EECS3311 Group Project

Team: Snake

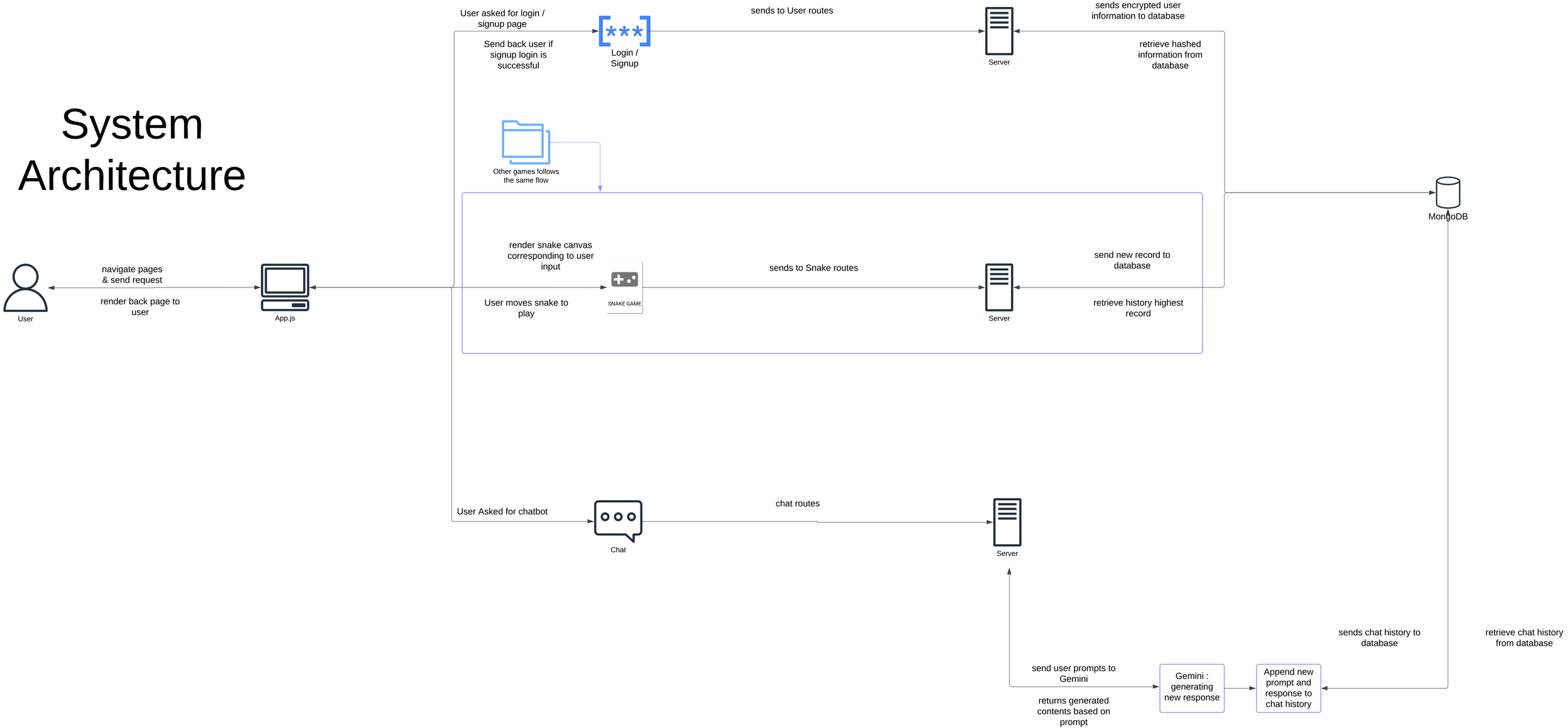
Members:

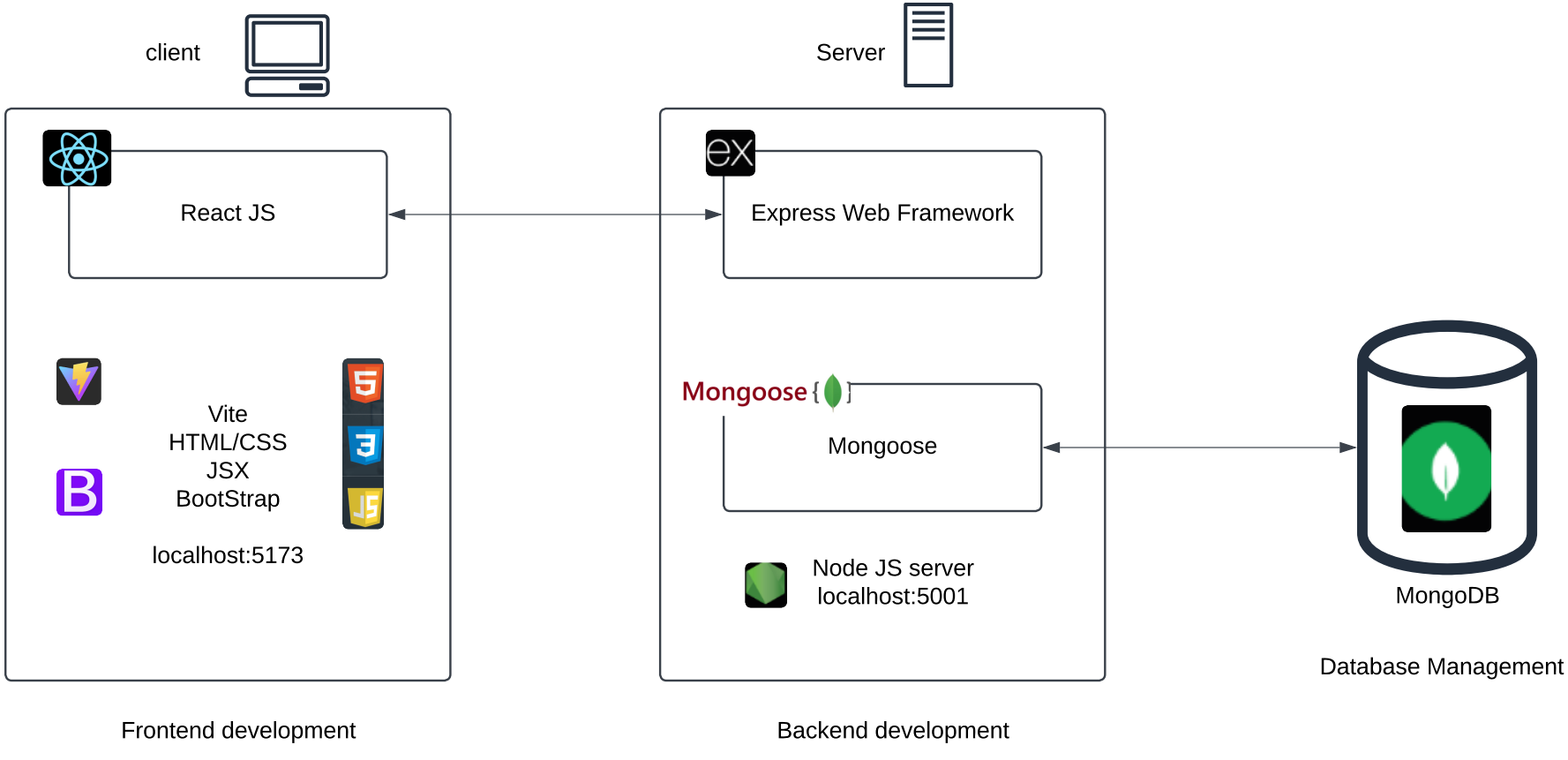
Yihan Wang | Junwei Quan | Lynne Hamd | Aryan Kalra | Pratham .

System Design

Table of Contents:

System Architecture




client
Server
React JS
Express Web Framework
Vite
HTML/CSS
JSX
BootStrap
localhost:5173
Mongoose
Mongoose
Node JS server
localhost:5001
MongoDB
Database Management
Frontend development
Backend development

Class	AboutUs.jsx
Responsibilities	- Render an "About Us" section with content. - Provide a structured view containing a heading and a paragraph to display information about the team.
Collaborators	- App.js

Class	ChatBot.jsx
Responsibilities	- Render the main chatbot interface by encapsulating `ChatContainer`. - Pass user-specific data (`userEmail`) as props to `ChatContainer`.
Collaborators	- App.js - ChatBot.css - Database

Class	Dashboard.jsx
Responsibilities	- Render user profile information, achievements, and game statistics. - Manage and display the profile image with upload functionality. - Provide a logout function, allowing the user to reset their session.
Collaborators	- App.js - Dashboard.css - Database

Class	GameList.jsx

Responsibilities	- Render a list of games based on `gameListObject` prop. - Display the list title from `listType`. - Provide navigation links for each game item.
Collaborators	- App.js - GameList.css

Class	GamePage.jsx
Responsibilities	- Render the game interface with `GameHeader`, `GameBoard`, and `GameLoader`. - Pass relevant game data and user information to child components. - Log the received data and processed user details for debugging purposes.
Collaborators	- App.js - GamePage.css - GameHeader - GameBoard - GameLoader

Class	HomePage.jsx
Responsibilities	- Render a three-column layout with placeholder content for each section. - Provide a basic responsive structure using Bootstrap classes.
Collaborators	- App.js - Footer component

Class	LoginPage.jsx

Responsibilities	- Render login and registration forms with conditional switching between them. - Handle user login and registration through API calls and manage login state. - Conditionally render the `Dashboard` component upon successful login.
Collaborators	- App.js - LoginPage.css - Backend API

CLASS NAME: Footer.jsx

Parent Classes (if any): App.jsx

Subclasses (if any):

Responsibilities:

Generate Footer section of webpage

Collaboraters:
AboutUs.jsx
ChatBox.jsx
Dashboard.jsx
GameList.jsx
GamePage.jsx
HomePage.jsx
LoginPage.jsx

CLASS NAME: GameLoader.jsx

Parent Classes (if any): GameBoard.jsx

Subclasses (if any): Snake

Responsibilities:

Load and display content based on value it recieves in gameBody

Collaboraters:
SnakeGame
WordGame
AimTrainer
ImageMemoryGame
SlotGame

CLASS NAME: GameBoard.jsx

Parent Classes (if any): GamePage.jsx

Subclasses (if any):

Responsibilities:

Central layout for game area, displays game interface

Collaboraters:

GameLoader
HelpfulTips
PreviousAttemps

CLASS NAME: HelpfulTips.jsx

Parent Classes (if any): GameBoard.jsx

Subclasses (if any):

Responsibilities:

Display guidance and tips to assist players

Collaboraters:

SnakeGame
WordGame
AimTrainer
ImageMemoryGame
SlotGame

CLASS NAME: GameHeader.jsx

Parent Classes (if any): GamePage.jsx

Subclasses (if any):

Responsibilities:

Display name of game as a heading in game interface

Collaboraters:

SnakeGame
WordGame
AimTrainer
ImageMemoryGame
SlotGame

CLASS NAME: Navbar.jsx

Parent Classes (if any): App.jsx

Subclasses (if any):

Responsibilities:

Navigate links in the application for users to access different sections.

Collaboraters:

App.jsx
AiChatbot
GameList
UserLoginIn/Out
Dashboard

CLASS NAME: PreviousAttemps.jsx

Parent Classes (if any):

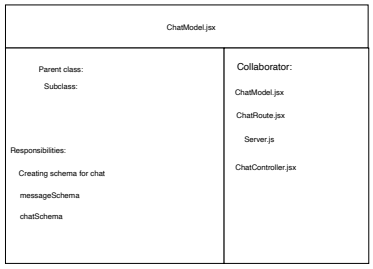
Subclasses (if any):

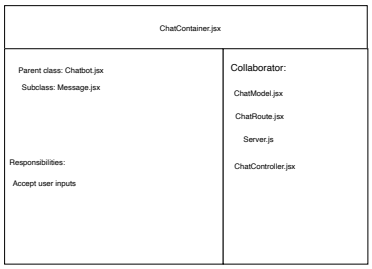
Responsibilities:

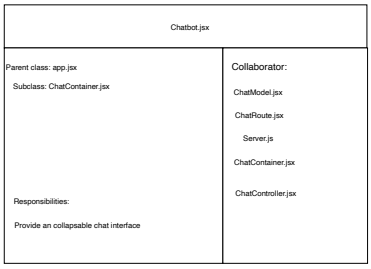
Allows users to check their past attempt

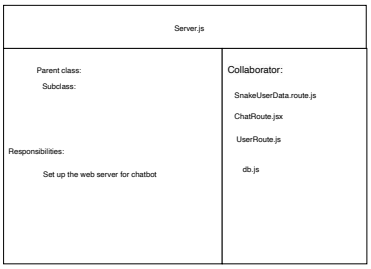
Collaboraters:

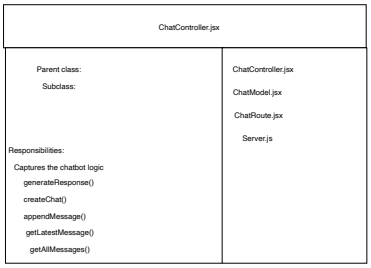
SnakeGame
WordGame
AimTrainer
ImageMemoryGame
SlotGame
User

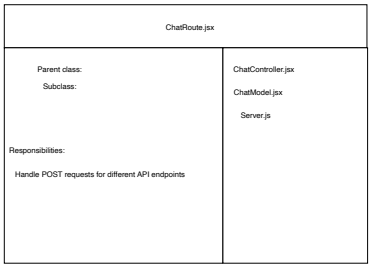

ChatModel.jsx
Parent class:
Subclass:
Responsibilities:
Creating schema for chat
messageSchema
chatSchema
Collaborator:
ChatModel.jsx
ChatRoute.jsx
Server.js
ChatController.jsx

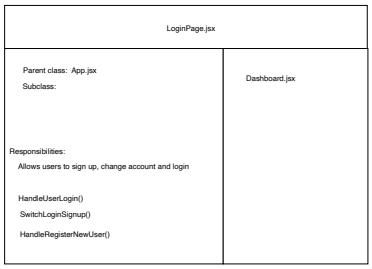

ChatContainer.jsx
Parent class: Chatbot.jsx
Subclass: Message.jsx
Responsibilities:
Accept user inputs
Collaborator:
ChatModel.jsx
ChatRoute.jsx
Server.js
ChatController.jsx

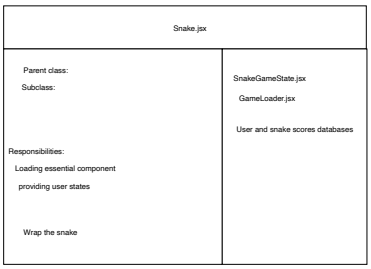

Chatbot.jsx
Parent class: app.jsx
Subclass: ChatContainer.jsx
Responsibilities:
Provide an collapsable chat interface
Collaborator:
ChatModel.jsx
ChatRoute.jsx
Server.js
ChatContainer.jsx
ChatController.jsx

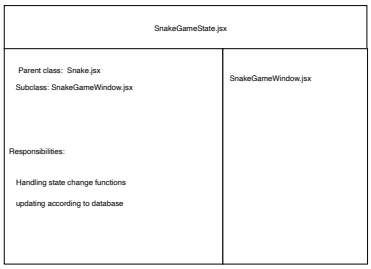

Server.js
Parent class:
Subclass:
Responsibilities:
Set up the web server for chatbot
Collaborator:
SnakeUserData.route.js
ChatRoute.jsx
UserRoute.js
db.js

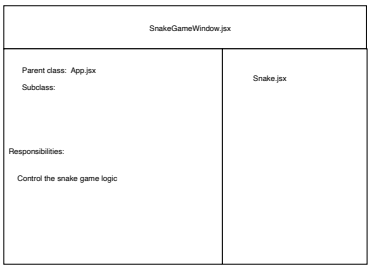

ChatController.jsx
Parent class:
Subclass:
Responsibilities:
Captures the chatbot logic
generateResponse()
createChat()
appendMessage()
getLatestMessage()
getAllMessages()
ChatController.jsx
ChatModel.jsx
ChatRoute.jsx
Server.js

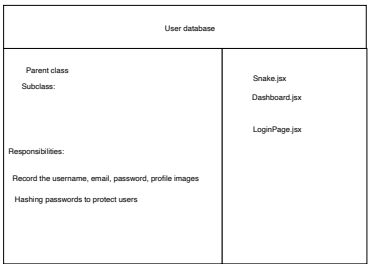

ChatRoute.jsx
Parent class:
Subclass:
Responsibilities:
Handle POST requests for different API endpoints
ChatController.jsx
ChatModel.jsx
Server.js


LoginPage.jsx
Parent class: App.jsx
Subclass:
Responsibilities:
Allows users to sign up, change account and login
HandleUserLogin()
SwitchLoginSignup()
HandleRegisterNewUser()
Dashboard.jsx


Snake.jsx
Parent class:
Subclass:
Responsibilities:
Loading essential component
providing user states
Wrap the snake
SnakeGameState.jsx
GameLoader.jsx
User and snake scores databases


SnakeGameState.jsx
Parent class: Snake.jsx
Subclass: SnakeGameWindow.jsx
Responsibilities:
Handling state change functions
updating according to database
SnakeGameWindow.jsx


SnakeGameWindow.jsx
Parent class: App.jsx
Subclass:
Responsibilities:
Control the snake game logic
Snake.jsx


User database
Parent class
Subclass:
Responsibilities:
Record the username, email, password, profile images
Hashing passwords to protect users
Snake.jsx
Dashboard.jsx
LoginPage.jsx