1. The user enters an invalid age - ie. basically just 0 2. The application displays 'Invalid age.' 3. The use case returns to step 2 of the basic path.
Exception paths:	**[E01]** 3. Data not saved due to server issues → display 'Save failed.' 4. The use case returns to step 2 of the basic path.

<table>
<tr><td><b>Business Rules:</b></td><td><b>[BR01]</b> weight must be at least > 0
<b>[BR02]</b> age must be > 0 and < 150 [unrealistic age]</td></tr>
<tr><td><b>Data description</b></td><td>
<table>
<tr><th>Name</th><th>Type</th><th>Length</th><th>Mask</th><th></th></tr>
<tr><td>Weight</td><td>double/float</td><td>2-3 #’s</td><td>-</td><td></td></tr>
<tr><td>Height</td><td>two ints [foot, inch]</td><td>singular number [ie. 5] & double number [ie. 5 inch or 10 inch]</td><td>-</td><td></td></tr>
<tr><td>Age</td><td>int</td><td>2 digits</td><td>-</td><td></td></tr>
<tr><td></td><td></td><td></td><td></td><td></td></tr>
</table>
</td></tr>
<tr><td><b>Prototype:</b></td><td><b>[PRO01]</b>



<b>[PRO02]</b></td></tr>
</table>

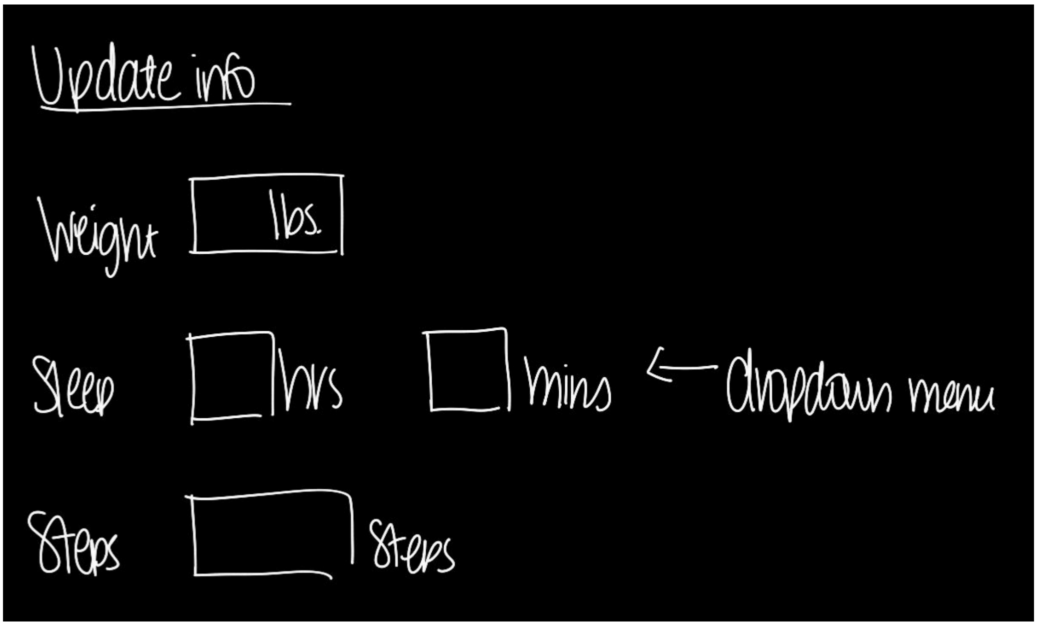
Weight
lbs.
height
ft
in.
dropdown menu
age
years
Save

UC03 Name:	Health Goals
Description:	The software must enable user to input health goals [weight loss/gain, sleep, steps]
Actor:	User who has gone through registration and basic info. menus
Entry condition:	Health goals page
Basic path:	1. The system presents the "Health Goals" page containing a. Set Goals button 2. The actor can input updated weight, input sleep [hopefully a menu and not input, easier for the code], input steps 3. The actor clicks 'save' **[E01]** 4. The application verifies inputs **[BR01][BR02][A01][A02]** 5. The application saves the new input.
Alternative paths:	**[A01]** 1. The actor doesn't fill out one of the fields. 2. The application displays '[Weight, Height, Age] cannot be blank!' [field!'] 3. The use case returns to step 2. **[A02]** 1. The actor enters a invalid weight - ie. basically just 0 2. The application displays 'impossible weight' 3. The use case returns to step 2.
Exception paths:	**[E01]** 5. Server sync fails → display 'Failed to update goals.' 6. The use case returns to step 2.
Business Rules:	**[BR01]** weight must be at least > 0 **[BR02]** sleep > 0 hr 0 min to be valid to be tracked
Data description	

Name	**Type**	**Length**	**Mask**	

weight loss/gain	double/float	< 100	-	
sleep	2 int [hr + min]	hr is one digit, min is typically 2 digits	-	
steps	int	accommodate for thousands	-	

Prototype: **[PRO01]**


Update info
Weight
lbs.
Sleep
hrs
mins
← dropdown menu
Steps
steps