

SE UNIT 4 ASSIGNMENT

TEAM 9

SUMANTH.C(PES1UG20CS711)

ROHAN T.P (PES1UG20CS682)

VINOD KUMAR E(PES1UG20CS702)

NIRANJAN T S (PES1UG20CS709)

Problem Statement – 1: Unit Testing

A unit is the smallest block of code that functions individually. The first

level of testing is Unit testing and this problem statement is geared towards the same.

- Discuss with your teammates and demarcate units in your code base

- o Note: discuss why the code snippet you have chosen can be classified as a unit

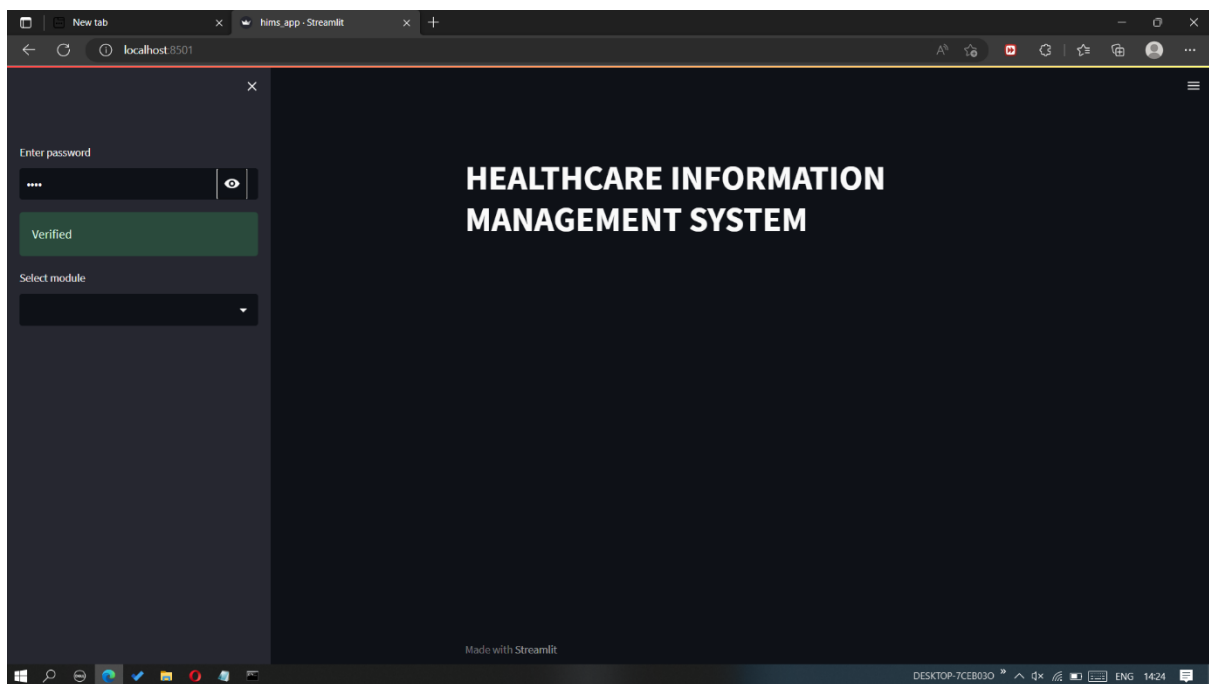
- Develop test cases for both valid and invalid data

- Ideate how you could further modularize larger blocks of code into compact units with your teammates

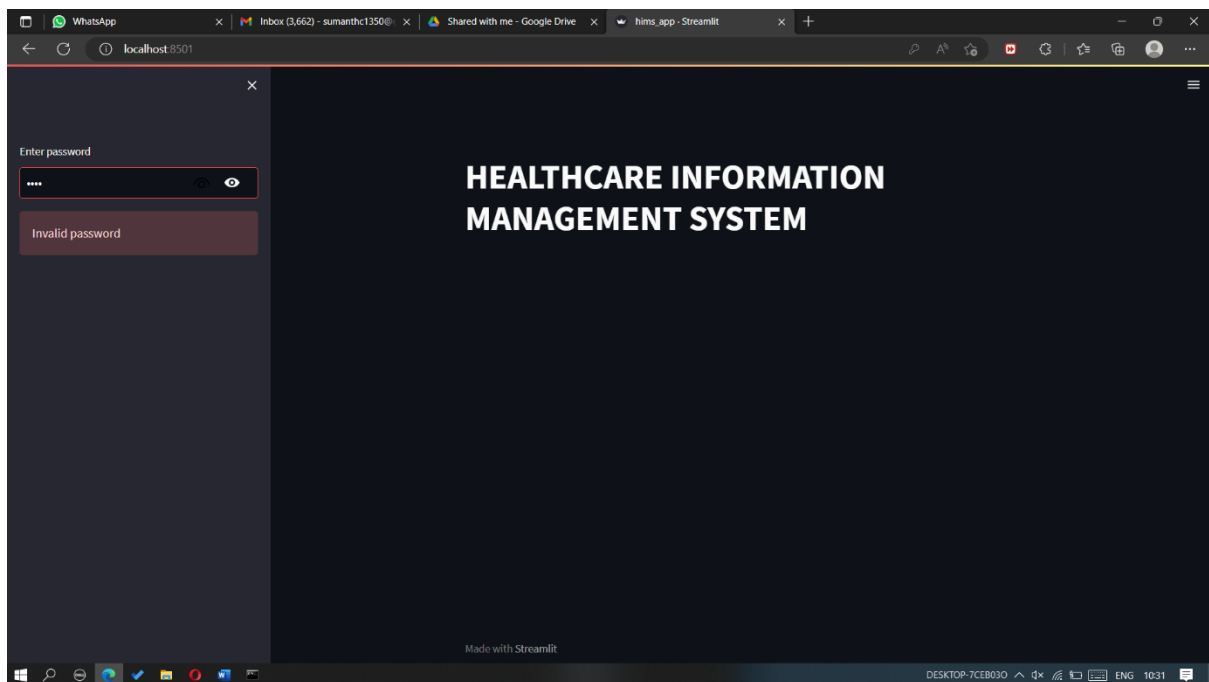
OUTPUT:

- 1.

Valid case:

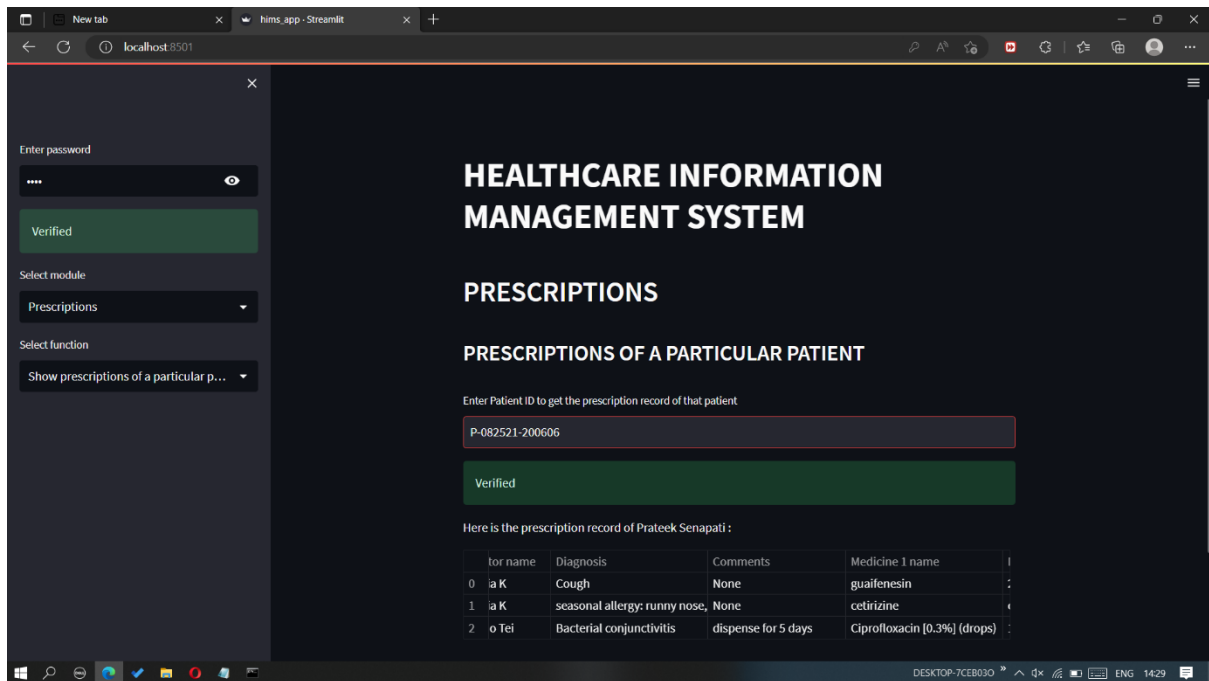


Invalid case:

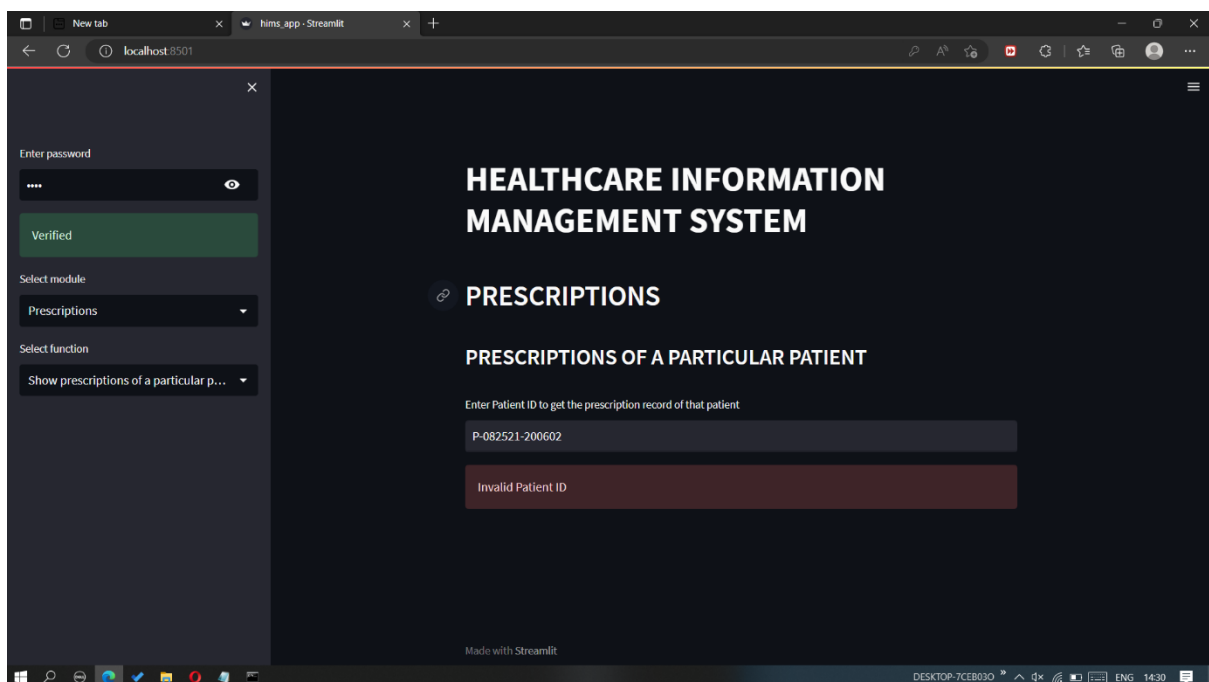


2.

Valid case:



Invalid case:



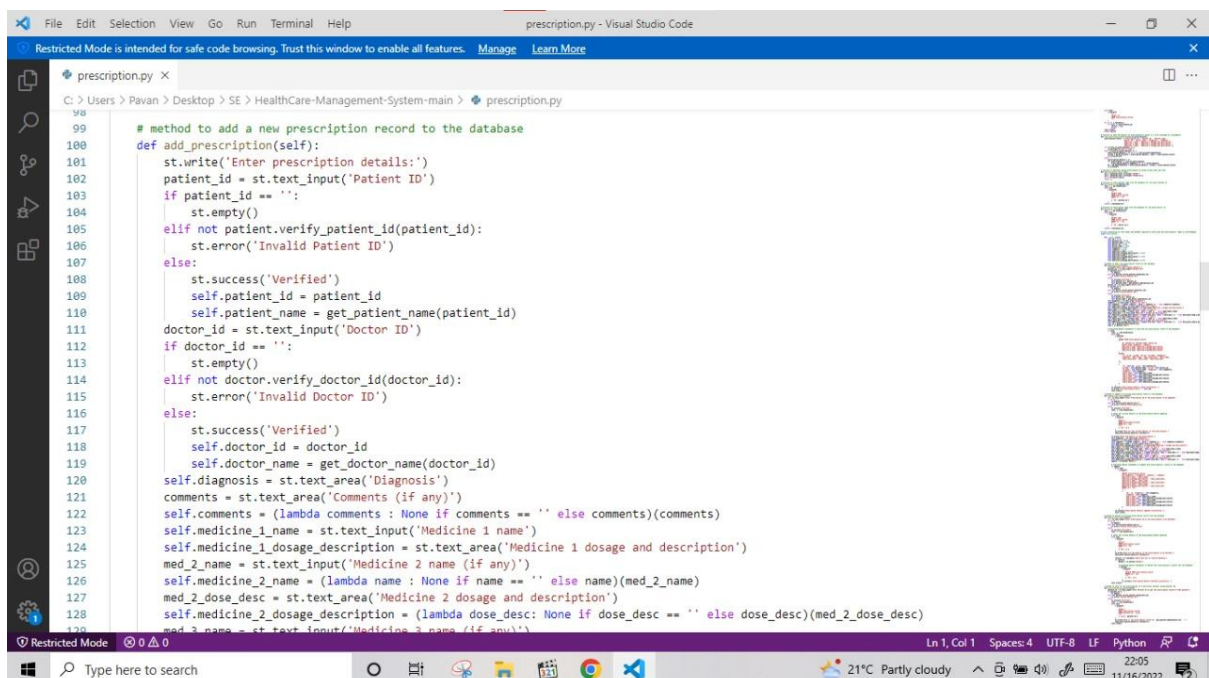
Problem Statement – 3: Static Testing

Static testing involves validating your code without any execution.

Under this problem statement, you will be expected to analyse and calculate the cyclomatic complexity of your code.

- Using the unit you selected in the first problem statement as an example, develop the control flow graph of your problem statement.
- Using the Control flow graph, calculate the cyclomatic complexity of your code.
- Using the cyclomatic complexity as an indicator, Ideate and code your unit again to reduce complexity

OUTPUT:



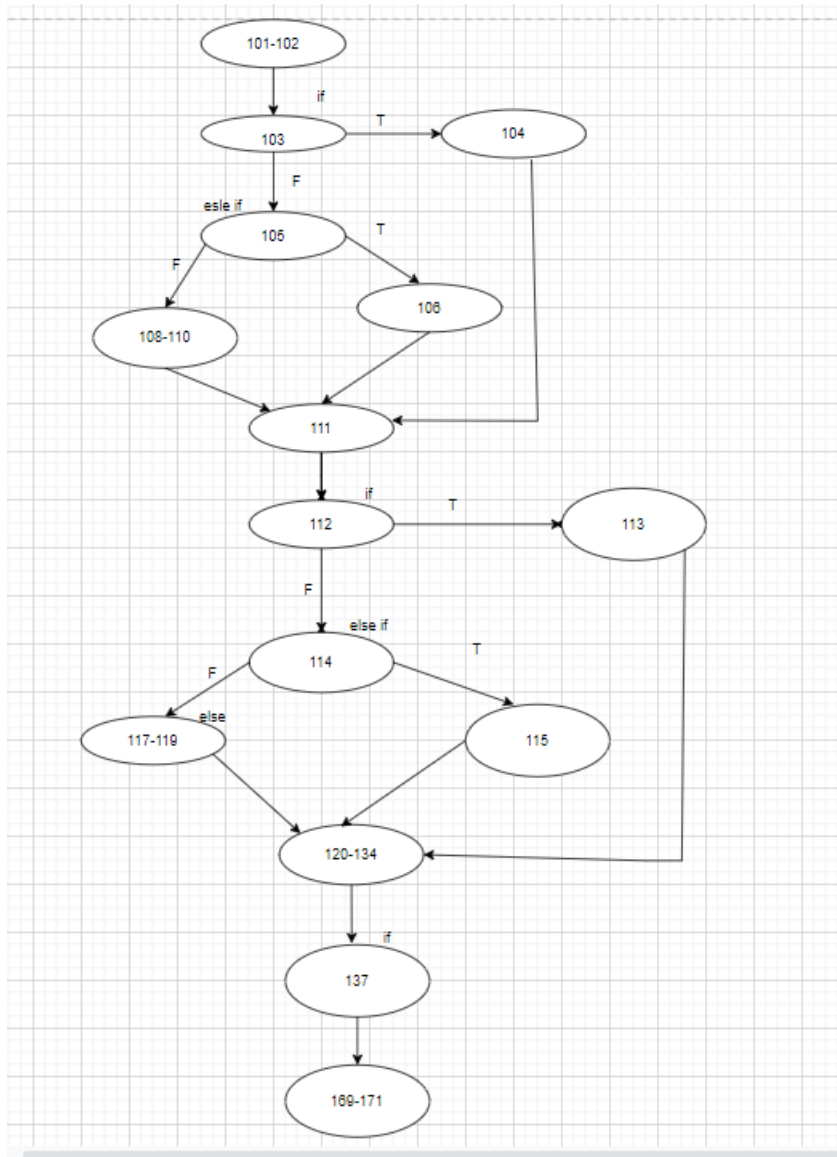
```
99 # method to add a new prescription record to the database
100 def add_prescription(self):
101     st.write('Enter prescription details:')
102     patient_id = st.text_input('Patient ID')
103     if patient_id == '':
104         st.empty()
105     elif not patient.verify_patient_id(patient_id):
106         st.error('Invalid Patient ID')
107     else:
108         st.success('Verified')
109         self.patient_id = patient_id
110         self.patient_name = get_patient_name(patient_id)
111     doctor_id = st.text_input('Doctor ID')
112     if doctor_id == '':
113         st.empty()
114     elif not doctor.verify_doctor_id(doctor_id):
115         st.error('Invalid Doctor ID')
116     else:
117         st.success('Verified')
118         self.doctor_id = doctor_id
119         self.doctor_name = get_doctor_name(doctor_id)
120     self.diagnosis = st.text_area('Diagnosis')
121     comments = st.text_area('Comments (if any)')
122     self.comments = (lambda comments : None if comments == '' else comments)(comments)
123     self.medicine_1_name = st.text_input('Medicine 1 name')
124     self.medicine_1_dosage_description = st.text_area('Medicine 1 dosage and description')
125     med_2_name = st.text_input('Medicine 2 name (if any)')
126     self.medicine_2_name = (lambda name : None if name == '' else name)(med_2_name)
127     med_2_dose_desc = st.text_area('Medicine 2 dosage and description')
128     self.medicine_2_dosage_description = (lambda dose_desc : None if dose_desc == '' else dose_desc)(med_2_dose_desc)
129     med_3_name = st.text_input('Medicine 3 name (if any)')
```

```
File Edit Selection View Go Run Terminal Help prescription.py - Visual Studio Code
Restricted Mode is intended for safe code browsing. Trust this window to enable all features. Manage Learn More

prescription.py
C:\Users> Pavan > Desktop > SE > HealthCare-Management-System-main > prescription.py
122 self.comments = (lambda comments : None if comments == '' else comments)(comments)
123 self.medicine_1_name = st.text_input('Medicine 1 name')
124 self.medicine_1_dosage_description = st.text_area('Medicine 1 dosage and description')
125 med_2_name = st.text_input('Medicine 2 name (if any)')
126 self.medicine_2_name = (lambda name : None if name == '' else name)(med_2_name)
127 med_2_dose_desc = st.text_area('Medicine 2 dosage and description')
128 self.medicine_2_dosage_description = (lambda dose_desc: None if dose_desc == '' else dose_desc)(med_2_dose_desc)
129 med_3_name = st.text_input('Medicine 3 name (if any)')
130 self.medicine_3_name = (lambda name : None if name == '' else name)(med_3_name)
131 med_3_dose_desc = st.text_area('Medicine 3 dosage and description')
132 self.medicine_3_dosage_description = (lambda dose_desc: None if dose_desc == '' else dose_desc)(med_3_dose_desc)
133 self.id = generate_prescription_id()
134 save = st.button('Save')
135
136 # executing SQLite statements to save the new prescription record to the database
137 if save:
138     conn, c = db.connection()
139     with conn:
140         c.execute(
141             """
142             INSERT INTO prescription_record
143             (
144                 id, patient_id, patient_name, doctor_id,
145                 doctor_name, diagnosis, comments,
146                 medicine_1_name, medicine_1_dosage_description,
147                 medicine_2_name, medicine_2_dosage_description,
148                 medicine_3_name, medicine_3_dosage_description
149             )
150             VALUES (
151                 :id, :p_id, :p_name, :dr_id, :dr_name, :diagnosis,
152                 :comments, :med_1_name, :med_1_dose_desc, :med_2_name,
153                 :med_2_dose_desc, :med_3_name, :med_3_dose_desc
154             )
155             """
156         )
157         {
158             'id': self.id, 'p_id': self.patient_id,
159             'p_name': self.patient_name, 'dr_id': self.doctor_id,
160             'dr_name': self.doctor_name, 'diagnosis': self.diagnosis,
161             'comments': self.comments,
162             'med_1_name': self.medicine_1_name,
163             'med_1_dose_desc': self.medicine_1_dosage_description,
164             'med_2_name': self.medicine_2_name,
165             'med_2_dose_desc': self.medicine_2_dosage_description,
166             'med_3_name': self.medicine_3_name,
167             'med_3_dose_desc': self.medicine_3_dosage_description,
168         }
169         st.success('Prescription details saved successfully.')
170         st.write('The Prescription ID is: ', self.id)
171         conn.close()
172
173 # method to update an existing prescription record in the database
```

```
File Edit Selection View Go Run Terminal Help prescription.py - Visual Studio Code
Restricted Mode is intended for safe code browsing. Trust this window to enable all features. Manage Learn More

prescription.py
C:\Users> Pavan > Desktop > SE > HealthCare-Management-System-main > prescription.py
143 (
144     id, patient_id, patient_name, doctor_id,
145     doctor_name, diagnosis, comments,
146     medicine_1_name, medicine_1_dosage_description,
147     medicine_2_name, medicine_2_dosage_description,
148     medicine_3_name, medicine_3_dosage_description
149 )
150 VALUES (
151     :id, :p_id, :p_name, :dr_id, :dr_name, :diagnosis,
152     :comments, :med_1_name, :med_1_dose_desc, :med_2_name,
153     :med_2_dose_desc, :med_3_name, :med_3_dose_desc
154 );
155 """
156 {
157     'id': self.id, 'p_id': self.patient_id,
158     'p_name': self.patient_name, 'dr_id': self.doctor_id,
159     'dr_name': self.doctor_name, 'diagnosis': self.diagnosis,
160     'comments': self.comments,
161     'med_1_name': self.medicine_1_name,
162     'med_1_dose_desc': self.medicine_1_dosage_description,
163     'med_2_name': self.medicine_2_name,
164     'med_2_dose_desc': self.medicine_2_dosage_description,
165     'med_3_name': self.medicine_3_name,
166     'med_3_dose_desc': self.medicine_3_dosage_description,
167 }
168
169 st.success('Prescription details saved successfully.')
170 st.write('The Prescription ID is: ', self.id)
171 conn.close()
172
173 # method to update an existing prescription record in the database
```



CYCLOMETRIC COMPLEXITY= $e-n+2p$

=5.

The given function can be further modularised by breaking it into two functions ,which can reduce cyclomatic complexity.