**Production Incident Management**

**Team #26**

**Zackary Finer**

**Antoine Ngu**

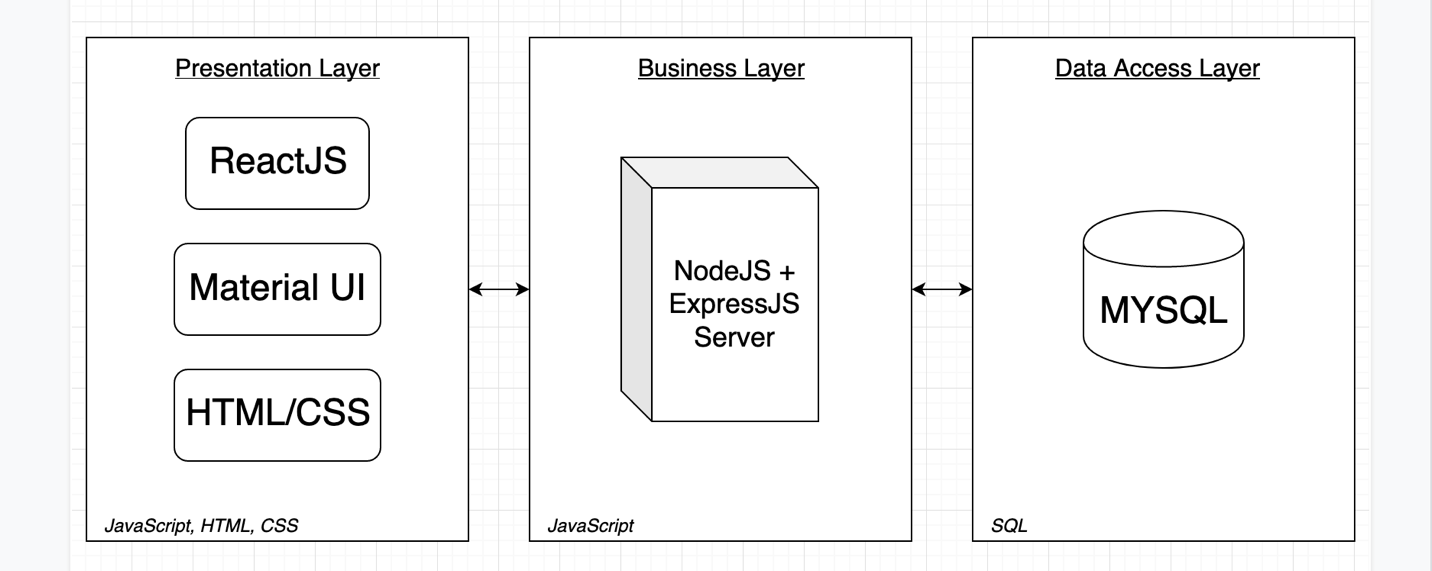
**Dale Christian Seen**

**Project Overview**

Our database management service will streamline the process of collecting, monitoring, and analyzing production incidents.

Any software company will need to manage production incidents. On top of management of these production incidents, the corporation will also need to gather analytics on any production incidents that have occurred (such as what systems are usually effected, and what possible sources of these production incidents are) so they can correctly direct developers to resolve the source of these issues.

**System Environment**



The presentation layer will consist of the frontend and UI frameworks/libraries used to render the GUI that will be presented to the user. More information on the framework/libraries and the reasons as to why the team chose to add them to the environment can be found in the Non-functional issues section. The programming language written here will be JavaScript and will also include HTML/CSS.

The business layer will consist of NodeJS with ExpressJS. ExpressJS is a popular web application framework that is used by many new applications for their server-side web programming. It runs on the NodeJS platform and will be used to connect the database and frontend together. All the business logic of the app will be written here. The programming language used for this layer will be JavaScript.

The data access layer will consist of the MySQL database as is required for the project. The main language will be SQL. For development purposes, the team will be using a popular server-side development software called *XAMPP* without the Apache HTTP server and only the MySQL database.

**Functional Requirements**

* Allow tech-support to file production incident tickets:
  + Production incidents should include the date they were opened, what software or product was affected, what client was affected by the production issue, a description of the production incident.
* Process production incident tickets and dispatch them to the appropriate team or teams.
* Allow members of development teams to view production incident tickets, and move them along a timeline of states (opened, in progress, in verification, closed) as they resolve the issue.
* Allow members of the development teams to file a production incident resolution report once the production incident ticket has reached the “closed” state.
* Prevent un-authorized users from accessing confidential information; users must have accounts, and accounts, and accounts with certain roles can access specific functionalities:
* Provide an analytics center, which would allow an administrator or site manager to view:
  + The number of production incidents assigned to a given team.
  + The number of production incidents open for a given piece of software.
  + Mean production incident resolution time (on average, how long it takes for production incidents to be resolved.

Users and Roles:

* Developer accounts will be assigned to teams and will be able to view open tickets assigned to them and their team. Along with being able to change the state of a ticket and file an incident resolution report upon closure, they will be able to make notes when they change the state (such as any other findings while the fix was “in-progress” for the ticket).
* Tech support accounts will be able to file tickets.
* A Team Manager will encapsulate all rights associated with developer roles, with the addition of being able to assign team members to tickets.
* A Site Manager account will be change user account’s roles.
* Developers and Team Managers will have access to analytics pages on their respective teams, while Site Managers will have access to all team’s analytics data.

**Non-functional Issues:**

The team will be using Facebook's React library to build the GUI of our application. React is a popular frontend framework among web developers because it makes the task of building an app's GUI much easier and simpler by breaking the various parts of a web page into encapsulated components. Knowing this, the team decided to use this technology so that the team can spend more time on the backend implementation and interactions with the database.

Along with React, the team will also use Google's Material UI framework to make our GUI more responsive to varying screen sizes and uniform in its overall styling. Using Material UI would also make our frontend much easier to implement and thus the team can spend more time on the backend and database.

For the security of our app and its users, we will be using various libraries for hashing and salting user credentials, web tokens for keeping track of sessions and access control, and user authorization.