## **San Francisco State University**

## **CSC 648 - 848**

## **Milestone 0 Submission Form**

**Section 01 Team 02**

Nadir Ali

Ryan Flannery

Kayla Maa

Joyce Fu

Andrew Dahlstrom

Aidan Bayer-Calvert

Jason Avina

Below is a list of the technologies used in Team's software stack:

* Sever Host: AWS EC2 for hosting both frontend and backend
* Operating System: Amazon Linux 2
* Database: AWS RDS for MySQL database management.
* Web Server: Vercel
* Front-end Framework : React, ShadCN, Tailwind
* Server-Side Language: Javascript, Typescript
* Web Application Framework : Express
* Back-end: Express.js, Node.js, Python
* Additional Technologies: Git and ssh
* IDE: VS Code
* API Integration: IGDB or Amazon API for fetching video game data

| Item | Credentials |
| --- | --- |
| Website URL | https://csc648-01-fa24-team02.vercel.app/ |
| Cloud VM instance | Our Team set up an Amazon EC2 instance to serve the webpage's backend. To access this backend, I have sent a secure email with an ec2 key pair sent to both the professor and TA  -download and unzip key pair file note file path of key pair(/file/path/of/  -change permissions on key pair: chmod 400 /file/path/of/ratingsbackendkey.pem  -access ec2 from terminal: ssh -i /file/path/of/ratingsbackendkey.pem ec2-user@54.200.162.255 |
| Database URL | Our team's database is hosted on Amazon AWS console using an RDS instance. There are multiple ways to connect to a database for example MySQL workbench, command prompt, terminal or using javascript code, as a few examples. In all cases here is the necessary information to connect to our database:  database name: 'ratingsdb'  host: 'ratingsdb.c10o6s8wguyo.us-west-2.rds.amazonaws.com'  user: 'ratingsuser'  password: 'ratingspass'  Using this info anyone can access the remote database and edit it. For example, using terminal with mysql installed, type in the bash shell(not inside the mysql client and no quotes):  'mysql -h ratingsdb.c10o6s8wguyo.us-west-2.rds.amazonaws.com -u ratingsuser -p'  It would then prompt for a password which is: 'ratingspass'  This allows all privileges to the user to view and edit the database. The database has a single table called 'users', each row is a user with a userid, name and password for the login page. To inspect this type:  'use ratingsdb;' 'select \* from users;' |

Team Meeting Schedule:

* Mondays at 6:30 PM in person
* Wednesdays at 8:00 PM on zoom

Communication Channel:

* Discord

Software Stack Knowledge Survey (Scale 0 - 5 with 5 being expert):

| Name | AWS | Express | React | Node.js | Vercel | Flask - py | SQL |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Jason Avina | 3 | 1 | 1 | 1 | 0 | 3 | 4 |
| Kayla Maa | 0 | 0 | 1 | 1 | 0 | 0 | 2 |
| Ryan Flannery | 0 | 1 | 2 | 2 | 0 | 0 | 1 |
| Aidan Bayer-Calvert | 1 | 3 | 2 | 3 | 1 | 0 | 1 |
| Nadir Ali | 1 | 0 | 3 | 0 | 0 | 0 | 1 |
| Joyce Fu | 0 | 0 | 0 | 1 | 0 | 0 | 2 |
| Andrew Dahlstrom | 3 | 0 | 0 | 2 | 0 | 2 | 3 |

Study Plan Summary:

| 9/18 | 9/23 | 9/30 | 10/7 | 10/14 | 10/21 | 10/28 | 11/4 | 11/11 | 11/18 |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| M0 DUE | M1 finish goal | M2 finish goal | M3 finish goal |  | M4 finish goal |  | M5 finish goal |  | FINAL PROJ DUE |
| learn: how to load project, how to make changes, how to clone/pull/push to github, how to interact with database | learn: what react is, what all the files in project are, how to write javascript and react, and push changes, understand vercel | learn: sql for connected database, http protocols to send + receive data to and from client and server using requests | learn: how use react and javascript to get search results and built a UI, tailwind, express and other front end frameworks | same  -> | learn: how to simply replicate previous work for any other databases and http protocols necessary | same -> | Ensure that all compon-ents desired are included and the rough draft of the project is doing basically what we want | same -> | learn: how to put the finishing touches on the project |

Based on our familiarity, we set up the following detailed 4 week study plan.

Front-End Developers (React, CSS, UI Design): Kayla, Nadir, Aidan, Jason, Andrew

#### **Week 1: Introduction to React and Basic Components**

* **Objectives**:
  1. Learn the basics of React and how to build components.
  2. Understand JSX, props, and state.
* **Key Topics**:
  1. React Components, JSX, Props, State.
  2. Functional components and hooks.
* **Resources**:
  1. **React Official Tutorial**: This covers the basics of React, component creation, and state management.
  2. **React Docs - Main Concepts**: Explore React's core concepts with examples.

#### **Week 2: Routing and Front-End Design (React Router, CSS/Tailwind)**

* **Objectives**:
  1. Implement navigation with React Router.
  2. Style components using CSS and Tailwind.
* **Key Topics**:
  1. React Router for handling routes (Login, Profile, Game List, Game Details).
  2. Using Tailwind CSS for responsive design and utility-first CSS.
* **Resources**:
  1. **React Router Tutorial**: Learn how to implement routing in a React app.
  2. [**Tailwind CSS Crash Course**](https://www.youtube.com/watch?v=UBOj6rqRUME): A full crash course on using Tailwind CSS to style your React app.

#### **Week 3: Form Handling and State Management (User Registration, Login, Reviews)**

* **Objectives**:
  1. Handle forms in React for user input (registration, login, reviews).
  2. Understand how to manage state and interact with back-end APIs.
* **Key Topics**:
  1. Controlled forms and input handling.
  2. Using useState and useEffect for managing form data.
* **Resources**:
  1. **Handling Forms in React**: Learn how to handle forms in React, including controlled and uncontrolled components.
  2. **React State Management Tutorial**: A guide to managing state in React applications.

#### **Week 4: API Integration and Data Fetching**

* **Objectives**:
  1. Learn how to fetch data from the back-end API and display it in the front-end.
  2. Work on displaying the list of video games and user reviews.
* **Key Topics**:
  1. Using fetch or Axios to get data from the backend.
  2. Handling API responses and rendering data in components.
* **Resources**:
  1. **React API Calls with Fetch**: This guide covers making API calls in React using fetch.
  2. **Axios HTTP Client**: A comprehensive guide to making HTTP requests with Axios.

Back-End Developers (Node.js, Express, MySQL): Ryan, Aidan, Jason, Joyce, Andrew

#### **Week 1: Introduction to Node.js and Express.js**

* **Objectives**:
  1. Get familiar with Node.js and Express.js.
  2. Learn how to set up a basic Express server.
* **Key Topics**:
  1. Node.js setup, creating a basic Express server.
  2. Handling routes and middleware in Express.
* **Resources**:
  1. **Node.js Official Docs**: A basic guide to understanding and setting up Node.js.
  2. [**Express.js Crash Course**](https://www.youtube.com/watch?v=L72fhGm1tfE): A YouTube tutorial covering Express.js setup and routing.

#### **Week 2: Database Design and MySQL Integration**

* **Objectives**:
  1. Learn how to set up a MySQL database and interact with it using Express.js.
  2. Understand basic database design for user profiles, games, reviews, and ratings.
* **Key Topics**:
  1. MySQL setup, connecting Node.js with MySQL.
  2. Writing queries for user data, games, and reviews.
* **Resources**:
  1. [**MySQL Crash Course**](https://www.youtube.com/watch?v=7S_tz1z_5bA): A YouTube tutorial on MySQL basics and connecting to Node.js.
  2. **Node.js and MySQL Tutorial**: Learn how to perform CRUD operations in Node.js with MySQL.

#### **Week 3: Authentication and JWT**

* **Objectives**:
  1. Implement user registration and login with secure authentication.
  2. Learn how to use JWT (JSON Web Token) for session management.
* **Key Topics**:
  1. User registration, password hashing (bcrypt), and JWT for authentication.
  2. Securing routes with authentication middleware.
* **Resources**:
  1. **JWT Authentication in Node.js**: A guide on implementing JWT-based authentication with Express.js.
  2. [**Bcrypt for Password Hashing**](https://www.npmjs.com/package/bcrypt): Official documentation for using bcrypt to hash passwords.

#### **Week 4: API Development and Testing**

* **Objectives**:
  1. Develop RESTful APIs to handle user profiles, game data, reviews, and ratings.
  2. Write test cases to validate the APIs.
* **Key Topics**:
  1. Building REST APIs in Express.js and handling requests/responses.
  2. Writing unit and integration tests for the APIs.
* **Resources**:
  1. [**RESTful API with Express**](https://developer.mozilla.org/en-US/docs/Learn/Server-side/Express_Nodejs/routes): A tutorial on building a RESTful API with Express.
  2. **Jest and Supertest for API Testing**: Learn how to write tests for Node.js APIs using Jest and Supertest.