SportSphere

Riley Mei, Yuri Fung, Tal Rabani, Scott Davis, Sophia Crawshaw

SportSphere

- Sports betting information
- Track betting history
- Messaging forums to connect with community

Tools used

- **VSCode:** Used as the primary code editor.
- **Microsoft Azure:** Hosting platform.
- **Node.js and Handlebars:** Backend and frontend technologies.
- **Local Host:** Development environment.
- **HTML:** Frontend markup language.
- **The Odds API:** Sports Betting API for fetching relevant data.
- **GitHub:** Version control and collaboration.
- **SQL:** Database management.
- **Docker** (for containerization), **Bcrypt** (for password hashing).
- **Discord:** For communication
- **Chai and Mocha:** For testing

VSCODE

We rate this tool 5/5 since it is the best IDE in our opinions, and the most intuitive to use

- Since we have a consistent IDE, it means that we can just look at each others computers in order to see what each other are working on without needing to push, improving collaboration

Microsoft Azure

- We used Microsoft Azure for Cloud Hosting
- This tool was very useful for us, and I would give it a 4/5
- This tool enabled a unified view of the website, meaning that we can make local changes but still be able to see what the final version will look like from any device
- Since we all have different computers and operating systems, cloud hosting ensured everyone had the same view

Node.Js

- We used Node.JS as our package manager
- Team used Node.js for server-side development, rating it 4.5/5 for versatility and async handling.
- Node.js fostered seamless collaboration, aided by npm's extensive library ecosystem. Its event-driven architecture ensured scalability, while cross-platform compatibility maintained consistency across diverse environments.

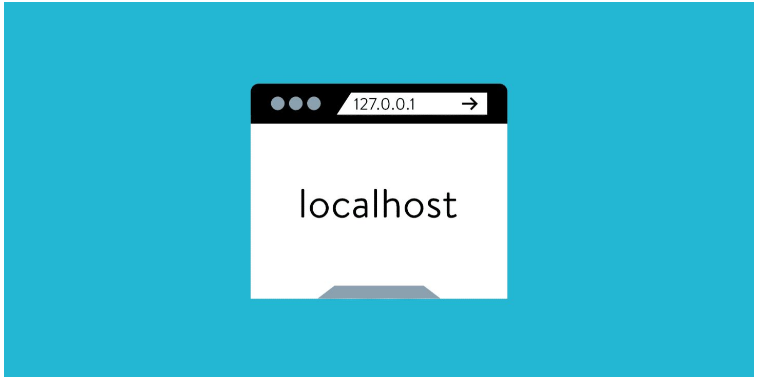
Handlebars

- Team employed Handlebars for templating and dynamic content rendering on the frontend.
- We rate it 2/5 for its simplicity and effectiveness in generating dynamic HTML content.
- Handlebars streamlined our frontend development process by allowing us to create reusable templates and easily inject data into them. Its intuitive syntax and compatibility with JavaScript made it easy for team members to grasp and utilize effectively. Handlebars' support for partials and helpers further enhanced our productivity by promoting code reuse and organization. However, Handlebars' lack of built-in support for more advanced features like conditional logic and loops may require additional workarounds in certain scenarios. Overall, Handlebars provided a solid foundation for building dynamic and maintainable frontend components in our project.
  NO IMAGE RENDER STATIC GG

Chai and Mocha

- Team utilized Chai and Mocha for testing
- We rate them 4/5 for their comprehensive features and ease of use.
- Chai provided expressive assertion styles, making test cases readable and maintainable. Its extensive plugin ecosystem allowed for customization to suit specific testing needs. Mocha served as a flexible test framework, offering support for asynchronous testing and various reporting options.
- Its simple setup and intuitive interface streamlined the testing process, enabling quick identification of bugs and regressions.
- While occasional compatibility issues between Chai and Mocha arose, overall, they proved to be reliable tools for ensuring the quality and stability of our codebase.

LocalHost



- Team utilized localhost for local development
- We rate it 5/5 for its reliability and accessibility.
- Localhost provided a stable environment for testing and debugging code, ensuring smooth development workflow.
- Its simplicity allowed quick setup and configuration across different machines, facilitating collaboration without dependency on external services.

HTML

- Team employed HTML for front-end development
- We rate it 4/5 for its simplicity and widespread support.
- HTML facilitated the creation of structured and interactive web pages, enhancing user experience. Its compatibility across browsers ensured consistent rendering of content, while its intuitive syntax enabled rapid prototyping.
- Despite its limitations in dynamic functionality, HTML served as a foundational language for building modern web applications.

RAPID API

Rapid

- Team leveraged RapidAPI for integrating third-party APIs
- We rate it 4/5 for its extensive API marketplace and ease of use.
- RapidAPI streamlined the process of discovering, testing, and implementing APIs, accelerating development cycles. Its centralized dashboard provided a unified view of integrated APIs, simplifying management and monitoring.
- Additionally, RapidAPI's comprehensive documentation and SDKs facilitated seamless integration into our applications. Despite occasional latency issues, RapidAPI proved invaluable for accessing a wide range of functionalities and services.

GITHUB

- Our team utilized GitHub for version control and collaboration.
- We rate it 5/5 for its robust features and widespread adoption in the development community.
- GitHub streamlined our code management process, allowing seamless collaboration among team members. Its pull request and code review features ensured code quality and consistency. Additionally, GitHub's project management tools enhanced our workflow by providing an organized platform for tracking tasks and issues. Overall, GitHub significantly contributed to the efficiency and success of our project development.

SQL

- Team utilized SQL for database management
- We rate it 4.5/5 for its reliability and flexibility
- SQL enabled efficient querying and manipulation of relational databases, ensuring data integrity and consistency. Its standardized syntax and rich set of features simplified complex operations such as joins and transactions. S
- QL's support for indexing and optimization enhanced performance, enabling fast retrieval of data even from large datasets. Additionally, SQL's compatibility with various database management systems ensured portability across different environments.
- While mastering SQL requires some learning curve, its powerful capabilities made it an indispensable tool for managing data in our projects.

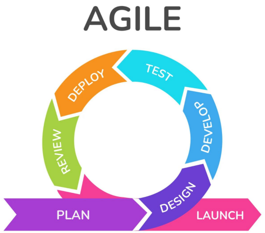
Docker

- Our team leveraged Docker for containerization and deployment.
- We rate it 5/5 for its efficiency and flexibility in managing application dependencies and environments.
- Docker streamlined our development process by encapsulating applications into lightweight, portable containers, ensuring consistency across different environments. Its ease of use and robust documentation facilitated quick setup and deployment, allowing us to focus more on development and less on environment configuration. Docker's scalability and resource efficiency also contributed to improved performance and resource utilization. Overall, Docker played a crucial role in simplifying our deployment pipeline and enhancing our development workflow.

Discord

- Team utilized Discord for communication and collaboration.
- We rate it 5/5 for its versatility and ease of use in facilitating team communication.
- Discord served as our primary platform for real-time communication, allowing seamless interaction among team members. Its intuitive interface and rich feature set, including text channels, voice channels, and direct messaging, promoted efficient collaboration and coordination.
- Discord's support for integrations and bots further enhanced our workflow by automating routine tasks and providing timely notifications. Additionally, Discord's accessibility across devices ensured that team members could stay connected and engaged regardless of their location.
- Overall, Discord significantly contributed to the cohesion and productivity of our team throughout the project lifecycle.

Methodologies Used



Agile Development:

- Embraced Agile principles for adaptability.
- Worked in a 4-week sprint
- Iterated quickly and gathered feedback.
- Daily stand-ups kept team aligned.

Test-Driven Development (TDD):

- Used Chai and Mocha for reliability.
- Wrote tests before implementing features.
- Identified edge cases and validated functionality.

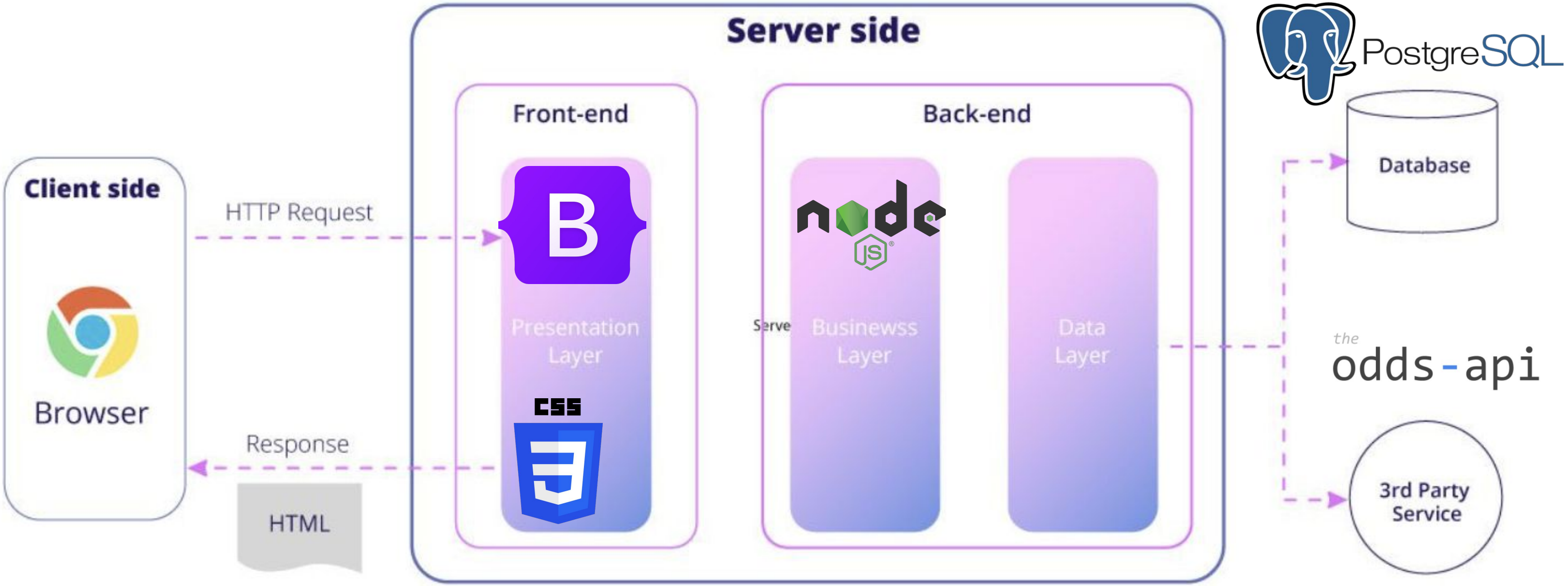
Continuous Integration and Deployment (CI/CD):

- Implemented CI/CD pipelines with GitHub Actions.
- Automated testing and deployment.
- Delivered updates quickly and confidently.

Scrum Framework:

- Adopted Scrum for project management.
- Sprint planning and goal setting.
- Regular reviews and retrospectives for improvement.

Architecture Diagram



Challenges

- Displaying Images in handlebars
- Finding a good API with sports betting data
- Docker
- Testing routes
- Bcrypt
- Routing

Overcoming Challenges

Displaying Images in Handlebars:

- Research: Yuri and Riley researched handlebars on stackoverflow for displaying images in Handlebars templates.
- Implementation: After exploring different approaches, Yuri and Riley implemented a solution by passing array of url and calling it.

Finding a Good API with Sports Betting Data:

- Research: Scott researched compatible APIs that provide sports betting data.
- Evaluation: Our team evaluated APIs based on criteria such as data coverage, reliability, and ease of integration.
- We then integrated it into our project, following the API documentation and guidelines provided.
- Testing: We thoroughly tested the API integration to ensure that it retrieves and displays sports betting data accurately within our application.

Overcoming Challenges

Docker:

- Learning: We familiarize ourselves with Docker and containerization principles through tutorials, documentation, and hands-on practice.
- Setup: Our team set up Docker environments for development, testing, and production, configuring Dockerfiles and docker-compose.yml files as needed.
- Troubleshooting: When encountering issues, we tackled Docker-related problems by referring to documentation, forums, and seeking assistance from peers or online communities.

Bcrypt:

-
- Integration: We integrated Bcrypt for password hashing and salting into the authentication system of our application.
- Configuration: We configured Bcrypt parameters to balance security and performance, considering factors like cost factor and hash length.
- Testing: We thoroughly tested the password hashing functionality to ensure that passwords are securely stored and validated during authentication processes.
-

Overcoming Challenges

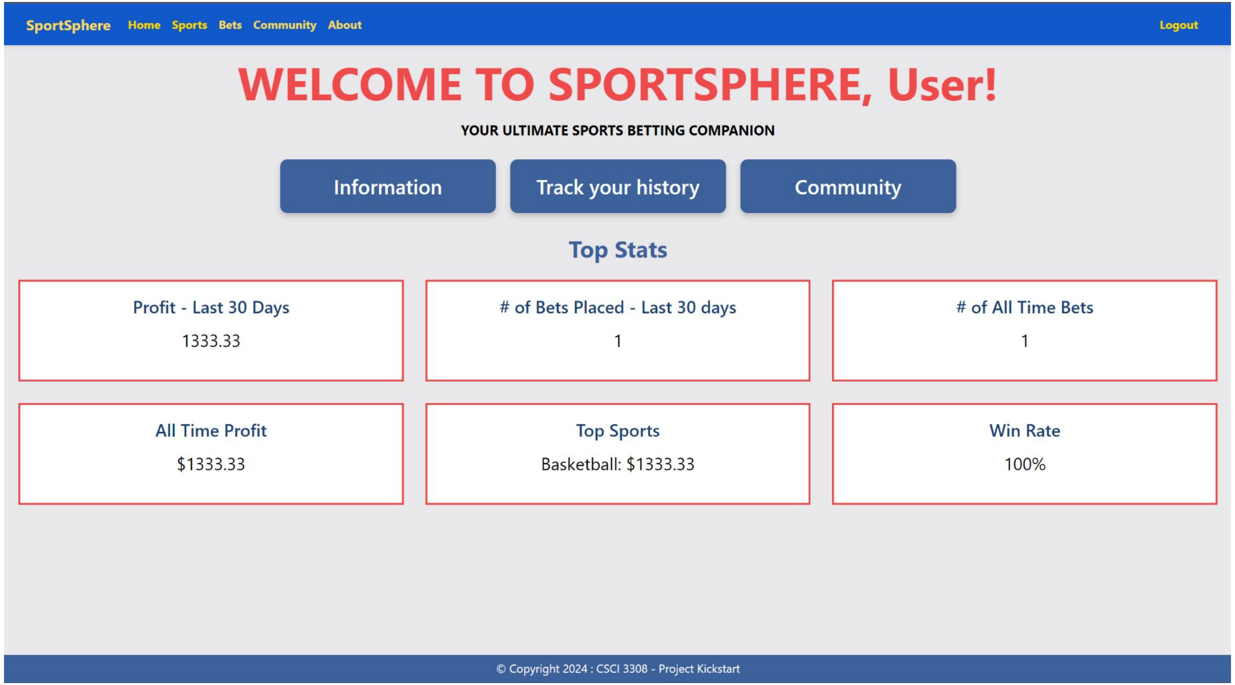
Routing:

- Design: We designed clear and intuitive routing structures for the application, mapping URLs to corresponding controller actions or middleware functions.
- Implementation: We implemented routing using frameworks like Express.js, defining routes for different HTTP methods and URL patterns.
- Middleware: To handle authentication, authorization, and error handling, we implemented middleware functions that intercept and process incoming requests.
- Testing: We tested routing functionality through manual testing and automated tests to ensure that requests are routed correctly and handled appropriately.

Future Scope/Enhancements

- Expand Sports Coverage: Plans to include more sports, including minor ones, to cater to a broader audience.
- Improve enhancements such as personalized recommendations, real-time notifications, or social media integration.
- Advanced Analytics: Integrate analytics features to provide users with insights into their betting patterns and trends.
- Ability to edit/delete previous tracked bets

Demo



WELCOME TO SPORTSPHERE, User!

YOUR ULTIMATE SPORTS BETTING COMPANION

Top Stats

Profit - Last 30 Days

1333.33

# of Bets Placed - Last 30 days

1

# of All Time Bets

1

All Time Profit

$1333.33

Top Sports

Basketball: $1333.33

Win Rate

100%

© Copyright 2024 : CSCI 3308 - Project Kickstart