QA Cinema

Team Earth Industries

Introduction

Today we are going to talk about how we developed a full-stack Cinema website, from splitting up the group into two smaller teams and delegating roles, to linking the front and back end to create a functioning site.

Purpose

Task: To create a full-stack website that should present information about movies, listings, upcoming releases.

The application must be suitable for a given client specification, with utilisation of supporting tools, methodologies and technologies that encapsulate all modules covered during training.

Agile approach

We planned our work by breaking down the tasks into smaller chunks so we can complete it effectively

Planning and brainstorm ideas

Split team into FEDs and BEDs

Split the tasks accordingly to our strengths

Created Wireframes to show the overall design of the webpage

User stories and milestones

MoSCoW Methodology- Jira Kanban Board

MoSCoW is a way of prioritising requirements for the project - Must have, Should have, Could have and Will not have.

As a team we created user stories based on the project specification and we grouped them based on the categories in MoSCoW.

With our task hierarchy we were able to create two sprints based on priority.

The first sprint focused mainly on back-end development whereas the second sprint mainly focused on front-end development

Sprint Plan

Initial 2 Sprints:

Site API:

- Bookings
- Movies
- Viewings

Main Website:

- 8 pages
- Bookings
- Current & upcoming viewings

Future Sprints

Sprint 3 - Users

- User Administration
  - Login/Registration pages
  - Profile management
- Booking management (BED pre-existing)

Sprint 4 - Social

- Logo design
- Discussion
  - Comments
  - Queries

Technologies Used

- Github
- Jira
- HTML, CSS, JS
- Postman
- Mongo Db
- Express js
- React JS
- Node js

M E R N

Web Accessibility and Responsiveness

Dark, well-contrasted colour scheme. Dark blue and white font.

Strict use of non-bright colours

Images that fail to load can be identified by reading the text

Pages are responsive to screen size, so can be easily accessed on all devices

Search bar functionality= Easy accessibility for all users

Form boxes e.g. message can be extended for user to see full message.

These are the majority. More will be covered in demonstration.

Risk Assessment

Description of Risk	Risk Impact	Risk Likelihood	Responsibility	Evaluation	Mitigation
Database is attacked and data is breached	High	Medium	Back-end developers	If the URI is exploited or exposed. The database could be compromised, allowing attackers to edit, delete, steal data etc.	Regularly update/backup database to restore data. Encrypt any user data that is stored.
Invalid data input	Medium	Medium	Developer/User	Incorrect data on the application ruins credibility and reliability	Test if the data has been entered
Denial of service attack	High	High	Developers	The site can be flooded with more traffic and data than it can handle causing it to shut down.	Implement a service that redirects abnormal traffic.
Invalid use of the contact page	Low	Medium	Front-end developers / User	If a user pays for a ticket and they don't enjoy it, there is a chance that the contact email is flooded by angry users.	To send an email, the 'subject' box of the email must be filled out, to allow emails to be filtered through.

Evaluation

Problems we faced?

How it set us back?

What we did to overcome this?

Any changes we would make for future sprints/project?

Thank you for listening. Now it's time for the

Application Demonstration