Team 18: SpongeScript SquarePants

GitHub Repo: https://github.com/cse110-sp25-group18/warmup-exercise.git

YouTube Link: https://youtu.be/zvvFUsIAclg

SWOT Analysis

> Utilizing generative AI for grammar correction and text polishing

Team

Strengths

- Team members are experienced and possess relevant skills.
- Some members showed strong enthusiasm and initiative toward project tasks.

Weaknesses

- Lack of timely communication among members.
- Team members were not familiar with each other at the beginning.
- Uneven participation; some members were inactive.
- Unclear division of responsibilities.
- Some team members may feel lost or unsure about their roles and tasks, leading to a sense of uselessness.

Opportunities

- Multiple team collaboration models were explored.
- Shifted from group-based task allocation to a task board system, encouraging flexibility and autonomy.

    - Promoted higher motivation due to voluntary task picking.
    - Improved task quality by allowing members to choose based on their strengths.
- Opportunity to establish a fair contribution tracking system to motivate involvement and accountability.

Threats

- Progress could be blocked if a key task is delayed by one person.
- Online collaboration limited team cohesion and efficiency.
- Lack of standardized practices led to frequent code merge conflicts.

Tech

Strengths

- **Clear Architecture**: Uses MVC pattern with separated modules. `GameController` handles core logic, while components are well-scoped and reusable.
- **Consistent Visual Style**: SpongeBob-themed UI with matching fonts, buttons, and colors. Layout and interactions are mostly complete.
- **Modern Technologies**: Uses ES6 modules, Web Components, and Shadow DOM. Modular CSS is applied to organize styles.
- **Good Code Structure**: Uses object-oriented design. Patterns like Factory are applied in `CardFactory`, making logic clean and extendable.

Weaknesses

- **Overloaded Controller**: `GameController.js` has over 700 lines of code and handles UI, logic, and animations all together.
- **Complex Event Handling**: Event logic is deeply nested. Some modules are tightly coupled, making debugging difficult.
- **Scattered State Management**: Game states and button states are updated in multiple places, making flow hard to follow.

Opportunities

- **Component-Based Development**: Web Components can be further split into smaller units for better reuse and flexibility.
- **Centralized State Handling**: A unified system can be introduced to manage game flow and button states more clearly.
- **Testing Potential**: The modular structure is suitable for adding unit tests and improving code reliability.
- **UI Scaling**: Current theme structure supports more themes or responsive designs in the future.

Threats

- **Style Drift Risk**: Without a fixed color system, the visual theme may become inconsistent during future updates.
- **Maintenance Difficulty**: As logic and features grow, the controller and tightly coupled parts may become hard to manage.
- **Browser Compatibility**: Shadow DOM and some ES6 features may need polyfills in older browsers.
- **Performance Concerns**: Heavy DOM operations and animations could cause performance issues as complexity increases.

Tool

Strength

- **Use of AI:** fast development with AI, better understanding with others' code,
- **GitHub Issues:** increased productivity, ease of collaboration and task management
- **Figma:** app used to make our designs easily to create a user-friendly and appealing UI
- **Developer Tools:** very helpful for debugging and testing
- **Slack Task Tracker:** we used a task tracking canvas in our Slack channel to assign tasks which are not code related such as testing and designing, and

keep track of progress in one platform, which was helpful in streamlining communications and managing tasks effectively.

- Prettier: we use prettier with github action to ensure there is no code formatting difference in reviewing PR

Weakness

- **Use of AI:** Needs a careful code review and prompt engineering to make sure we are not using faulty or unnecessary code.
- **GitHub Issues:** initially had trouble using issues for task tracking given the different experience levels and familiarity with GitHub tools in a team.
- **Slack Task Tracker:** The use of this new platform for task tracking was confusing at first and required timely communication and engagement from team members

---

Opportunity

- **Use of AI:** could be used as a guide to expand our Blackjack game functionalities in the future
- **Developer Tools:** with more experience and practice using developer tools, we could have a more effective and streamlined debugging and testing process

---

Threat

- **Use of AI:** Can introduce bugs or faulty logic into our implementation if it is not reviewed carefully, understood, and adjusted to fit our needs.
- **GitHub Workflow:** If the development workflow and member restrictions are not handled properly, this could cause confusion and conflict with merging and version control

▼ Review of the process

▼ TimeLine

Tuesday

We set up our repository and completed the initial configurations. This included setting up an issue template, a pull request template, branch protection rules, member privileges, commit standards, and a development workflow.

Thursday

We experimented with various methods of team collaboration. Initially, we formed small groups, with each group leader assigning tasks within their team. However, we soon encountered several issues with this approach:

- Uneven workload distribution among groups
- Difficulty in evenly breaking down large tasks across teams
- High communication overhead between groups
- Limited understanding of individual members' skillsets, which hindered optimal task assignment

---

Wednesday

In response to these challenges, we adopted a task board model. Tasks were posted on a shared board, and team members could freely claim the ones they wanted to work on. This approach offered several advantages:

- **Increased motivation**: since members chose tasks voluntarily, they were more committed to completing them
- **Improved quality control**: individuals selected tasks based on their own capabilities, leading to better outcomes

However, this method also had its drawbacks:

- Some tasks remained unclaimed for extended periods (in such cases, group leaders would intervene)
- A few members did not participate in any tasks at all

To address the issue of non-participation, we considered introducing a penalty system. As a result, we decided to track each member's contributions at the end of the project to identify those who had made no effort.

Friday

Began Stage 1.

Saturday

Worked on Stage 2 and Stage 3.

Sunday

Completed the final report and made some improvements.

Contribution evaluating

▼ Miaoxin Chen

- commits: 24
- Lead developer. Massive code contribution. Ensure develop workflow. Doing Code review and test.
- work output:
  - **chore: update basic file structure**

    feat(html): add a tough code frame

    feat(css,script): add initial css and script to debug

    fix: github pages css file load error

    feat: add some function to improve game experience

    feat: exclude hand cards from shuffle

    fix: flip the card in hand

    BREAKING CHANGE: refactor: card.js to mvc

    fix: card click to flip

    feat!: working black jack

    BREAKING: refactor: main.js refactored to mvc of game(system,player)
  - **doc: add REAME.md**

chore: update pr template

chore: update issue templates

▼ Thomas Rocha

- commits: 9
- Improved card overlap, shuffle animations, and gameplay UI enhancements.
- Work Output
    - **fix: 🐛 fixed bug #10**

      feat: deck representation for > 1 cards and shuffle animation

      fix: card remains flipped after shuffling

      feat: add card dealing and improve animations for shuffling

      feat: cards overlap instead of creating a new line

      fix: disable buttons during animations and prevent premature round restarts
      feat: card deck size stays constant

      feat: added betting system

      feat: updated buttons to work at proper times

▼ Ryan Soe

- commits: 4
- Doing core logic (deck, dealer), animations, and critical bug fixes.
- WorkOutput
    - **feat: deck representation for > 1 cards and shuffle animation**

      fix: Deck disapears/is renderer by mistake when shuffle

      feat: created a "dealer" hand and added the ability to deal to both player and dealer

feat!: working black jack

▼ Daniel Bonkowsky

- commits: 3
- Added rules page and fixed HTML structure, improving user experience.
- WorkOutput
    - **fix: <html> tag closed before <body>**

      feat: added a blackjack rules page (#44)
      Deprecated Card Flip

▼ Kayla Phillips

- commits: 2
- Contributed alternative styles and layout matching the team design vision.
- WorkOutput
    - **style: alternative style based on initial design**

      style: 🎨 added style to rules page

▼ Angel Cortez Gutierrez

- Lead Team, doing communication work. Assigning tasks and checking up with team members to review progress. Testing.

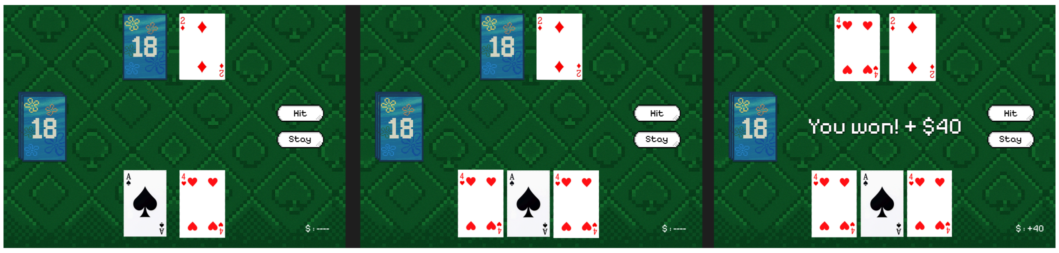
▼ Max Kim

- commits: 1
- Contributed valuable test code for card flipping, improving code quality.
- WorkOutput:
    - **feat: Added test for card flip and functionality of card flip**

▼ Domonick Marshall

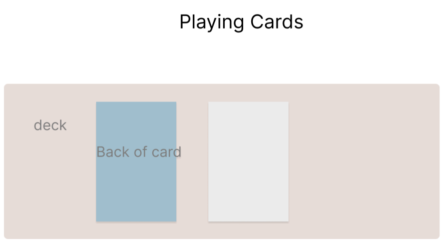
- commits: 2

- Designed card styles and initial blackjack prototype
- WorkOutput:



▼ Ashley Zhou

- commits: 2
- add CSS styling for rules page; helped establish initial visual design.
- WorkOutput:
    - **style: css stylesheet for rules and background**



▼ Justin Nguyen

- commits: 1
- add CSS styling for main page
- WorkOutput: **feat: changed css for card background to match example in design. also changed background of entire website to fit the theme. (#16)**

▼ Alan Diaz

- made video to show app functionalities.
- communicated health-related circumstances that affected his contribution to the code

▼ Ashton Bothun

- made a video showing functionalities and features of blackjack game.