THE UNIVERSITY OF WESTERN AUSTRALIA
School of Computer Science and Software Engineering

1st SEMESTER EXAMINATIONS 2019

CITS3403 AGILE WEB DEVELOPMENT

FAMILY NAME: _____ STUDENT ID: _____

GIVEN NAMES: _____ SIGNATURE: _____

This paper contains: **??** pages (including the title page)
Time allowed: 2 hours 10 minutes

This exam contains 10 questions worth 5 marks each. Candidates must attempt ALL questions. The questions should be answered in the space provided in this examination paper.

PLEASE NOTE

Examination candidates may only bring authorised materials into the examination room. If a supervisor finds, during the examination, that you have unauthorised material, in whatever form, in the vicinity of your desk or on your person, whether in the examination room or the toilets or en route to/from the toilets, the matter will be reported to the head of school and disciplinary action will normally be taken against you. This action may result in your being deprived of any credit for this examination or even, in some cases, for the whole unit. This will apply regardless of whether the material has been used at the time it is found.

Therefore, any candidate who has brought any unauthorised material whatsoever into the examination room should declare it to the supervisor immediately. Candidates who are uncertain whether any material is authorised should ask the supervisor for clarification.

This page is intentionally left blank

1. (5 marks)

Define the following terms and explain their specific role in exposing a web application to a user via a web browser.

- HTML

- CSS

- Javascript

- Cookies

- HTTP requests/responses

2. (5 marks)

Given the following HTML code describe the style that is applied to the table.

```
<html>
  <head>
    <meta charset='UTF-8'/>
    <style>
      table, td, th {
        border: 1px solid black;
      }

      .even {
        background-color: blue;
      }

      table td + td{
        font-style: italic;
      }

      #warning{
        background-color: red;
        font-weight:bold;
      }

      td:hover{
        border: 2px solid green;
      }
   </style>

  </head>
  <body>
    <table>
      <tr class='even'><th>A</th><th>B</th><th>C</th></tr>
      <tr><td>1</td><td>2</td><td>3</td></tr>
      <tr class='even'><td>i</td><td>ii</td><td id='warning'>iii</td></tr>
    </table>
  </body>
</html>
```

3. (5 marks)

Provide a javascript functions to transpose every table in a web page. For example, in Figure 1 we can see a table before and after it is transposed.

Mon	**Tue**	**Wed**	**Thur**	**Fri**
Bill	Joanne	David	Michelle	Agnes
Michael		Arthur	Janet	John

Mon	Bill	Michael
Tue	Joanne	
Wed	David	Arthur
Thur	Michelle	Janet
Fri	Agnes	John

Figure 1: A table before and after it is transposed.

Hint: HTML elements have an attribute called `children` *that contains a collection of the children of the element.*

4. (5 marks)

Explain the difference between sever side rendering of a web page with flask and client side rendering using a REST API, AJAX and jQuery. Describe the advantages and disadvantages of each approach.

5. (5 marks)

Suppose that you are writing a web application called *BucketList* where users can publish a list of things they would like to do before they die. For each user there is a list of things to do, with a checkbox indicating whether or not they have done it yet. Other users can see the list and provide a comment on the list items.

Propose a set of models to represent the data entities in this application. You do not have to present them fully in python, and you are not required to specify imports and all methods. However the types of fields and relations between the models should be clear.

6. (5 marks)

Given the *BucketList* application described in Question 5, sketch a wire-frame mock of a users page which shows their name, the items on their bucket list, and any comments other users have made.

Sketch a Jinja template to render the mock you have drawn. State any assumptions you require.

7. (5 marks)

Describe the directory structure of a typical Flask project, and highlight the roles of the main files.

8. (5 marks)

Give the six key architectural constraints of the REST software architecture, and give a short description of each constraint.

9. (5 marks)

Define the following terms in the context of software testing:

- Fixtures

- Assert

- Stub

- Selenium

- Coverage

10. (5 marks)

Describe the main activities of an agile web development process, in the context of a web development project. Particularly, highlight

(a) the different roles people have in the project,

(b) how the application requirements are gathered and maintained,

(c) how the project is divided into stages

(d) how source code is shared, and

(e) how the correct operation of the application is ensured.

EXTRA BLANK PAGE

EXTRA BLANK PAGE

EXTRA BLANK PAGE

End of Paper

Sample Solutions

1.

- Hypertext mark up language: for organising the content of a webpage
- Cascading Style Sheets: for describing the visual style of different elements of a web page, and organising how those styles are managed.
- Javascript is a programming language that is interpretted in the browser. it enables dynamic functionality and client side processing.
- Cookies are small packets of data sent to and stored in the browser. The web URL that created the cookie can use that cookie to remember users and manage security etc.
- HTTP requests responses are part of the hypertext transfer protocol. Requests are typically GET/POST depending on whether the browser is requesting data or sending data to the server. Responses have a code indicting if the request was successful, and if data was successfully requested will include the requested data.

2.

The table has

1. The Table has a black border
2. every even row has a blue background
3. The bottom right cell is red
4. when you hover over a cell it goes green
5. the text in every cell except the leftmost is italic.

3.

```
<html>
  <head>
    <meta charset='UTF-8'/>
```

```
    <title> Exam Test</title>

    <script>
    function transpose(tbody){
      var rows=tbody.children;
      var newrows=[];
      for(var i=0; i<rows.length; i++){
        data = rows[i].children;
        for(var j=0; j<data.length; j++){
          if(i==0)
            newrows[j]=document.createElement('TR');
          newrows[j].appendChild(data[j].cloneNode(true));
        }
      }
      while(rows.length>0)
        tbody.removeChild(rows[0]);
      console.log(tbody);
      for(i = 0; i<newrows.length; i++)
        tbody.appendChild(newrows[i]);
      console.log(tbody);
    }

    window.onload=function(){
      tables = document.getElementsByTagName('TBODY');
      for(var i = 0; i<tables.length; i++){
        transpose(tables[i]);
      }
    }
    </script>
  </head>
  <body>
    <h1>Table Test</h1>

    <table>
      <tr><th>Mon</th><th>Tue</th><th>Wed</th><th>Thur</th><th>Fri</th></tr>
      <tr><td>Bill</td><td>Joanne</td><td>David</td><td>Michelle</td><td>Agnes</td></tr>
      <tr><td>Michael</td><td> </td><td>Arthur</td><td>Janet</td><td>John</td></tr>
    </table>
  </body>
</html>
```

4.

Server side rendering will construct the web pages on a server, use templates (JINJA) and data from the

database. client side rendering will contruct the webpage on the client machine using jQuery and DOM manipulations. The REST API will take data from the database and send it as JSON to the client page.

Client side rendering uses less bandwidth, and puts less demand on the server. However the pages take longer to load initially as the code is larger.

5.

```
class User(db.Model):
  __tablename__='users'
  id=db.Column(db.Integer, primary_key=True)
  name=db.Column(String)
  pw_hash=db.Column(db.String)


class ListItem(db.Model):
  __tablename__='items'
  id=db.Column(db.integer, primary_key=true)
  description=db.Column(db.String)
  complete=db.Column(db.Boolean)
  owner=db.Column(db.Integer, db.ForiegnKey('users.id'))

class Comment(db.Model):
  __tablename__='comments'
  id=db.Column(db.integer, primary_key=true)
  comment=db.Column(db.String)
  item=db.Column(db.Integer, db.ForeignKey('items.id'))
  commentator=db.Column(db.Integer, db.ForiegnKey('users.id'))
```

6.

```
{%extends 'base.html'%}
  {%block content %}
  <H1>Hi, {{user.name}}</H2>

  <H2>Bucket List</H2>
  <ul>
  {% for item in items%}
```

```
 <li> {{item.description}}
    {% if item.completed%}
       Done!
    {%endif%}
    <ul>
    {% for comment in item.comments%}
      <li> {{comment.commentator}} says {{comment.comment}}</li>
    {% endfor %}
    </ul>
 </li>
 {% endfor %}
 </ul>
{%endblock%}
```

7.

- app

- routes.py

- virtual-environment directory

- `static` directory

- `app/templates/` directory

app is the module that runs the application it inherits from the Flask object routes matches URL requests with function to execute those requests virtual environment stores all the imports and environment variables for running the app static serves css, javascript and other files that don't change. app/templates contains the Jinja templates used for rendering

app.db app_name.py app models.py forms.py __init__.py controllers.py templates index.html routes.py venv all the package imports etc config.py

---

8.

Clientserver architecture The roles of the client and the server are distinguished clients can request information and the server will serve the requests

Statelessness The server retains no information about the previous requests from the clients

Cacheability Intermediatries can cache responses to common queries and serve those directly to clients

Layered system Provides abstraction so the client can not, and does not need to distinguish between the final server and an intermediary.

Code on demand (optional) Servers can temporarily extend or customise the functionality of a client by transferring executable code: for example, compiled components such as Java applets, or client-side scripts such as JavaScript.

Uniform interface The uniform interface constraint is fundamental to the design of any RESTful system. Resource identification in requests Resource manipulation through representations Self-descriptive messages Hypermedia as the engine of application state (HATEOAS)

9.

- Fixtures - Temporary Databases, doubles etc used in the testing process Create by SetUp and removed by TearDown
- Assert - A point in the test where the test can fail
- Stub - A replacement for code so a unit can be tested independently
- Selenium - A web based test environment
- Coverage - The amount of code that is executed by a test suite.

10.

People include product owner/manager/scrum master, developer, designer, and *client*

Requirements are maintained through user stories and wire-frame mocks

Project runs through iterations with each iteration delivering working verified code

Source is shared and tracked using versioning software such as GIT

The correct operation is ensured through testing.