Perl

Task 1: Find the number of friend circles.
Estimated Time: 1 day

We will use an
n*n
*n***n* **square matrix.** For example, cell [i,j] will hold value 1 if these users are friends. If not, the cell will hold the value 0. In the illustration below, there are two friend circles in the above example. Nick is only friends with Amy, but Amy is friends with Nick and Matt. This forms a friend circle. Mario makes another friend circle on his own.

	Nick	Amy	Matt	Mario
Nick	1	1	0	0
Amy	1	1	1	0
Matt	0	1	1	0
Mario	0	0	0	1

Task 2: Implement a search filter to find products in a given price range, category, brand.
Estimated Time: 1 day

You can have the list of products in a csv file or database.

Eg:- We can assume that the selected price range is low = 7 and high = 20, so our function solution should only return the product details with prices {9, 8, 14, 20, 17}.

Task 3: Create a module to handle the Cart items.
Estimated Time: 1 day

Module should have methods to

- Add new item to stock (Item name, unit price, tax % etc)
- add/delete item to cart
- Get the items in the cart
- Change the quantity for the selected item
- Calculate the total amount
- Calculate the tax

You can use the Database to store the cart items.

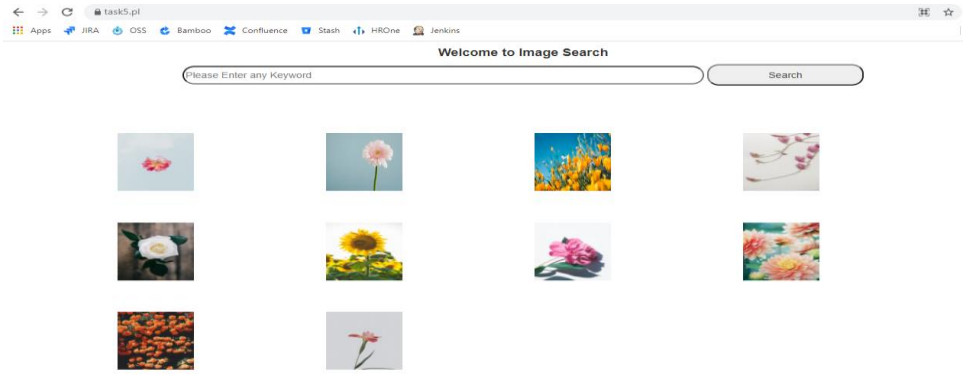
Task 4: Create a module to manage the playlist
Estimated Time: 1 day

Module should have methods to

- create new playlist
- Delete playlist
- Add songs to the playlist
- Delete songs from the playlist
- Get playlist
- Shuffle the playlist

Task 5: Create a module to search images(Image Gallery) using the third party API.
Estimated Time: 1 day
You can use the Unsplash's Image API .
Module should have the methods to

- Get the images by keywords



-

Task 6: Create a module to search recipes using the third party API
Estimated Time: 2 days
You can use the The MealDB API
Module should have the methods to

- List by name or first letter
- List ingredients for the selected recipe
- *List all Categories, Area, Ingredients*
- Filter by main ingredient
- Filter by Category
- Filter by Area

Task 7: Roman Numeral to Decimal numbers Converter
Estimated Time: 1 day

The numeric system represented by Roman numerals originated in ancient Rome and remained the usual way of writing numbers throughout Europe well into the Late Middle Ages. Roman numerals, as used today, employ seven symbols, each with a fixed integer value.

See the below table the *Symbol — Value* pairs:

- I — 1
- V — 5
- X — 10
- L — 50
- C — 100
- D — 500
- M — 1000

Accept the Roman number as input and print the decimal number.
If the invalid number is given, return the error message.

Task 8: A GitHub user search App
Estimated Time: 1 day

APIs allow you to use the real world data that drives platforms like GitHub. You can communicate with the remote servers and get data that you can use to build an app.

In this project you create a search app that uses GitHub API to retrieve user information when a valid username is input. It should display avatar, username, followers count, repository count, top 4 repositories based on forks and stars.

To get the data you need to communicate with GitHub API. you can either

- Read Docs
- Check API directly

Task 9: Create a Student Management System
Estimated Time: 2 days

You can also create a system for adding new students into an existing database, enrolling them in the existing courses, and generating unique student IDs. The project will also teach you how to create a system displaying the status of each student, including personal information, ID, fee balance, and courses which student is enrolled in.

Module should have methods to

- Add new student and return ID
- Get student info by ID
- Enroll courses
- Get courses enrolled by student
- Store the fee paid
- Get available courses
- Get course fee
- Get course details

Task 10: How many Sundays fell on the first of the month during the twentieth century (1 Jan 1901 to 31 Dec 2000)?

You are given the following information, but you may prefer to do some research for yourself.
Estimated Time: 1 day

- 1 Jan 1900 was a Monday.
- Thirty days has September,
  April, June and November.
  All the rest have thirty-one,
  Saving February alone,
  Which has twenty-eight, rain or shine.
  And on leap years, twenty-nine.
- A leap year occurs on any year evenly divisible by 4, but not on a century unless it is divisible by 400.

Task 11:
Create a module to handle Shapes like rectangle, square, circle, triangle etc. It should have the functions to get the input data, calculate the area and display the area.
use the OOPs concept.
Estimated Time: 1 day

Task 12:

https://documenter.getpostman.com/view/10808728/SzS8rjbc#auth-info-1e29ed29-066c-4494-ae5d-a6174a8fc551

Build a API using perl CGI to get the covid data for given date range and country or particular date. Build an interface in React to accept the date range and country. Use the API in the React application to display the data. (Authentication part can be ignored)
Estimated Time: 2 day

Task 13:
Given a chess board having N * N cells, you need to place N queens on the board in such a way that no queen attacks any other queen.
Estimated Time: 1 day

Input:
The only line of input consists of a single integer denoting N.
Constraints: N can be greater than 1 and less than or equal to 10
use the OOPs concept.

Example Input: N=4
OutPut:
0 1 0 0
0 0 0 1
1 0 0 0
0 0 1 0

React

Task 1: Calculator App.
Estimated Time: 1 day
Create a reactjs app to mimic calculator functions

Task 2 : Quiz App
Estimated Time: 2 days
Create a Quiz app for a set of questions either stored in a Constant file as json or try fetching from the database.
Should include the following feature:

- Ask users to press a button to start the quiz
- Every question should have four possible options as answer
- Next button should be visible when users answer the current question, and also should display a message whether its correct or wrong answer.
- If the user answers the question correctly, it should increase their score.
- At the end of the quiz, total score should be shown

Task 3 : Book Finder Application:
Estimated Time: 1 days
Develop an application for users to be able to find books easier by simply entering a query. For example, a query could be the author's name or the title of the book.
Should include the following feature:

- Firstly, an input field should be provided for users to be able to enter their queries for search.

- Design this application in a way to make users see the list of the books on screen
    - Example:

- On clicking on the particular book show following information
  description, authors, pages count, category, Publisher name, publishedDate, language, and preview link

Task 4: Image Gallery App

Estimated Time: 1 day

Build a React application to search the images . Use the code written for Perl task 5 as API.

Task 5: Recipe Search App

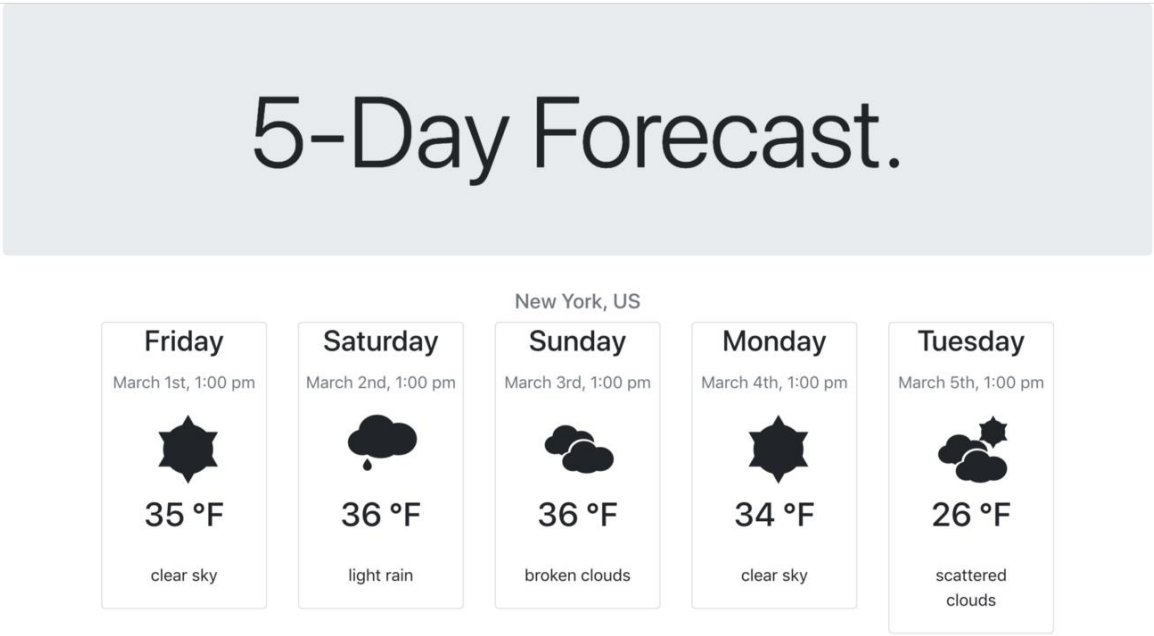
Estimated Time: 1 day

Build a React application to search the recipes . Use the code written for Perl task 6 as API.

Task 6: Weather App

Estimated Time: 2 day

The weather app must have all the basic functions, including city name, current weather icon, temperature, humidity, wind speed, etc. It must display the recording of both high and low temperatures of each day, including apt images for sunny/rainy/cloudy/snowy weather conditions. It should have a responsive design and refresh every five minutes with the exact temperature and weather conditions.
- Include the functionality wherein the user can click on a particular day of the week to see the hourly forecast.
- the default page should show the data for the current day
- use the react-router for displaying the data for the particular date.
- you can get the data from the api : https://openweathermap.org/
- you can accept the city name as input

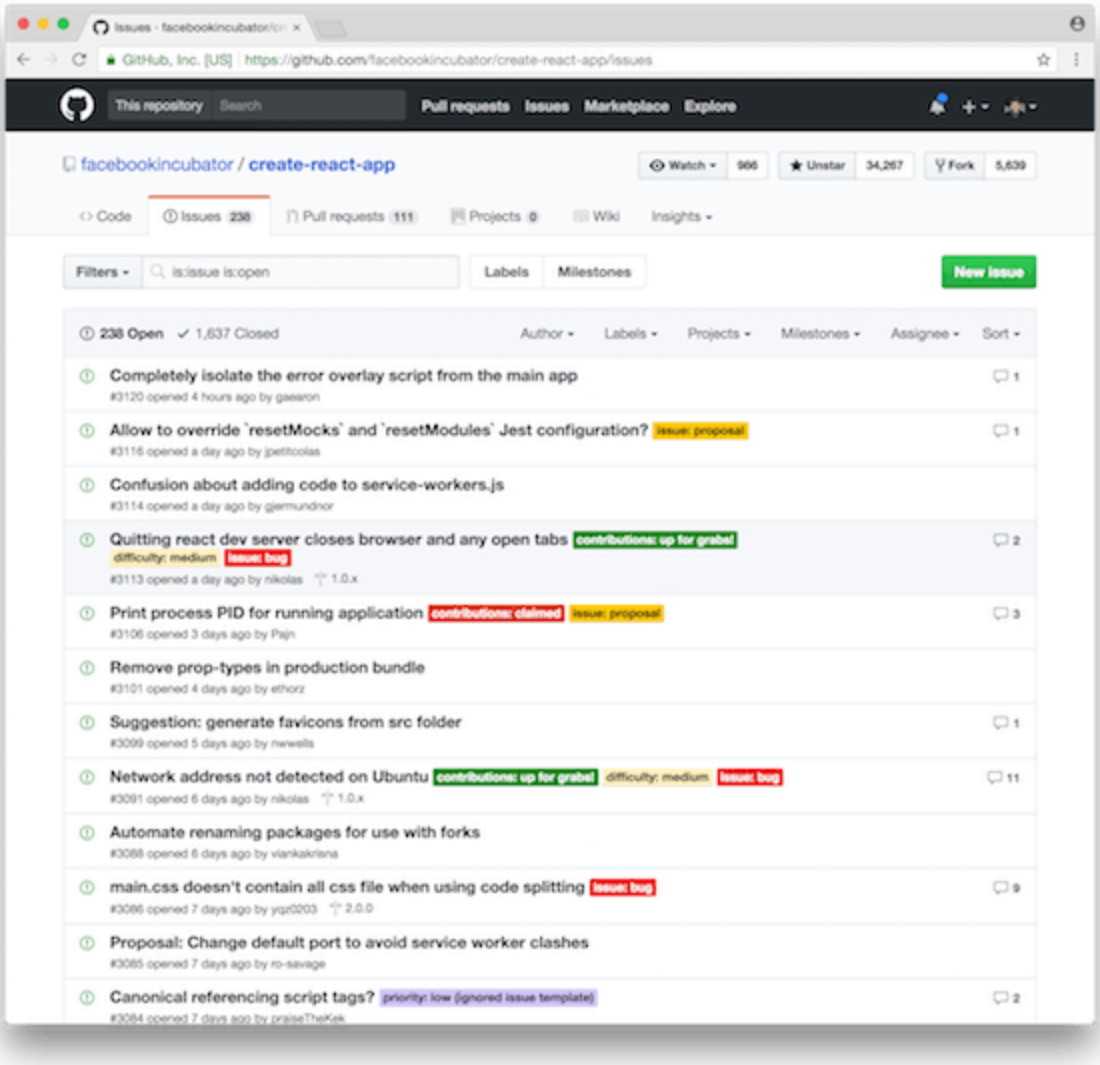


Example websites:

- https://www.accuweather.com/en/in/india-weather
- https://weather.com/en-IN/weather/tenday/l/Bangalore+Karnataka?canonicalCityId=3a11e3a8b23fbf8668c01ac049c8b4d8df31f270ab2b1b195f5756a4407ee3d4
- icts.res.in/weather/India/Karnataka/Bangalore/1

Task 7: Github Issues Page
Estimated Time: 2 day

This project is to design a simplified version of Github's Issues page. Collect open issues from Github's API and display them in a list. After this, add a pagination control to facilitate navigation across a complete list of issues. If you add a React Router, you can navigate directly to a particular page of your choice. You can follow the similar design in the below image. You can ignore the header and other links. Just you need to list the issues, filter options and add pagination.



Useful Reference Links

React Tutorial : https://www.youtube.com/watch?v=QFaFIcGhPoM&list=PLC3y8-rFHvwgg3vaYJgHGnModB54rxOk3&index=1
https://reactjs.org/tutorial/tutorial.html
https://react-tutorial.app/app.html
https://www.udemy.com/course/react-tutorial/learn/lecture/13683138#overview

Working with Ajax request:
https://jasonwatmore.com/post/2020/07/17/react-axios-http-get-request-examples
Handling error response in axios:

https://www.npmjs.com/package//axios#handling-errors

Using proxy for Ajax:
https://medium.com/bb-tutorials-and-thoughts/react-how-to-proxy-to-backend-server-5588a9e0347

React Test script reference:
https://testing-library.com/docs/react-testing-library/example-intro/
https://jestjs.io/docs/tutorial-react
https://jestjs.io/docs/setup-teardown
https://levelup.gitconnected.com/how-to-write-unit-tests-with-react-testing-library-d9624fd2b707

Perl Tutorial PDF: http://academy.delmar.edu/Courses/ITSC1358/eBooks/PracticalPerl.pdf
Perl Coding Standards : https://perldoc.perl.org/perlstyle
Perl Special Variables: https://perldoc.perl.org/perlvar
Perl OOPS: https://youtu.be/rCCnRIX7mLo
https://youtu.be/PSt-uVePXak
Perls OOPS concepts: https://www.perltutorial.org/perl-oop/
https://www.geeksforgeeks.org/object-oriented-programming-oops-in-perl/
Useful Perl Modules:
https://drive.google.com/file/d/1xQkMvHqA1WFuUhRfQZGWFYuMxHRPDDtJ/view?usp=sharing
Perl Built-in functions list: https://metacpan.org/pod/distribution/perl/pod/perlfunc.pod
Perl POD documentation: https://perldoc.perl.org/perlpod
https://stackoverflow.com/questions/35508007/using-get-options-and-pod-usage-in-perl
Perl File operations : https://perlmaven.com/file-test-operators

Setting HTTP Status code using CGI: https://metacpan.org/pod/CGI#Creating-a-standard-http-header
Standard HTTP Status: https://datatracker.ietf.org/doc/html/rfc2616#section-10.4.5

Perl test script references:
https://metacpan.org/pod/Test::Deep#With-Test::Deep
https://perlmaven.com/testing-a-simple-perl-module

Web Security Vulnerability training:
https://www.hacksplaining.com/owasp
https://www.hacksplaining.com/lessons