CINEFY: Movie booking Application

Submitted by: - Parth Parashar (PSU ID: - 923928157)

Other Project Partners: - Pragati Rathore and Kirtan Patel

GITHUB repository Links: -

https://github.com/parasharparth/cinefy-backend

https://github.com/parasharparth/cinefy-frontend

The link for my presentation is: -

https://youtu.be/xhggLGwXZfU

I tried reaching out to the other members of the team for preparing a group presentation but there wasn't any response from any of them which is why I had to prepare and submit my part of the project presentation in this link.

This document consists of the following topics: -

Tech Stack: -

The project tech stack which was decided upon in the beginning of the project included: -

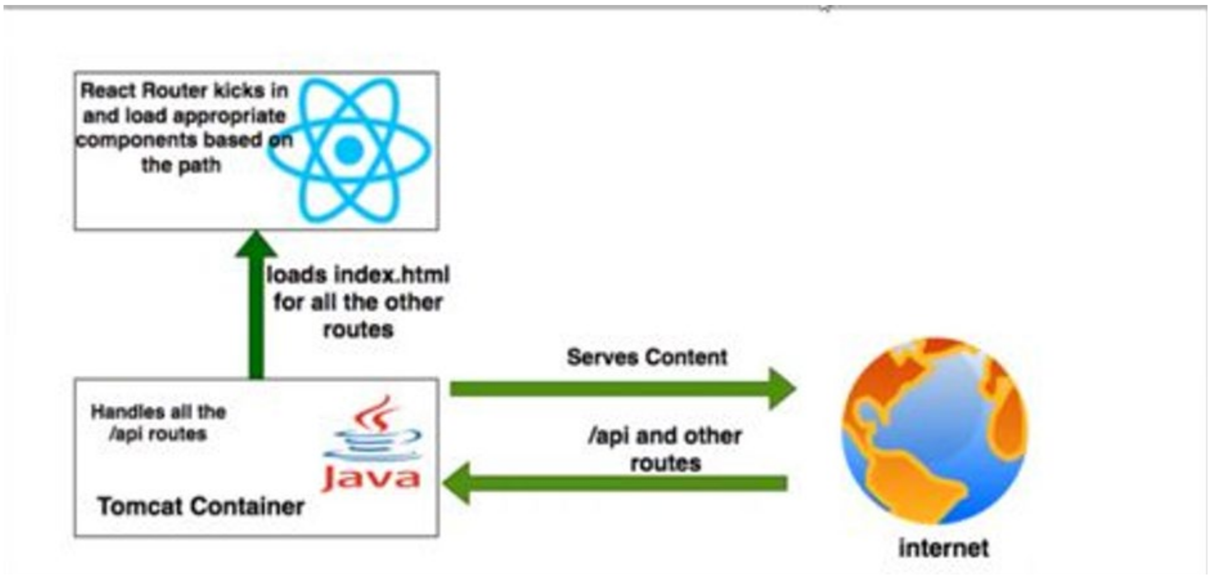
Backend- Spring-boot (Java) and Docker

Database used- POSTGRE SQL

Front-end- React JS.

Architecture followed: -

The architecture being followed for the project is: -


React Router kicks in and load appropriate components based on the path
loads index.html for all the other routes
Serves Content
Handles all the /api routes
Java
Tomcat Container
/api and other routes
internet

From this picture, instead of tomcat container, we have Docker container.

Instead of internet, we will have POSTGRE SQL.

Also, the routing of the data will be done using the java containers, which in case of spring-boot are the microservices which will be called by both react and POSTGRE SQL.

For the data coming from the POSTGRE, different set of microservices will be used and for the data coming to and from the REACT microservices, different set of API(s) will be used. The docker will act as a container for encapsulating all the activities for backend.

Running the project: -

For starting the project, we need to start the backend first. For starting the backend, there are three components which need to be started in the given order: -

1) Docker

2) POSTGRE SQL Server
3) Java Sprint-boot application using maven scripts

For starting the spring-boot project, open the project in intellJ or other editor such as eclipse and then click on the run profiles sections where we have to create the run profile: -

```
$JAVA_HOME/bin/javac  -java backend-SNAPSHOT-0.0.1.jar
```

Click on run and the project will start and will create the required microservices will be created.

For starting the front-end, we need to get the project from the github repository in a code editor such as VSCode and then in the integrated terminated, run the following commands: -

```
Npm build

Npm build run

Npm start
```

Npm start will prepare the react application and start it on the front-end which in this case is a web-browser.

MY PROJECT JOURNAL: - (Detailing my work and contribution to the project)

While the discussions around the project were there from the beginning of the quarter, it all started with the formation of teams.

After the team was selected, it was week-2 when we started discussing and working on the project

<u>Week of 24 April – 1 May: - (Called week-1)</u>

In this week, we had a team meeting where we pondered over the topic of the project. There was a plethora of suggestions which were put across the board ranging from a e-commerce marketplace to a booking system.

Since we all are movie fanatics, we decided to make our project on a movie ticket booking system. This can further be extended to have other booking systems as well.

In this week, we discussed what tech stack we are going to use. Since it is a full-stack project, we decided to use react + Java spring-boot + POSTGRE SQL for the project.

I was always curious as to how different java frameworks worked in conjunction with the integrated front-end where both use microservices to render the different parts of the web application. Because of this, I decided to take the java spring-boot part of the project. Kirtan took up the front-end part of the project and Pragati decided to take up the devops part.

We also decided which features would be included in the web application. The features which were agreed upon in the first meeting were: -

1) Login Page
2) Home Page
3) Booking Page
4) Payment Page
5) About us Page

Since the Spring-boot framework was new to me, I did some digging as to what I need to have in-place for succeeding in making microservices in Spring-boot.

Some of the materials which I studied included (but were not limited to) are: -

https://spring.io/projects/spring-boot

https://www.javatpoint.com/spring-boot-tutorial

https://www.baeldung.com/spring-boot

Week-2

During this week of the project, I decided to make the skeleton projects for both front-end and back-end. For back-end, I did so by using Spring initializer pack from eclipse which creates the basic file structure containing the following files: -

1)src (main + test)

2) pom.xml

3) dependency graph and folder

4) .mvn

For Front-end, I created the skeleton project by using the React npx command.

The command for the same is given below: -

Npx create-react-app cinefy-frontend

This creates the skeleton react application.

After the skeleton project was created, I wanted to put the projects in a github repository so that every member of the team can start working on the project as soon as possible. I tried to make folders in the github repository for front-end and back-end and then tried to push the skeleton projects (both front-end and back-end) into the github repository. The projects did push in the github repository. But there was an issue, none of the project properties were being recognised by the GIT command prompt. This is because both projects were from different structural configurations. This meant that it was futile to put both of them in the same repository and expect them to work.

I did try a few other things over the next couple of days to overcome this situation, but it did not improve. So, in the end, I decided to put both front-end and back-end in different repositories for the sake of simplicity and getting the project started.

After this, I did start working on the featured movie page for which I created the MVC classes (Domain, controller) along with the JPA repository class for getting data from the table. The domain which maps to the JPA repository element was also created.

I also decided to put the different versions of HTTP errors as user defined classes and put in a couple of error classes as well. I did this because I could modify the message according to out convenience and not print the default error message as they do not appear good for user interaction. This week was a smooth sailing for the project as there was some form of communication as well as there was the work was being finished at a very fast pace.

<u>Week-3: -</u>

During this week, I picked up on the remaining work for the featured movie page by creating the "service" methods to be called in the controller to pass the data effectively. Tested the service with the help of test data and postman.

The next thing which I did in this week was to create the about us page and its domain. I created the classes containing JPA methods to handle the About page.

All this was useful as we wanted to test out the About us page.

Apart from this, I also created the microservice for the booking page. This included the domains (for mapping elements), JPA repository and model. New HTTP exceptions were added as well. Simultaneously, I also prepared the services for login page as well. The domains were added for both these pages, and I waited for the front-end to catch up with the back-end at this point.

I wanted to test these but that could not happen because the database was running behind from the devops end and I ended up testing these services using dummy data and postman again so as to make sure that all the bases are covered.

Then after this, I started working on the Home - page and created the domains for it in this week.

<u>**Week-4: -**</u>

Now, for this week, I had the task of completing the Home page. I completed the home page by creating all the classes which were required to test the functionality of the home page.

I had an issue with the annotations for home page. Since I had modelled it using class instead of interface, I had to write my own getter and setter along with the other hashing functions for the page to get the data from the database. It included the functions such as findall() as well which were used to get the data from the repository to the controller. It did work out nicely in the end.

At this point in time in the project. I wanted us to test all the functionality of the project by integrating the front-end and back-end which was the task of the devops person but for some reason, when I tried to contact the other team members, nobody responded. The meeting which was mandatory for the project never happened. To test the code which I had written, I again inserted dummy data into the database and tested the microservice of home page using postman.

I tested various types of data for the application so that I can simulate the various input types.

I was getting increasingly worried about the devops part because I had no idea as to what was being done in the devops part as there was a proper lack of communication at this point and even after reaching out to the members, it never resolved. I wanted the project to work properly so I started integrating small snippets of code for the facilitation of the smooth data transfer from front-end to back-end. JSON object was something that we had agreed upon to use as a standard for data transfer so I started introducing code which could handle all the possibilities. I introduced the utility package and conversions package.

The use of the utility package is to improve the mappings and also implement the container for the integration of docker. The classes boxed and unboxed objects is to cover the possibilities of docker being boxed or unboxed. I did not write code in it as it could lead to confusion for the devops person. But I did create a proper structure in the project so that it could help in understanding better. Same goes for the conversions.

Week-5

The utility package which I created in the last week was extended to include the conversions of java arrays to list, hashmap and double list to hashmap as well. This could be used to store the integer or JSON objects. The code can be modified for the data type which needs to go.

So the day of the submission when we had to submit the draft (1st June), I tried to reach out and discuss as to what we need to do, there seemed to be some sort of confusion about the architecture of the project within the team, which led to changed architecture and code of the project in a way that my java part looked redundant. This is why I wanted to have a meeting the last week. I tried to reason with the team but to no avail. The communication seemed off between the team members and the focus now shifted from team to individuality. My code was ignored for the sake of running the project. The integration seemed to be done as it pleased to the developer operations member. The project now looked completely different than it should have been. All this last-minute rush for project is also highlighted in the commit history as well where almost all integration commits were on a single day (the day of the submission).

In the end, the architecture is lost, the use of java became minimal and the use of microservices was just for the sake of it. I am still unable to understand where it went wrong in the communication. I had even tried to explain the architecture to the team members twice before as well, and I did it again but the architecture which they thought was different from the one decided earlier. All this led to me being left stranded and disappointed.

This is when I decided to write a mail to professor seeking help with whom I had a zoom meeting later and explained things from my end.

I have committed the code to the main branch whereas the database code which I wrote for database connectivity is still present in my dev branch on git.

These unfortunate, events in my humble opinion, led to the downfall of the project and after this, the communication completely stopped. Even the database seems to have only a single table which is a far cry from the multiple table architecture we had agreed upon in the beginning.

But still, I decided to chip in and help the other team members with the front-end and connectivity part. I still improved the UI on the About us page by modifying the code for the same.

There were buttons present for the user to navigate to the booking page from the home page and information bet transferred onto the booking page regarding the movie selection from the home page.

I implemented the method for transferring the information and navigating to the correct page on button link. I used <link> tag for the same.

I also implemented the movie list to be displayed on the booking page.

I then implemented the accessibility code in CSS for the about us page.

But I was very disappointed by the fact that the original architecture that we were supposed to use was changed at the very last hour without consulting or informing me which left me in a place where a lot of the code which I wrote became non-usable in the context of the project. It should have been a collective decision but it felt like a decision made by individual.
Since I was primarily coding in java, it looked very bad on my part. What made it even worse was the fact that there was no real effort to make use of the java code when on my end, all I was thinking was how the project could have good amounts of equal contributions from everyone.

Even after multiple reminders and tries to reach out to the team, the communication was not there which led to this situation.

OVERALL EXPERIENCE

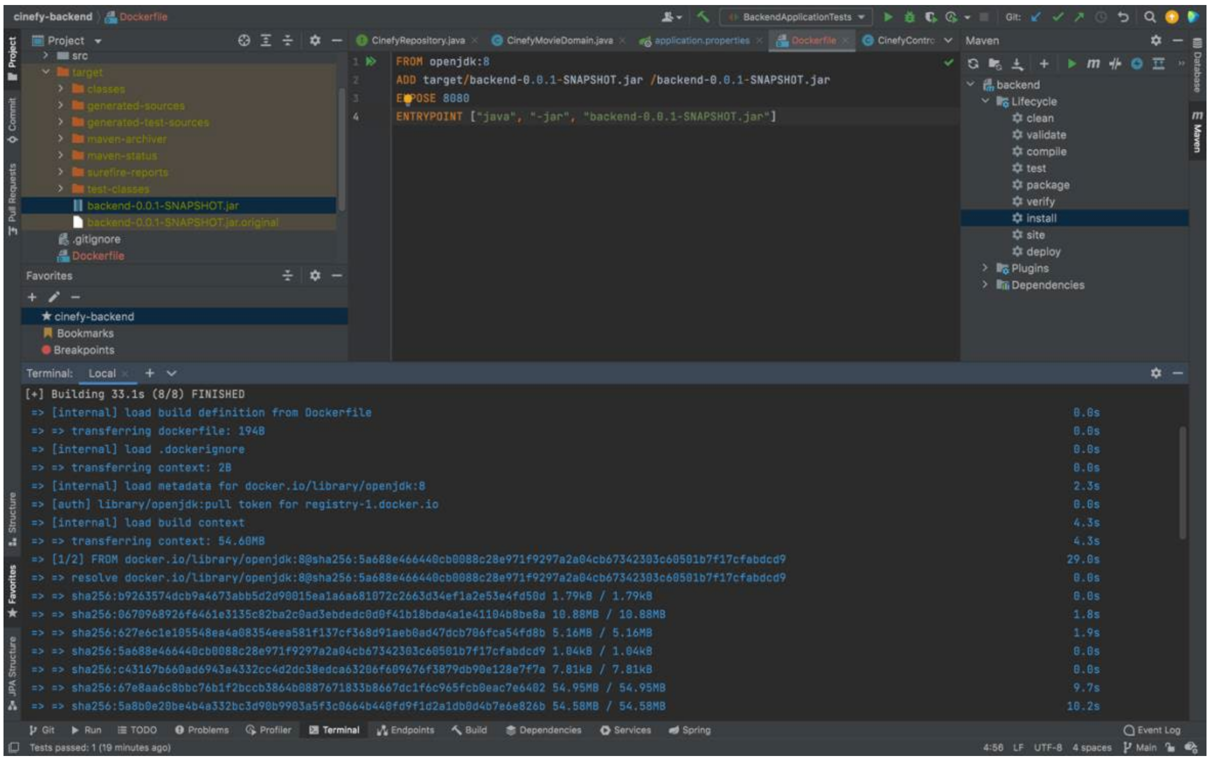
My overall experience in the project was that I was able to learn a lot about full-stack development and how it is an entirely team dependent thing. The need for effective communication in the team and proactive code integration is a must for the development of the project. Team effort and environment is of utmost importance. But I got to learn a lot in terms of the technical knowledge. I was able to learn a completely new framework in terms of spring-boot. It definitely made me aware of the advanced java practices like the use of spring annotations, the use of pom.xml for dependency, the use of dependency injections to make use of unrelated classes is something that would definitely help me write java code more compactly and efficiently. I was also able to learn the importance of react microservices which if connected properly with the backend can render an application flawlessly and is so easy to debug. Even if one microservice malfunctions, everything does not come crashing down. I came to understand this when my home page domain and controller were continuously throwing errors which mostly included the dependency errors relating to missing dependencies in the pom.xml. But I was still able to run the other components without them failing at the same time. This means that when implemented properly, microservices are a real boon. The distribution of work also becomes easy as no one is really dependent on the other.

What I really loved about the project was the even after so many hurdles, I was able to grow as a person who now appreciates the importance of good team environments.

While in the end, I was disappointed with how my work was not in proper use and the architecture was different, it was still a worthwhile experience.

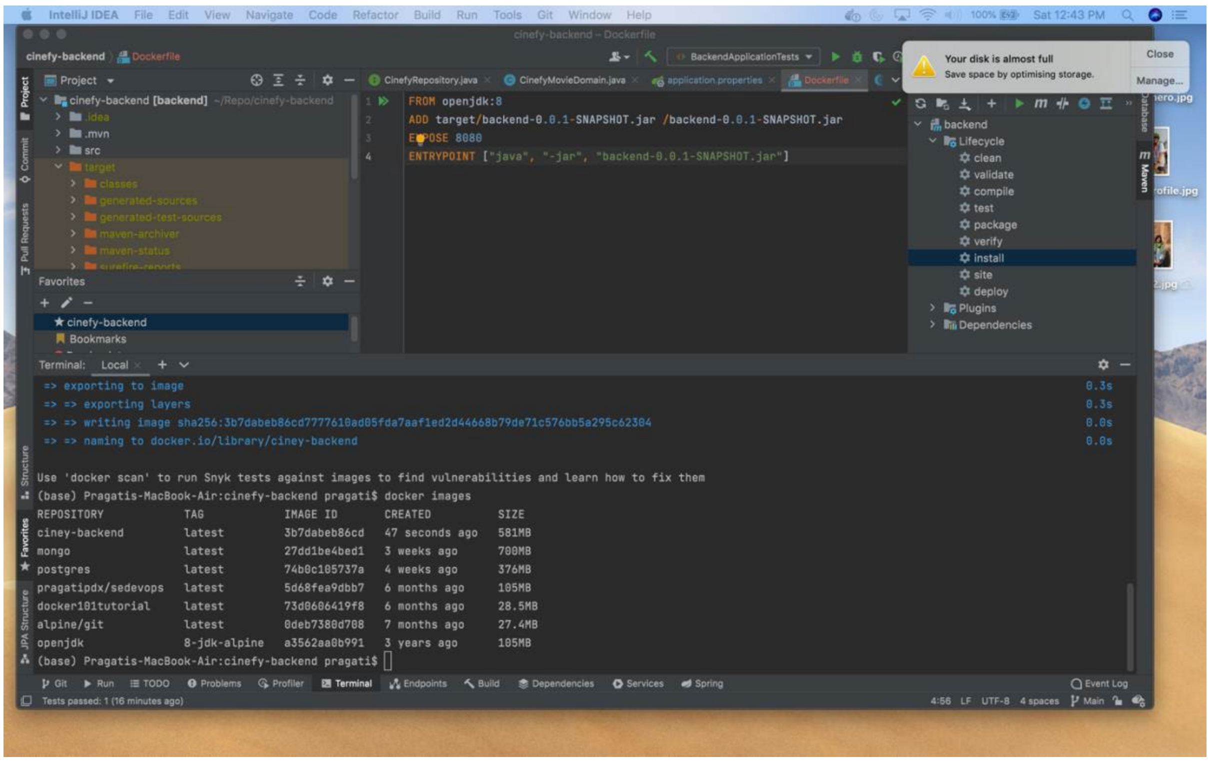
SNAPSHOTS OF THE PROJECT: -

Docker up and running: -



```
FROM openjdk:8
ADD target/backend-0.0.1-SNAPSHOT.jar /backend-0.0.1-SNAPSHOT.jar
EXPOSE 8080
ENTRYPOINT ["java", "-jar", "backend-0.0.1-SNAPSHOT.jar"]
```

```
[+] Building 33.1s (8/8) FINISHED
 => [internal] load build definition from Dockerfile                                                          0.0s
 => => transferring dockerfile: 194B                                                                          0.0s
 => [internal] load .dockerignore                                                                             0.0s
 => => transferring context: 2B                                                                               0.0s
 => [internal] load metadata for docker.io/library/openjdk:8                                                  2.3s
 => [auth] library/openjdk:pull token for registry-1.docker.io                                                0.0s
 => [internal] load build context                                                                             4.3s
 => => transferring context: 54.60MB                                                                          4.3s
 => [1/2] FROM docker.io/library/openjdk:8@sha256:5a688e466440cb0088c28e971f9297a2a04cb67342303c60501b7f17cfabdcd9   29.0s
 => => resolve docker.io/library/openjdk:8@sha256:5a688e466440cb0088c28e971f9297a2a04cb67342303c60501b7f17cfabdcd9   0.0s
 => => sha256:b9263574dcb9a4673abb5d2d90015ea1a6a681072c2663d34ef1a2e53e4fd50d 1.79kB / 1.79kB                0.0s
 => => sha256:0670968926f6461e3135c82ba2c0ad3ebdedc0d0f41b18bda4a1e41104b8be8a 10.88MB / 10.88MB              1.8s
 => => sha256:627e6c1e105548ea4a08354eea581f137cf368d91aeb0ad47dcb706fca54fd8b 5.16MB / 5.16MB                1.9s
 => => sha256:5a688e466440cb0088c28e971f9297a2a04cb67342303c60501b7f17cfabdcd9 1.04kB / 1.04kB                0.0s
 => => sha256:c43167b660ad6943a4332cc4d2dc38edca63206f609676f3879db90e128e7f7a 7.81kB / 7.81kB                0.0s
 => => sha256:67e8aa6c8bbc76b1f2bccb3864b0887671833b8667dc1f6c965fcb0eac7e6402 54.95MB / 54.95MB              9.7s
 => => sha256:5a8b0e20be4b4a332bc3d90b9903a5f3c0664b440fd9f1d2a1db0d4b7e6e826b 54.58MB / 54.58MB              10.2s
```



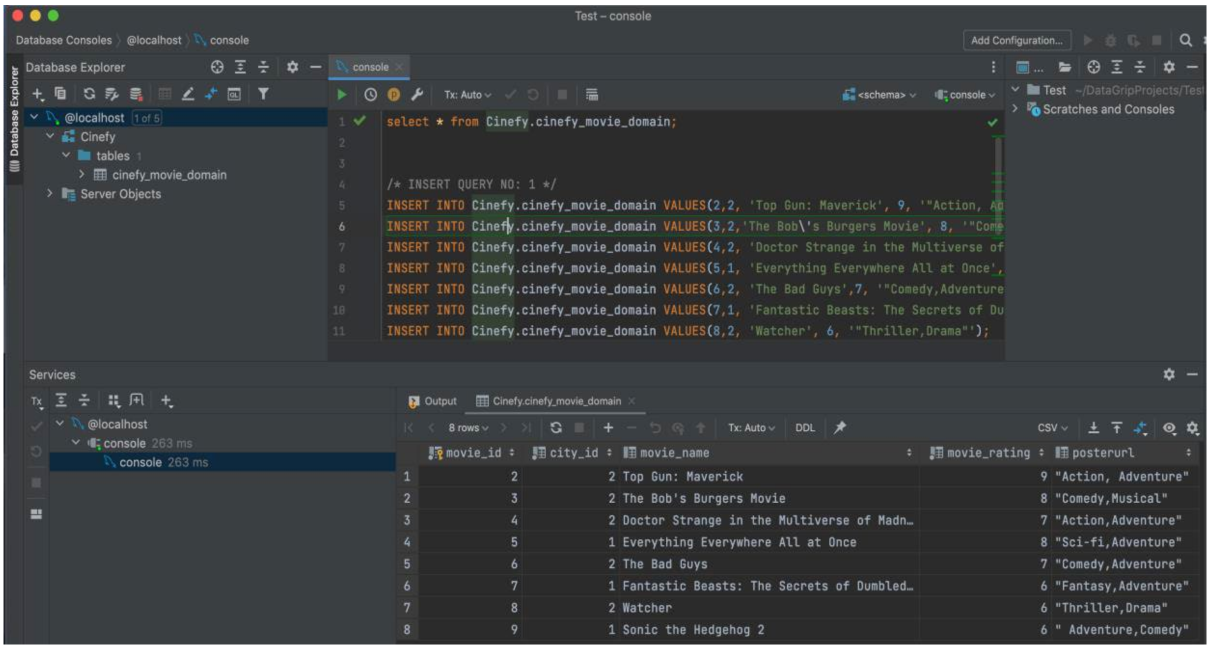
Your disk is almost full
Save space by optimising storage.

```
FROM openjdk:8
ADD target/backend-0.0.1-SNAPSHOT.jar /backend-0.0.1-SNAPSHOT.jar
EXPOSE 8080
ENTRYPOINT ["java", "-jar", "backend-0.0.1-SNAPSHOT.jar"]
```

```
 => exporting to image                                                                                        0.3s
 => => exporting layers                                                                                       0.3s
 => => writing image sha256:3b7dabeb86cd7777610ad05fda7aaf1ed2d44668b79de71c576bb5a295c62304                  0.0s
 => => naming to docker.io/library/ciney-backend                                                              0.0s

Use 'docker scan' to run Snyk tests against images to find vulnerabilities and learn how to fix them
(base) Pragatis-MacBook-Air:cinefy-backend pragati$ docker images
REPOSITORY           TAG            IMAGE ID       CREATED          SIZE
ciney-backend        latest         3b7dabeb86cd   47 seconds ago   581MB
mongo                latest         27dd1be4bed1   3 weeks ago      700MB
postgres             latest         74b0c105737a   4 weeks ago      376MB
pragatipdx/sedevops  latest         5d68fea9dbb7   6 months ago     105MB
docker101tutorial    latest         73d0606419f8   6 months ago     28.5MB
alpine/git           latest         0deb7380d708   7 months ago     27.4MB
openjdk              8-jdk-alpine   a3562aa0b991   3 years ago      105MB
(base) Pragatis-MacBook-Air:cinefy-backend pragati$ 
```

Final Data available in DB: -



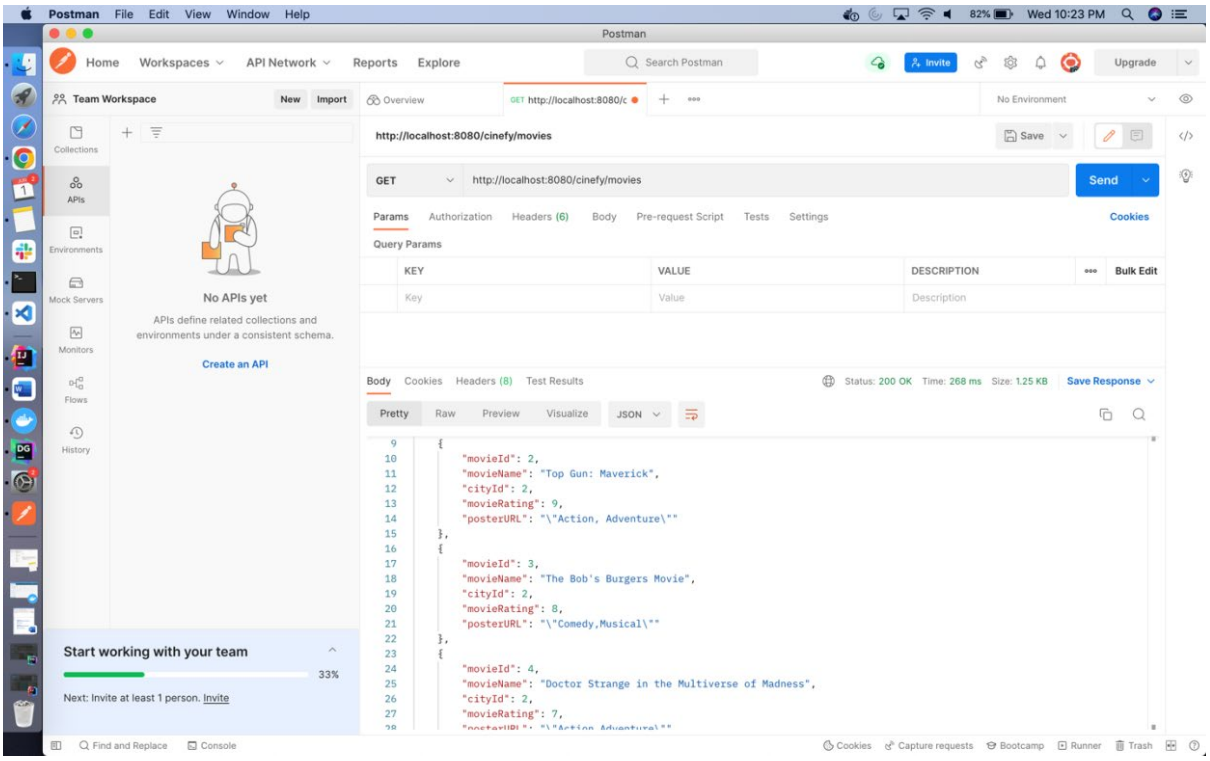
```sql
select * from Cinefy.cinefy_movie_domain;


/* INSERT QUERY NO: 1 */
INSERT INTO Cinefy.cinefy_movie_domain VALUES(2,2, 'Top Gun: Maverick', 9, '"Action, Ad
INSERT INTO Cinefy.cinefy_movie_domain VALUES(3,2,'The Bob\'s Burgers Movie', 8, '"Come
INSERT INTO Cinefy.cinefy_movie_domain VALUES(4,2, 'Doctor Strange in the Multiverse of
INSERT INTO Cinefy.cinefy_movie_domain VALUES(5,1, 'Everything Everywhere All at Once',
INSERT INTO Cinefy.cinefy_movie_domain VALUES(6,2, 'The Bad Guys',7, '"Comedy,Adventure
INSERT INTO Cinefy.cinefy_movie_domain VALUES(7,1, 'Fantastic Beasts: The Secrets of Du
INSERT INTO Cinefy.cinefy_movie_domain VALUES(8,2, 'Watcher', 6, '"Thriller,Drama"');
```

Cinefy.cinefy_movie_domain — 8 rows

	movie_id	city_id	movie_name	movie_rating	posterurl
1	2	2	Top Gun: Maverick	9	"Action, Adventure"
2	3	2	The Bob's Burgers Movie	8	"Comedy,Musical"
3	4	2	Doctor Strange in the Multiverse of Madn…	7	"Action,Adventure"
4	5	1	Everything Everywhere All at Once	8	"Sci-fi,Adventure"
5	6	2	The Bad Guys	7	"Comedy,Adventure"
6	7	1	Fantastic Beasts: The Secrets of Dumbled…	6	"Fantasy,Adventure"
7	8	2	Watcher	6	"Thriller,Drama"
8	9	1	Sonic the Hedgehog 2	6	" Adventure,Comedy"

Backend + Final Database

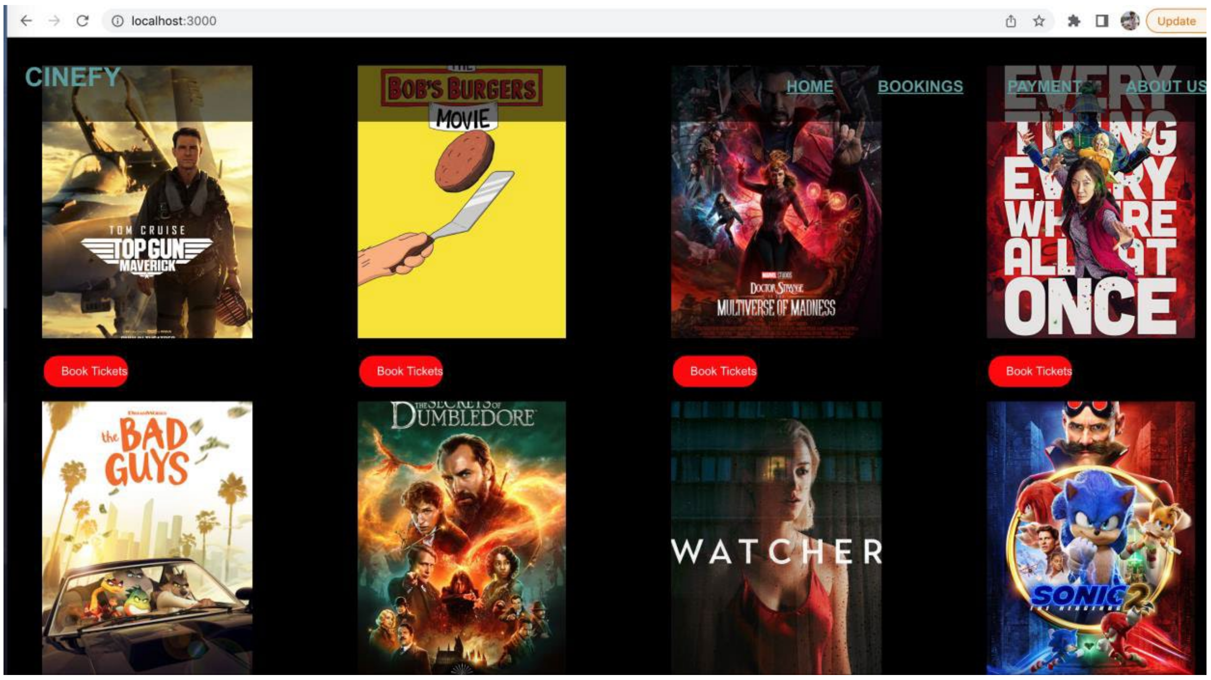


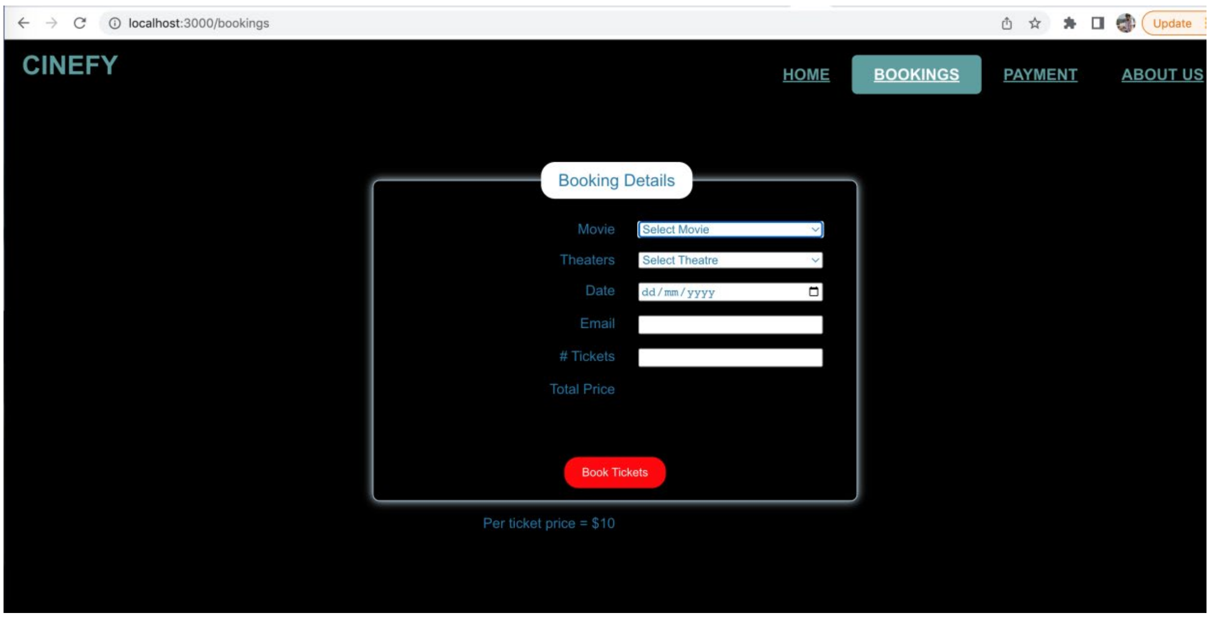
GET http://localhost:8080/cinefy/movies

Status: 200 OK Time: 268 ms Size: 1.25 KB

```json
    {
        "movieId": 2,
        "movieName": "Top Gun: Maverick",
        "cityId": 2,
        "movieRating": 9,
        "posterURL": "\"Action, Adventure\""
    },
    {
        "movieId": 3,
        "movieName": "The Bob's Burgers Movie",
        "cityId": 2,
        "movieRating": 8,
        "posterURL": "\"Comedy,Musical\""
    },
    {
        "movieId": 4,
        "movieName": "Doctor Strange in the Multiverse of Madness",
        "cityId": 2,
        "movieRating": 7,
```

UI: -

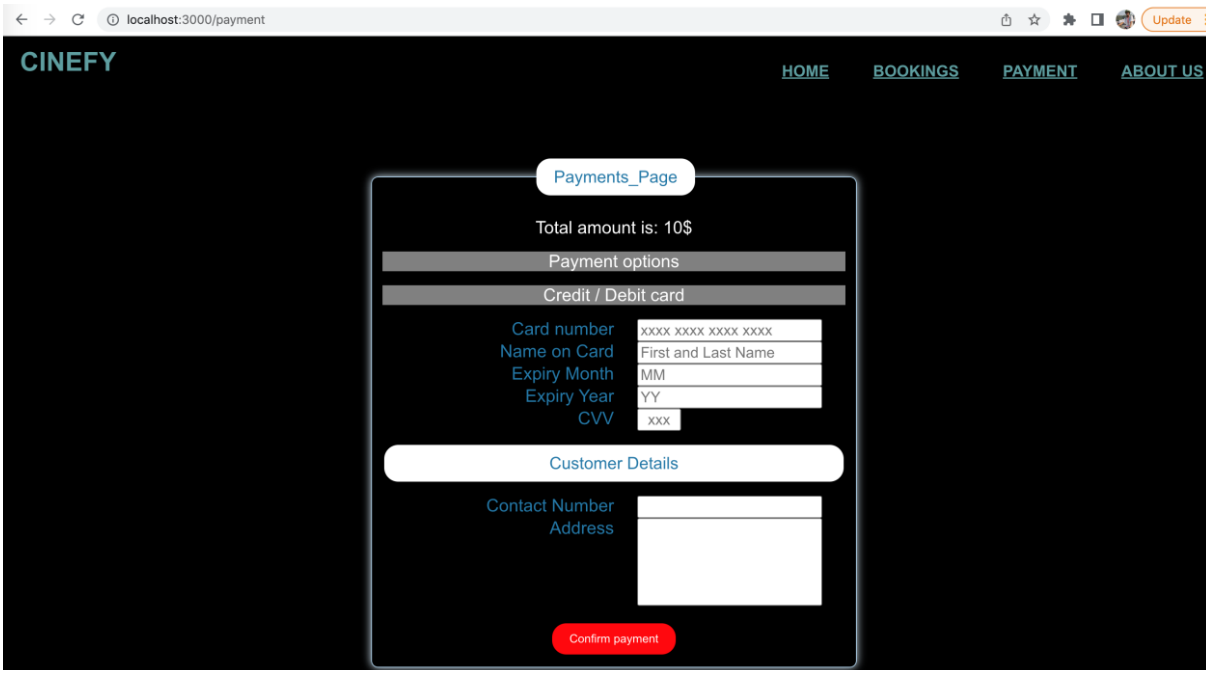




Booking Details

Movie	Select Movie
Theaters	Select Theatre
Date	dd/mm/yyyy
Email	
# Tickets	
Total Price	

Per ticket price = $10



Total amount is: 10$

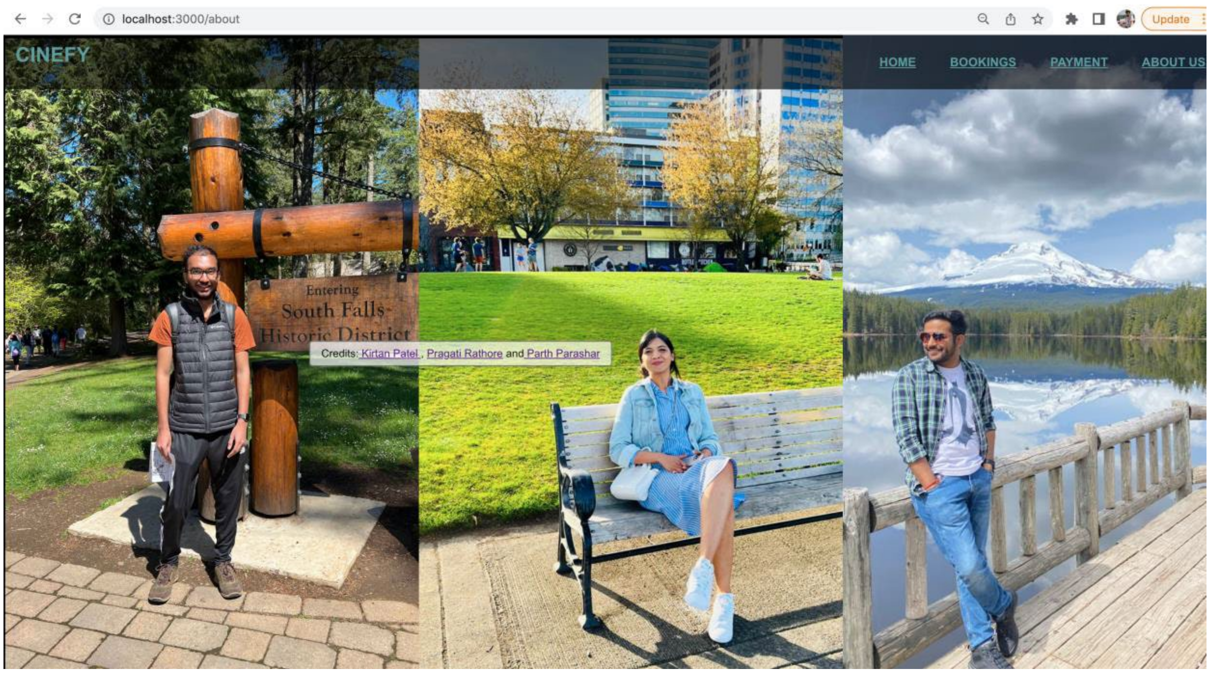
Payment options

Credit / Debit card

Card number	xxxx xxxx xxxx xxxx
Name on Card	First and Last Name
Expiry Month	MM
Expiry Year	YY
CVV	xxx

Customer Details

Contact Number	
Address	

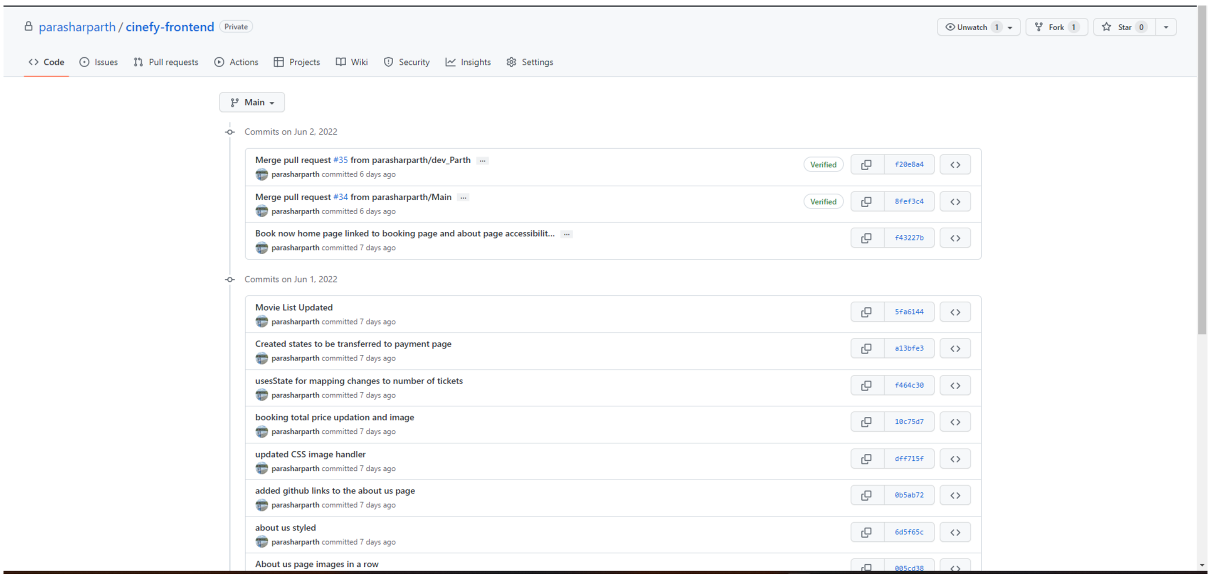


Credits: Kirtan Patel , Pragati Rathore and Parth Parashar

IMPROVEMENTS IN THE PROJECT: -

1) The major improvement which can be made in the project is the proper use of java backend which can actually make this as a proper microservice based web application.
2) Another major improvement in my opinion would be the proper react based UI.

FRONTEND COMMIT(S): -



parasharparth / cinefy-frontend Private

Unwatch 1 · Fork 1 · Star 0

Code · Issues · Pull requests · Actions · Projects · Wiki · Security · Insights · Settings

Main

Commits on Jun 2, 2022

Merge pull request #35 from parasharparth/dev_Parth …
parasharparth committed 6 days ago — Verified — f20e8a4

Merge pull request #34 from parasharparth/Main …
parasharparth committed 6 days ago — Verified — 8fef3c4

Book now home page linked to booking page and about page accessibilit… …
parasharparth committed 7 days ago — f43227b

Commits on Jun 1, 2022

Movie List Updated
parasharparth committed 7 days ago — 5fa6144

Created states to be transferred to payment page
parasharparth committed 7 days ago — a13bfe3

usesState for mapping changes to number of tickets
parasharparth committed 7 days ago — f464c30

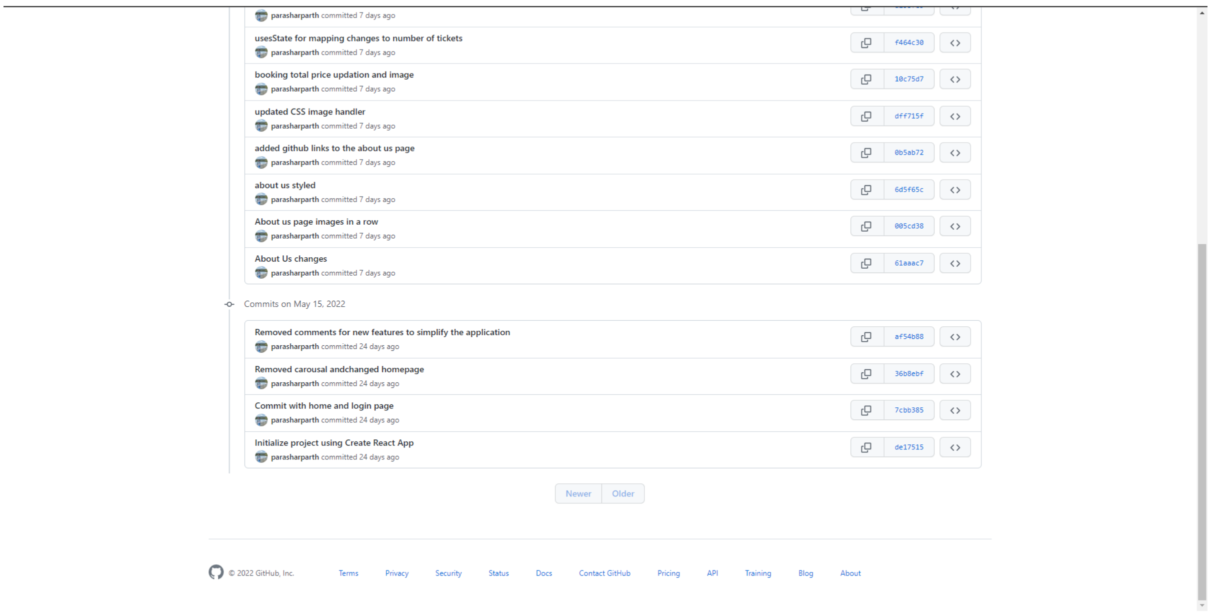
booking total price updation and image
parasharparth committed 7 days ago — 10c75d7

updated CSS image handler
parasharparth committed 7 days ago — dff715f

added github links to the about us page
parasharparth committed 7 days ago — 0b5ab72

about us styled
parasharparth committed 7 days ago — 6d5f65c

About us page images in a row — 005cd38



parasharparth committed 7 days ago

usesState for mapping changes to number of tickets
parasharparth committed 7 days ago — f464c30

booking total price updation and image
parasharparth committed 7 days ago — 10c75d7

updated CSS image handler
parasharparth committed 7 days ago — dff715f

added github links to the about us page
parasharparth committed 7 days ago — 0b5ab72

about us styled
parasharparth committed 7 days ago — 6d5f65c

About us page images in a row
parasharparth committed 7 days ago — 005cd38

About Us changes
parasharparth committed 7 days ago — 61aaac7

Commits on May 15, 2022

Removed comments for new features to simplify the application
parasharparth committed 24 days ago — af54b88

Removed carousal andchanged homepage
parasharparth committed 24 days ago — 36b8ebf

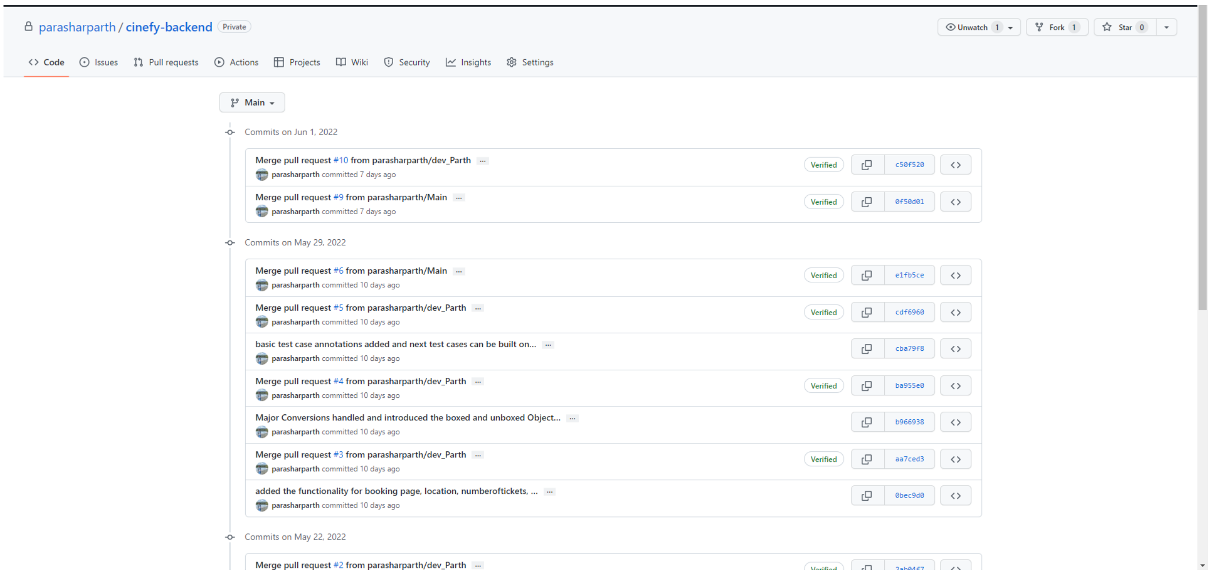
Commit with home and login page
parasharparth committed 24 days ago — 7cbb385

Initialize project using Create React App
parasharparth committed 24 days ago — de17515

Newer · Older

© 2022 GitHub, Inc. · Terms · Privacy · Security · Status · Docs · Contact GitHub · Pricing · API · Training · Blog · About

BACKEND COMMITS: -


parasharparth / cinefy-backend Private

Main

Commits on Jun 1, 2022

Merge pull request #10 from parasharparth/dev_Parth … Verified c50f520
parasharparth committed 7 days ago

Merge pull request #9 from parasharparth/Main … Verified 0f50d01
parasharparth committed 7 days ago

Commits on May 29, 2022

Merge pull request #6 from parasharparth/Main … Verified e1fb5ce
parasharparth committed 10 days ago

Merge pull request #5 from parasharparth/dev_Parth … Verified cdf6960
parasharparth committed 10 days ago

basic test case annotations added and next test cases can be built on… … cba79f8
parasharparth committed 10 days ago

Merge pull request #4 from parasharparth/dev_Parth … Verified ba955e0
parasharparth committed 10 days ago

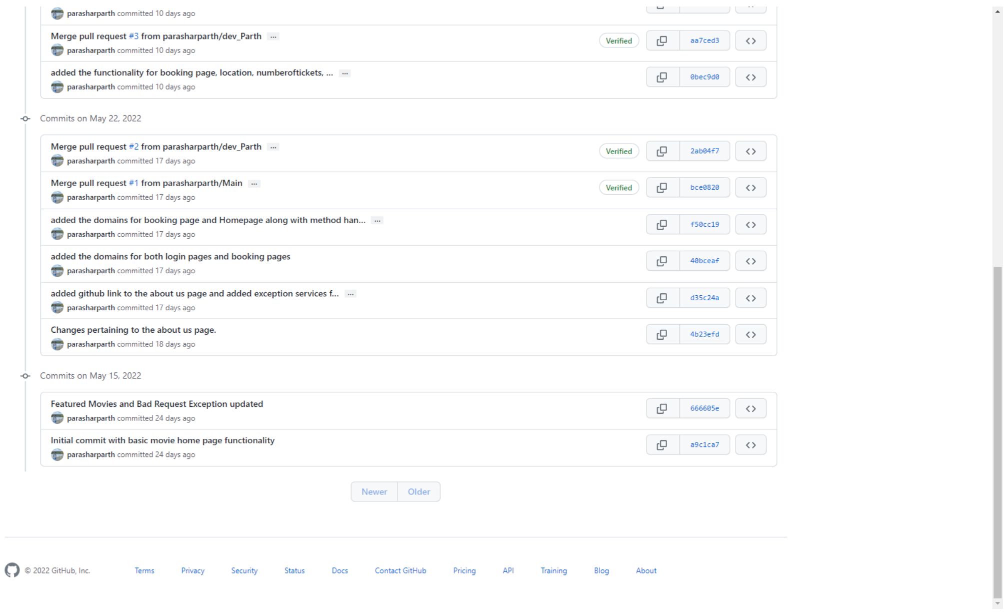
Major Conversions handled and introduced the boxed and unboxed Object… … b966938
parasharparth committed 10 days ago

Merge pull request #3 from parasharparth/dev_Parth … Verified aa7ced3
parasharparth committed 10 days ago

added the functionality for booking page, location, numberoftickets, … … 0bec9d0
parasharparth committed 10 days ago

Commits on May 22, 2022

Merge pull request #2 from parasharparth/dev_Parth …


parasharparth committed 10 days ago

Merge pull request #3 from parasharparth/dev_Parth … Verified aa7ced3
parasharparth committed 10 days ago

added the functionality for booking page, location, numberoftickets, … … 0bec9d0
parasharparth committed 10 days ago

Commits on May 22, 2022

Merge pull request #2 from parasharparth/dev_Parth … Verified 2ab04f7
parasharparth committed 17 days ago

Merge pull request #1 from parasharparth/Main … Verified bce0820
parasharparth committed 17 days ago

added the domains for booking page and Homepage along with method han… … f50cc19
parasharparth committed 17 days ago

added the domains for both login pages and booking pages 40bceaf
parasharparth committed 17 days ago

added github link to the about us page and added exception services f… … d35c24a
parasharparth committed 17 days ago

Changes pertaining to the about us page. 4b23efd
parasharparth committed 18 days ago

Commits on May 15, 2022

Featured Movies and Bad Request Exception updated 666605e
parasharparth committed 24 days ago

Initial commit with basic movie home page functionality a9c1ca7
parasharparth committed 24 days ago

Newer Older

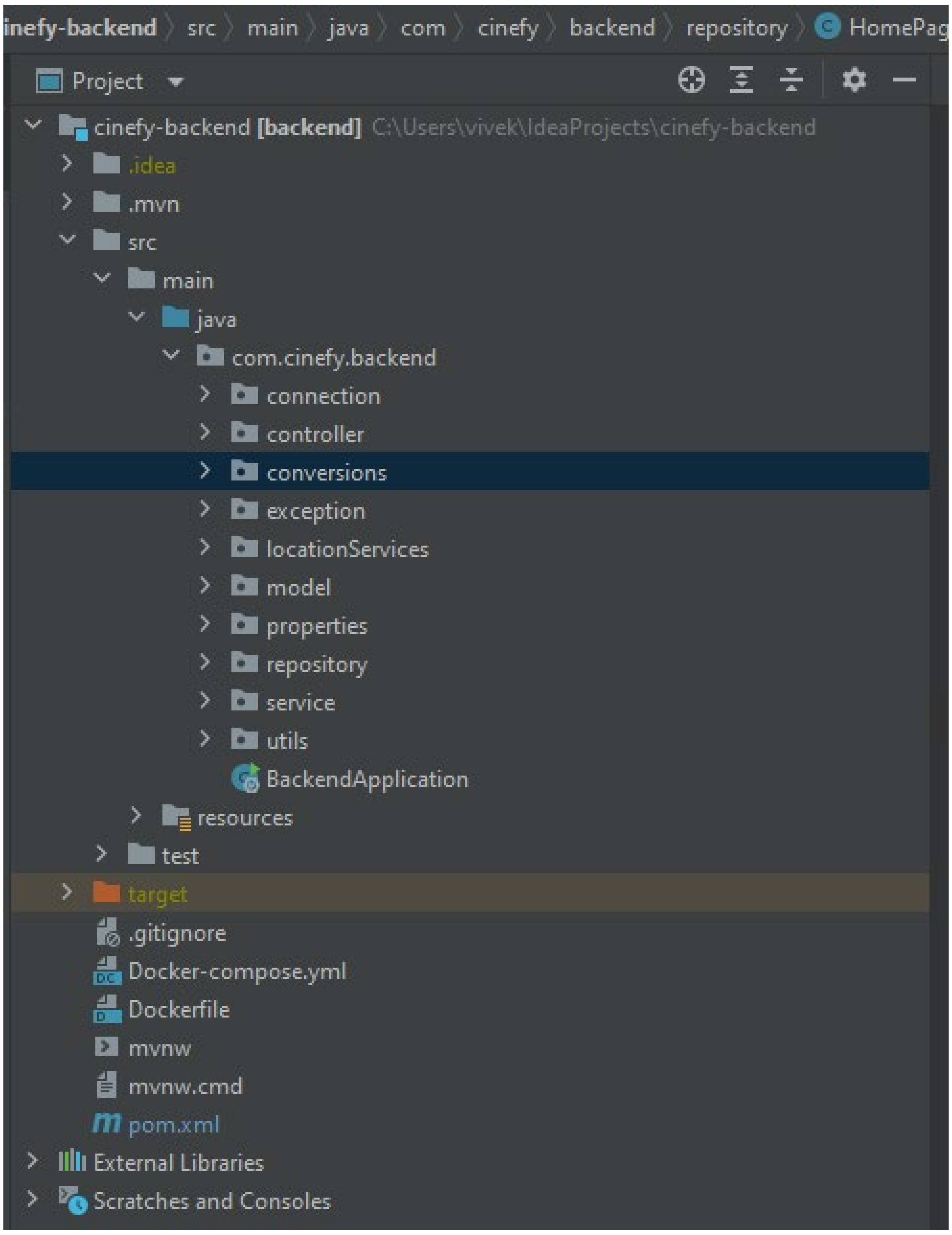
© 2022 GitHub, Inc. Terms Privacy Security Status Docs Contact GitHub Pricing API Training Blog About

NOTE: -

The backend commit is some commits ahead because I had prepared the db connectivity code which I had not committed because the devops person had prepared a different code for db connectivity. Not sure how it works, I have not committed it to the main branch but I have committed it to my dev_parth branch for the professor to look at just in case it is needed to reference.

CODE SNIPPETS: -

Backend structure: -



inefy-backend 〉 src 〉 main 〉 java 〉 com 〉 cinefy 〉 backend 〉 repository 〉 HomePag

Project

```
cinefy-backend [backend] C:\Users\vivek\IdeaProjects\cinefy-backend
  .idea
  .mvn
  src
    main
      java
        com.cinefy.backend
          connection
          controller
          conversions
          exception
          locationServices
          model
          properties
          repository
          service
          utils
          BackendApplication
      resources
    test
  target
  .gitignore
  Docker-compose.yml
  Dockerfile
  mvnw
  mvnw.cmd
  pom.xml
External Libraries
Scratches and Consoles
```