













Kanban Estimation

Home Page

Estimated completion 19hr. Assigned to Mohammed Mahroof.

Tasks  9		Add 	
<input type="radio"/> Added navigation bar			
<input type="radio"/> Check navigation bar works for all pages			
<input type="radio"/> Uploaded logo/name			
<input type="radio"/> Log in and Sign up Buttons work and redirect you			
<input type="radio"/> Widgets can be added			
<input type="radio"/> Widgets can be deleted			
<input type="radio"/> Widgets can be rearranged			
<input type="radio"/> Welcome message is displayed			
<input type="radio"/> Current location is displayed			

All the tasks for the homepage are relatively small tasks that should roughly take about 2 hours to complete and test. If a test fails, it could take longer so some tasks may be more difficult than others. This has led to an estimation of 19hrs give or take some. This time can either be used to help others If finished early or to make sure all the tasks are completely and thoroughly tested for a working solution.

Tech Report: API Integration

API integration has become a crucial aspect of modern software development. APIs (Application Programming Interfaces) are sets of prewritten instructions and tools that allow different software applications to communicate with each other. They enable developers to create robust applications that can easily interact with other systems, such as databases, payment gateways, and other web services making development of large-scale applications easier. In this tech report, we will explore API libraries/integration.

API integration is the process of connecting different software systems to exchange data and functionality. API integration enables developers to create applications that are scalable, flexible, and efficient. It also allows applications to communicate with external services and exchange data securely. Our system will use at least 4 APIs to provide our users with useful features and easier use of the system. These will be a map API, crime API and verification/authentication API. There may be more if we need to implement other features into our system.

The map API will integrate a map into our system which will allow users to view events in their local area, local businesses, crime alerts in a particular area or search for an emergency station. The crime API will use a database to pull data from a larger reliable database about crimes in the area and display them on the web application. This guarantees data integrity and provides the latest crime information from a reliable source. The verification/authentication will be an API that verifies a user is over 18 by checking their ID credentials they provide to the system. The widget API will be used on the home screen to display.

To pull data from an API you need to use HTTP requests which are also known as CRUD operations. These allow either creation, reading, updating or deleting from a database and in our system, we will be using the reading operation to pull reliable and accurate information from a large database and display it in our system. The general steps are:

1. Generate an entity: You can generate an entity using the JHipster command-line interface (CLI) using the command **jhipster entity <entity-name>**. This will generate the necessary Java classes, database schema, and REST API endpoints for your entity.
2. Start the application: You can start the JHipster application using the command **./mvnw**.
3. Access the API: Once the application is running, you can access the REST API endpoints by sending HTTP requests to the server's URL. You can use tools like Postman or cURL to send requests to the API.
4. Query the data: To retrieve data from the API, you can use the HTTP GET method and specify the endpoint URL for the entity you want to retrieve. For example, if you have an entity called **Product**, you can retrieve all products by sending an HTTP GET request to **http://localhost:8080/api/products**.
5. Handle the response: The API will return data in JSON format. You can parse the JSON response and use the data in your application.

Useful links:

What is an API: https://www.youtube.com/watch?v=ZveW4_ZItVY

Tech Stack/CI Development

Team Projects 2022-23 > team20-22 > Commits > 3c3b5da6

Commit 3c3b5da6 authored 11 minutes ago by Mohammed Mahroof

Browse files

Options ▾

Reformatted home page

parent f730066a main

No related merge requests found

Pipeline #4549 passed with stages in 8 minutes and 20 seconds

```
ismail@Ismails-MacBook-Pro-2 team20-22 % git add .
ismail@Ismails-MacBook-Pro-2 team20-22 % git commit -m "Reformatted home page"
Using node installed locally v16.17.0
Using npm installed locally 8.19.1
✓ Preparing lint-staged...
✓ Running tasks for staged files...
✓ Applying modifications from tasks...
✓ Cleaning up temporary files...
[main 3c3b5da] Reformatted home page
 2 files changed, 32 insertions(+), 101 deletions(-)
  rewrite src/main/webapp/app/home/home.component.html (70%)
  rewrite src/main/webapp/app/home/home.component.scss (79%)
ismail@Ismails-MacBook-Pro-2 team20-22 % git push
Enumerating objects: 17, done.
Counting objects: 100% (17/17), done.
Delta compression using up to 4 threads
Compressing objects: 100% (9/9), done.
Writing objects: 100% (9/9), 794 bytes | 794.00 KiB/s, done.
Total 9 (delta 7), reused 0 (delta 0)
To git.cs.bham.ac.uk:team-projects-2022-23/team20-22.git
 f730066..3c3b5da  main -> main
ismail@Ismails-MacBook-Pro-2 team20-22 %
```

I reformatted the home page to make it relevant and suitable for our website build. This will be tweaked and linked with the rest of the pages and databases to ensure a fully functioning homepage that provides information for the user at first glance.