UNIVERSITY OF LEICESTER

School of Computing and Mathematical Sciences

CO7201 Individual Project

Final Report Template

DanceFlow: Personalized Dance Sequence Builder and Scheduler

Dinesha Bungatavula

dlb29@student.le.ac.uk

239058881

Project Supervisor: Dr Paula Severi
Principal Marker: Dr Zedong Zheng

Word Count: 5357
28/04/2025

DECLARATION
All sentences or passages quoted in this report, or computer code of any form whatsoever used and/or submitted at any stages, which are taken from other people's work have been specifically acknowledged by clear citation of the source, specifying author, work, date and page(s). Any part of my own written work, or software coding, which is substantially based upon other people's work, is duly accompanied by clear citation of the source, specifying author, work, date and page(s). I understand that failure to do this amounts to plagiarism and will be considered grounds for failure in this module and the degree examination as a whole.

Name: Dinesha Bungatavula
Date: 28/04/2025

Abstract

Dance Flow web application is an innovative platform where the users are provided with a structured and engaging interface. There are many dancers who face the problems in staying consistent and organized to develop their skills, this web application comes with the various features which includes to create, manage and track the progress of their own sessions.
The Dance Flow offers the users to explore the variety of dance moves and personalize their own sequences with the preferred dance moves and schedule them for their later practice using the calendar. This scheduling helps the users to plan their practice sessions according to their convenience.
Progress tracking is one of the main functionalities of this system, which tracks the user's activity and their performance. Users can see the progress of their dance sessions and set the new goals according to their performance levels and improve their dance skills. This application also incorporates an intelligent recommendation system which suggests the users with the practice sessions based on the user's activity and community space for interaction and sharing the sequences with other users. This Dance Flow web application leverages modern web technologies to provide user-friendly interface to the dancers with different skill and performance levels.

Table of Contents

1. Introduction

In the era of digital transformation, the arts and education are rapidly adapting to the interactive platforms. Dance Flow is a digital platform that has been designed to full-fill all the necessary requirements of dancers and it bridges the gap between the traditional practice methods and innovative digital platforms. The Dance Flow web application is integrated with the library of dance moves, sequence customization, calendar for scheduling and progress tracking features which allows the users to stay organized, schedule and track the progress of their dance practice sessions.

1.2 Aim

The main aim of this project is to develop a Dance Flow Web application that provides the users with intuitive and flexible platform. In this platform the users are allowed to create and customize their dance sequences. With the help of this personalized dance sequence builder users will be able to develop and refine their dancing skills. This platform is suitable for all the dancers with different skill levels. They can choose and create their own sequence according to their dancing skills. The calendar has been integrated into this platform which helps the users to schedule the dance practice sessions on their own at their convenience. Additionally, the integration of progress tracking features helps the users to monitor the progress of their dance practice sessions at any time, set new goals and focus on the areas where they need improvement. This web application makes it easier for the users to organize and finish their sessions on time.

1.3 Objectives

The main objectives of the Dance Flow Web application include

- **Secure user authentication:**

Implementing the JWT-based secure user authentication system that allows the users to create and login with their accounts securely. This system also includes the role-based access control which makes sure a clear distinction between the admins and the regular users.

- **Develop a user-friendly interface:**

Designing a clean, interactive and easy-to-navigate interface for the users to easily explore the dance moves, create and personalize the sequences of their own choice. This also focuses on providing a great experience for both the beginners and advanced users.

- **CRUD operations:**

Enabling the full CRUD (create, read, update and delete) functionalities for both the users and admin. Users can create and modify their sequences as needed whereas admin can perform these functions for the dance moves in the database which improves the flexibility of this system.

- **Integration of interactive Calendar:**

Integrating a dynamic calendar system that allows the users to schedule their own dance practice sessions and manage them according to their preferences effectively. This feature also allows the users to stay organized and set reminders for their practice sessions to maintain the consistency of their dance routines.

- **Progress tracking:**

Enabling the progress tracking system to store and visualize the user's activity and performance which helps the users to monitor their progress and set the new goals for their practice sessions. This also levels up the motivation for the users to practice their sessions.

- **Auto recommendation system:**

- Designing a community space where users can share sequences and communicate with others.

1.4 Scope and limitations

This project aims to design and develop a user-friendly and interactive web application that helps in assisting the users to organize and manage their content and practice sessions. This application mainly focuses on two areas. Firstly, custom sequence creation where the users can create their own sequence and personalize them. Secondly, scheduling and the progress tracking system where the users can schedule and track a practice session. It also has a community space for the users to share their sequences and interact with others. This application is modular and extensible for future enhancements. However, the implementation of all the core features was met partially in the final system due to the time constraints and some of these limitations are listed below,

- The auto recommendation system is currently basic, and the AI-driven recommendations were not implemented in this system due to lack of time for the development.
- The calendar for now has been designed to schedule a dance practice session, advanced functionalities like rescheduling and different time zones are not supported.
- The progress tracking is limited to scheduled sessions and their completion status, no visual indicators like graphs are implemented.

1.5 Challenges

This project has been developed by facing the several challenges which includes the role-based access control for the users and admins to have the correct access to their respective dashboards without any mismatches as this is one of the key elements of this project. Initially, implementing the secure user authentication to

protect the user data which consists of personal information and session histories has been crucial. Additionally, integrating the calendar has been the huge challenge, as this project first used the FullCalendar.js library to have the calendar in user dashboard for scheduling the sessions but later due to limitations in flexibility and adaptability it was switched to more flexible and developer friendly Tui.Calendar. This replacement needed to be performed carefully to ensure the data integrity and reinstallation of the other library.

2. Literature Review

2.1 Introduction

Dance Flow, a web application aimed to provide the dancers with optimized dance practice scheduling system and enhance the personalized and interactive learning experience of the users. This literature review explores the key aspects of this project which includes the interactive and personalized learning experience, virtual dance education, personalized scheduling system, progress tracking and community engagement for the users.

2.2 Personalized and Interactive learning

There are huge number of dance learning platforms which provide the users with pre-recorded instructional videos and structured sessions among different dance styles for dance learning, but they all lack with the personalized and interactive learning which is the main feature for the users to enhance their learning experience. Research on personalized learning highlights the importance of interactive learning experiences [1]. The increase of web-based learning has created a room for all the learning sectors including dance. This web based personalized learning helps the users to develop the skills on their own and it leads to improved learning outcomes when supported by content adaptation [2]. Interactive learning emphasizes the user engagement with the content and the environment. The interactivity in multimedia learning environment improves the learner's engagement with the content and demonstrates higher cognitive retention [3]. The integration of all the necessary features will allow the users to improve their experience with personalized and interactive learning.

2.3 Virtual Dance education

Virtual dance education refers to the digital learning platforms which have a rapid growth and evolution. Especially, during and after the COVID-19 pandemic the virtual dance education has changed as a crucial mode of dance learning for all the dancers and choreographers. Virtual dance learning platforms had a significant growth during the pandemic as all the dancers migrated to the digital

platforms [4]. These platforms provide the flexibility and accessibility to have an interactive learning environment [3]. The main core of the virtual dance learning platform is the content of the dance moves and the flexibility to access them. The online dance learning is also growing in styles and categories which includes ballet training and many others [5]. The growth in particular styles and categories will help the dancers to learn what they prefer.

2.4 Personalized Scheduling System

The use of personalized scheduling system in the web application is more important for time management and learning consistency. This always allows the users to schedule their sessions on their own without missing any sessions. This scheduling system can be done by the integration of calendar libraries such as FullCalendar.js and Tui.Calendar due to their API rich environment and real-time synchronization. This visual representation of upcoming sessions in the calendar increases the user engagement and improves their learning experience. This not only improves the usage of the system but also the adaptive learning experience of the users [6]. Moreover, the integration of personalized scheduling system will provide a potential for data collection and suggesting the optimal sessions for the users based on the user's activity. This recommendation of sessions to the users will enhance the interaction with the system [7].

2.5 Progress Tracking

Progress tracking is the feature in the digital platform that allows the users to monitor their learning progress and performance. This feature is implemented based on session history and activity of the users.This stored session history of users in database is retrieved and shown for the users by using libraries such as chart.js. This progress tracking also helps the users in continuous improvement and to stay motivated with their work [8]. By adapting this progress tracking functionality, the digital platforms offer creative and gamified incentives like badges and awards for performance [9]. These gamified incentives will level up the user's activity and performance in order to achieve them and set new goals to gain those incentives. This progress tracking will help all the levels of users to track their progress, improve the skills and level up their performance.

2.6 Community engagement

Community features such as sharing the sequences, adding comments and having chats are important for the modern learning platforms as they provide the engagement to the users with other members [10]. This communication helps them to have shared learning experiences and, the users will have the space to learn new things from other users. This shared learning helps the low skilled users to engage with the high skilled, learn and increase their level of performance.

Research shows that the digital platforms with strong social skills and easy communication will directly boost the participation and usage [11]. This shared learning experience will significantly enhance the motivation and confidence of the users [12].

3. Requirements

This section has all the necessary and specific requirements of the project. These requirements are divided into the following three categories which are essential, recommended and optional.

3.1 Essential Requirements

The essential requirements are the important ones that should be met to achieve the base of the project. The following are the essential requirements:

- User registration: The users should be able to register and login to their securely. All the user information must be stored safely, and the passwords of the users should be hashed perfectly before storing them. The admin and regular must be taken to their respective dashboards without any mismatch of accessing the data with the help of role-based access control.
- Admin role: The admin role has the ability to create, read, update and delete the dance moves that are stored in the database and these changes should reflect the user dashboard. This role is critical to keep the content up to date ensuring that the learners have access to all the resources and to manage the user accounts.
- Database: A well-structured database that stores all the information of the users, dance moves, sequences and the scheduled dance practice sessions in their respective collections.
- Sequence creation: The users will be able to create and customize their own sequence by combining multiple dance moves that are available in the database for their personalized learning. These sequences are stored in their profiles and can access them whenever they login to their accounts.
- Progress tracking: This functionality will track the progress of the users based on their activity and the sessions they have created and practiced. This will give a visual representation of the progress to the users to keep the track of their dance practice sessions.
- Community features: Users will be able to share their personalized sequences with their preferred members who are existing in this application.

3.2 Recommended Requirements

The recommended requirements are the extended functionalities of the project that need to be done. The following are the recommended requirements:

- Role-based access control: This ensures that the admin and the users are directed to their own dashboards to make sure that the admin data is not accessed by the regular users. This will have the appropriate permissions for different roles. This separation allows the system to manage and access the content securely and efficiently.
- CRUD Operations:

The CRUD operations are essential for both the admin and user to manage and interact with the content of the application effectively.

  - Admin role: The admin role has the CRUD operations to make the changes in the dance moves and makes sure the content of the dance moves is up to date.
  - User role: The CRUD operations for the user role is to create, read, update and delete their sequence, as they only have the access to their own personalized sequence.

- Advanced features: The advanced search features like keywords and filter-based search options for easy access to the content. It will not only enhance the user experience but also saves time and makes navigation efficient mainly for the users with specified goals and preferences.
- Calendar Integration: The integration of calendar into the system will allow the users to schedule their dance practice sessions with their preferred sequence for their later practice. This will visually display the upcoming sessions of the users and help them to be prepared for that. This also allows the user to set remainders for their dance practice sessions without missing any of them.
- Notifications: The system will be automatically sending the notifications to the user about their scheduled sessions as a remainder before the start date of the session.
- Auto recommendations: The system tracks the user's activity and then checks the sessions history, considers if any gap between the sessions of the users and sends the recommended sessions to the users based on them.

3.3 Optional Requirements

The Optional requirements are the additional requirements that improve the project. The following are the optional requirements:

- Profile customization: The users will be able to customize and update their profile and details whenever they want, including the name and password. They can also reset their password before logging into their account if they forget it.

- Offline mode: Users have the option to download their sequences, if they want to practice them later which is more flexible.
- Voice Commands: Voice commands for the dance practice sessions like Start and Stop if the user wants to do in the middle of the dance practice session.

4. Technical Specifications

Frontend Technologies:

- HTML5: Used for building a structured, easily accessible and visually clear pages which enhances the navigation experience for both admin and user interfaces.
- CSS: Used for styling of the pages and to implement visually attractive and responsive web pages with interactive elements such as forms, buttons and modals which enhance the user interactivity and usability.
- JavaScript: Used for managing the dynamic web page behaviours such as event-handling, DOM manipulations, content loading and real-time updates.

Backend Technologies:

- Node.js: Used to run the server-side JavaScript and it ensures efficient server-side operations and seamless client-server interaction.
- Express.js: The framework that has been used to serve the static files, it simplifies the API endpoint creation, routing and middleware handling.

Database:

- MongoDB: Used for storing the data of users, dance moves, sequences and sessions. It has been chosen for it's scalability and flexibility as it supports the complex document structures.
- Mongoose: Used for the connection between the backend node.js and the database MongoDB.

Authentication:

- JWT Authentication: Used for secure user authentication with token-based verification and ensuring the security and data privacy.
- bcryptjs: Used for secured password hashing and encrypting them safely.

Libraries:

- Nodemailer: A node.js module that will send the emails easily. It has been integrated to send the email notifications and for the forgot password functionality.

- Multer: A middleware for node.js and express.js for easy handling of fie uploads. This will handle all the image files uploaded by the admin for the dance moves.
- tui.calendar: It has been used to have an interactive calendar for the user to schedule their dance practice sessions and set the remainders.
- Choices.js: A lightweight JavaScript library that allows the advanced searchable dropdowns and multiple selections. It has been used to have the searchable dropdown of dance moves for users while creating a sequence.

5. System Design and Architecture

The Dance Flow web application adopts a layered, modular architecture which has Model-View-Control (MVC) design pattern. This ensures the maintainability and scalability of the system. It consists of three main layers Data layer, Presentation layer, Application layer and are supported by the backend logic and APIs for the client-server communication.

5.1 Data Layer (Model)

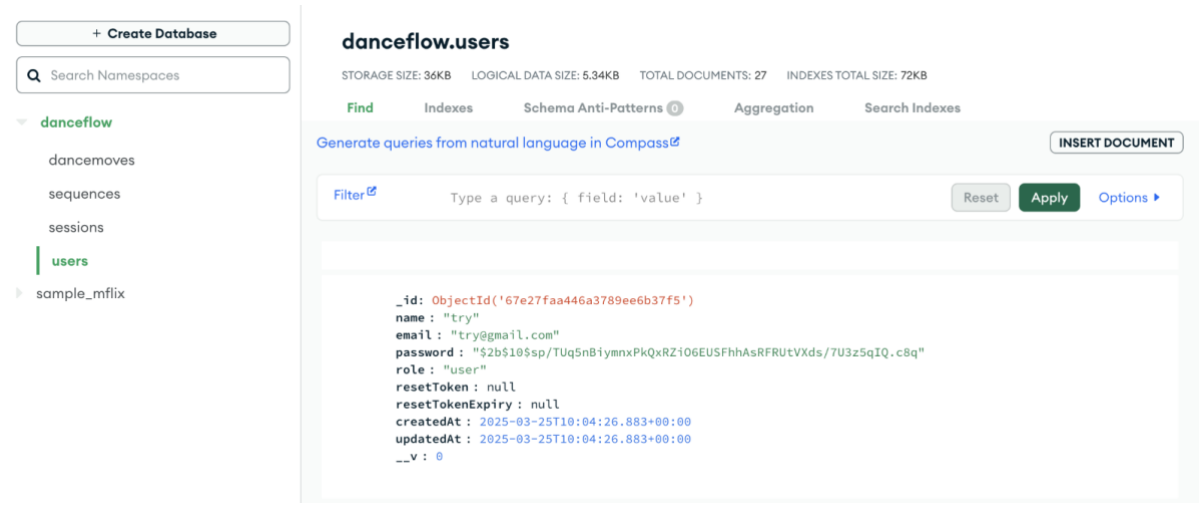
The data layer consists of all the models required for the Dance Flow Application and this is implemented using Mongoose which defines the clear structure for all the core entities in the application. This ensures the consistency, querying and easy validation across all the database collections. This data layer uses MongoDB a scalable, document-oriented NoSQL database. The data layer consists of the following models:

User Model (User.js):

This schema defines the structure for both the user and admin with the following fields

- name: the full name of the user added by them.
- email: the email id of the user which should be unique and used by the user to login.
- password: The password to be hashed by using bycrypt.js for the security.
- role: This defines the access and permissions to all the user whether it is admin or user.
- resetToken/resetTokenExpiry: These are used for the password reset of the users.
- timestamps: The time stamps are for getting the created and updated data of the user accounts.

```
models > JS User.js > ...
1   const mongoose = require("mongoose");
2
3   const UserSchema = new mongoose.Schema(
4       {
5           name: String,
6           email: { type: String, unique: true },
7           password: String,
8           role: { type: String, enum: ["user", "admin"], default: "user" },
9           resetToken: { type: String, default: null },
10          resetTokenExpiry: { type: Date, default: null }
11      },
12      { timestamps: true },
13      { collection: 'users' }
14
15
16  );
17
18  module.exports = mongoose.model("User", UserSchema);
19
```



Dance Move Model (DanceMove.js):

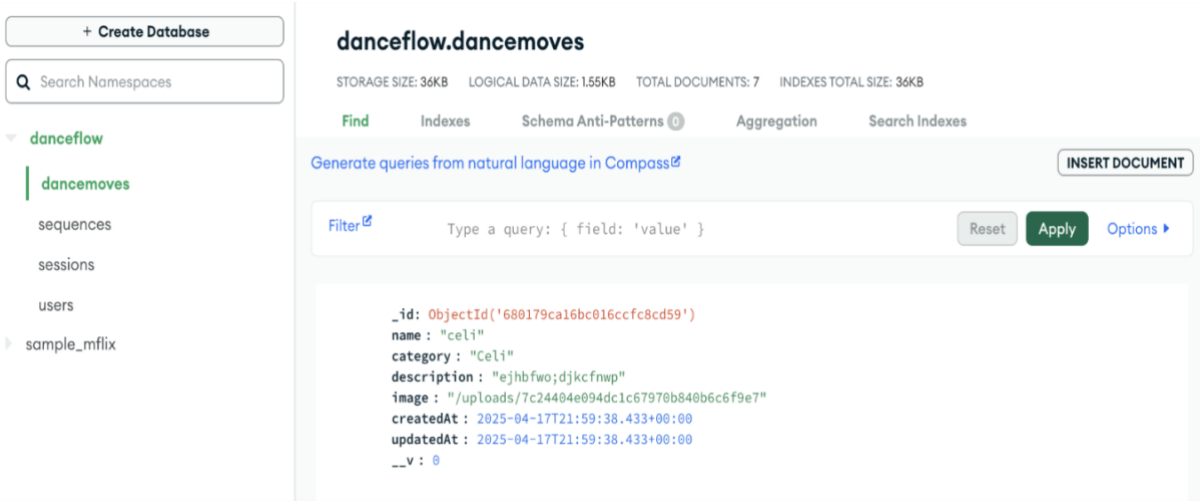
This schema manages all the data of dance moves content added by the admin in the admin dashboard. This stores the data of dance moves including the following fields:

- name: The name of the dance move.
- category: the category or the type of dance move.
- description: to provide a textual explanation and describe the dance moves.
- image: to store the uploaded image of a dance move and present it to the users.

- createdBy: this refers to the admin that has created the dance move.
- timestamps: this is for tracking the updates of the dance moves content.

models > JS DanceMove.js > ...

```
1   const mongoose = require("mongoose");
2
3   const DanceMoveSchema = new mongoose.Schema(
4       {
5         name: { type: String, required: true },
6         category: { type: String, required: true },
7         description: { type: String, required: true },
8         image: { type: String },
9         createdBy: { type: mongoose.Schema.Types.ObjectId, ref: "User" } // Reference to Admin
10      },
11
12      { timestamps: true },
13      { collection: 'dancemoves' }
14  );
15
16  module.exports = mongoose.model("DanceMove", DanceMoveSchema);
17
```

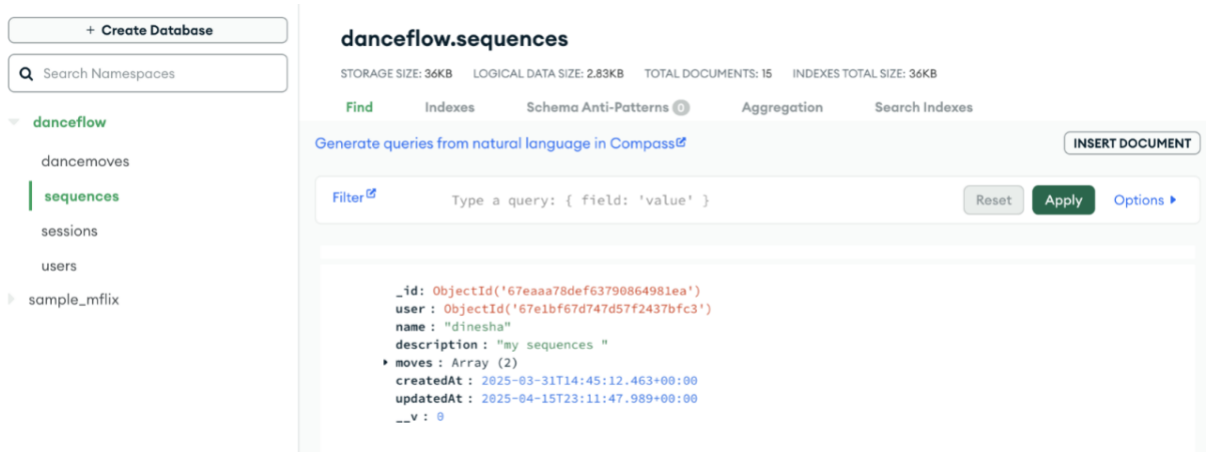


Sequence Model (Sequence.js):

This model represents the custom sequence created by the user and stores it with the respective user id. This has the following fields in it:

- name: The name of the sequence created by the user
- description: the description of the sequence that has been added while creating the sequence.
- moves: It is an array of the dance moves that are selected by the user from the existing dance moves to create a sequence.
- timestamps: this to track when the sequence is created and updated.

```
models > JS Sequence.js > ...
1   const mongoose = require("mongoose");
2
3   const sequenceSchema = new mongoose.Schema({
4     user: {
5       type: mongoose.Schema.Types.ObjectId,
6       ref: "User",
7       required: true,
8     },
9     name: { type: String, required: true },
10    description: { type: String, required: true },
11    moves: [{type: String}],
12  },
13  { timestamps: true }
14  );
15
16  module.exports = mongoose.model("Sequence", sequenceSchema);
17
```



Session Model (Session.js):

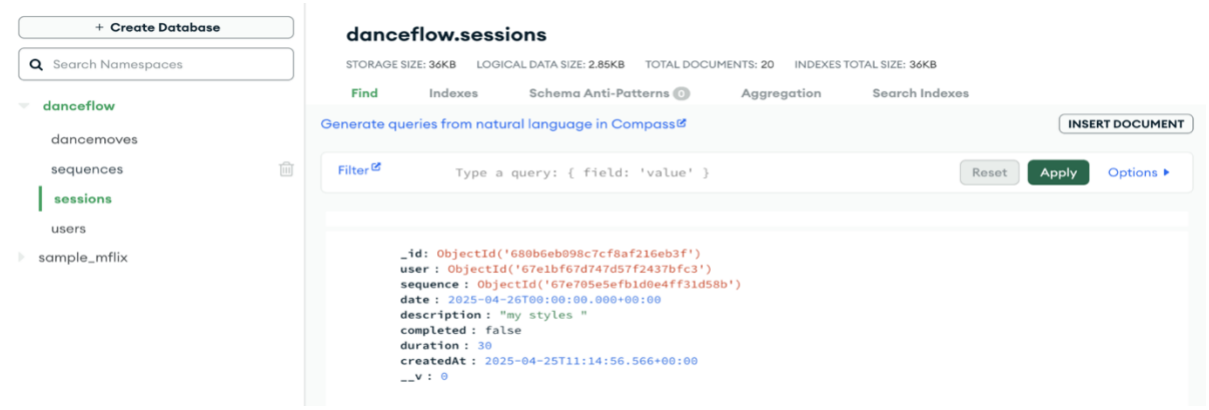
This schema is to store the individual practice sessions scheduled by the user. This has the following fields to have the session data of the users.

- user: this is to get the details of the user who created the session.
- sequence: this for linking the session with the sequence that will be practiced in that session.
- date: the date of the session when it is scheduled in the calendar.

- description: the description of the session that should be added by the user when creating a sequence.
- completed: this stores the completion status of the session.
- createdAt: this is to track when the session was created.
- duration: it is for the user to set the duration of their session; the system will set 30 min of duration by default if the user doesn't set any.

models > JS Session.js > ...

```
1   const mongoose = require("mongoose");
2
3   const sessionSchema = new mongoose.Schema(
4
5     {
6     user: { type: mongoose.Schema.Types.ObjectId, ref: "User", required: true },
7     sequence: { type: mongoose.Schema.Types.ObjectId, ref: "Sequence", required: true },
8     date: { type: Date, required: true },
9     description: { type: String },
10    completed: { type: Boolean, default: false },
11    createdAt: { type: Date, default: Date.now },
12    duration: { type: Number, default: 30 },
13  });
14
15  module.exports = mongoose.model("Session", sessionSchema);
16
```



5.2 Presentation Layer (View)

The Presentation layer is the place where user interacts directly with the system. This layer is built using HTML, styled with the responsive CSS and the JavaScript to create the user interface (UI) and to manage how the content is displayed and being interacted with the users. This layer is separated into two different roles: user and admin; with their respective custom dashboards and functionality. To get the users with their correct dashboard role-based access control is implemented.

Admin Dashboard:

The Admin Dashboard is a management interface that is tailored for the administrative tasks like managing the dance moves content that displays on the user dashboard and to monitor the user accounts. The admin dashboard is designed with the following sections:

- All Users: This section displays all the users with their name, email and role to the admin with a searchable table. When the admin clicks the "ALL USERS" section in the side navigation bar, the frontend sends the request and gets the data from the backend. This section allows the admin to monitor all the user accounts.

- Add Dance Move: This section allows the admin to add the dance moves with their name, category, description and image via Multer. Once admin fills all the fields in the "Add Dance Moves" form and uploads it, the dance move is then can viewed by the users in their dashboard.

- All Dance Moves: This is the section where the admin will be able to view all the dance moves added and this section also allows the admin to edit and delete the dance moves by searching and filtering them by category. If the admin wants to edit any of the dance moves the edit button will take to the "Edit Dance Move" form where any detail of the dance moves can be edited. The delete button allows the admin to delete the dance move.

- Edit Profile: In this section admin can update their personal details such as name and password.

- Logout: This button on the navigation bar allows the admin to log out of their account.

User Dashboard:

The User dashboard is the place where the regular users will interact with the system. This dashboard allows the users to do all their necessary activities in the process of learning the dance. In this the users can manage their profile details, explore the dance moves, build their custom and personalized sequences with the dance moves and save them for later practice. They can also schedule their own sessions with the in-built calendar available in the dashboard and keep the track of their learning progress as well. The user dashboard has the following sections to perform each functionality:

- Profile: This section will display the details of the user such as name, email and role once they are logged in.

- Dance Moves: This is section where the users will be able to explore all the dance moves by searching them with keywords and filtering them by category.

- Dance Sequences: This section displays all the sequences created by the users with a keyword and filterable search by dance move and the sequences in this section can be edited and deleted by the user. The edit button in the table will be giving the edit form after the click on it, in which the user can edit all the details of their custom and personalized sequence. The delete button in the table will delete the sequence.

- Create Sequence: The create sequence section will open the “Create New Dance Sequence” in which the user must fill the details of the sequence such as name, description and add some dance moves from the dropdown to create their own custom and personalized sequence.

- Calendar: The Calendar section helps the users to schedule their own dance practice sessions with the sequence created by them. This calendar will allow the users to mark their sessions as done in the past and the users can also edit the session created by them. Overall, the users can visually manage their own dance practice sessions in the calendar.

- My Progress: This section visually displays the progress of the user’s dance practice session in a table format. This section has the date format in which the user can search their session by entering from and to dates. They can also search their scheduled dance practice sessions with the sequences and status of the session.

- Edit Profile: In this section the user will be able to change their personal details such as name and password of their account at any time.

- Logout: This logout button on the navigation bar will allow the user to get logged out of their account.

5.3 Application Layer (Control)

The Application Layer is the main layer of the Dance Flow system. This layer processes the requests from the frontend, applies the logic and interacts with the data layer to get the requested data and sends the result to the frontend. It is built and runs on Express.js. It consists of express route handlers, authentication with JWT, role-based access control, CRUD operations, admin only controls and the server setup.

JWT Authentication:

The authentication system will ensure the secure login of the users to their accounts. This is done in the Dance Flow system like the below:

- Login Flow: When the user logs in to the application using their email and password, the details are checked by interacting with the database and creates a JWT token if the user exits. The token is then sent to the front end and stored in the local storage. Every future request will include this token in the authorization header.

- Token Verification: This is the way to keep the application secure. This will check the token every time the users make a request like sequence creation, Session requests etc., Once the front end sends token with every request that needs protection, the backend reads the token form the header and sends the data of the user based on request from the data base. If the token is invalid or expired the route denies the access.

Role-based access control:

The role-based access control will make sure that different roles have their appropriate permissions. There are two different roles in Dance Flow: Admin and User. The admin will have the access to manage the global content like the dance moves and to monitor the user accounts. The user will have the permissions to access and manage their own sequences, scheduling sessions and the profile. This Role-based access is implemented as below:

- When the user logs in to their account, the role is included in the JWT token.
- After verifying the token, the users are taken to their respective dashboards.

- If the regular user tries to access the admin dashboard, it prevents saying “Admin access only”.

```
const isAdmin = (req, res, next) => {
  if (req.user.role === "admin") {
    next();
  } else {
    return res.status(403).json({ error: "Admin access only." });
  }
};
```

CRUD Operations:

In the Dance Flow Web application, the CRUD operations are implemented for both the users and admin for different sections. The CRUD operations for admin to manage the content and for the user to manage their own practice sessions and sequences.

Admin role (adminRoutes.js):

The allows the admin to manage all the content of dance moves. The admin can create the new dance moves by uploading all the details of it and then it is stored in the database. The admin can update the dance move by editing it in the edit form and can also delete it from the database.

User role:

The user role has this CRUD operations for creating their custom sequence and to schedule their dance practice sessions using calendar.

- Sequence Creation (SequenceRoutes.js):

The authenticated user can create and customize their own sequences. The users can create, read, update and delete their sequences, the userID and the dance moves are then stored in the database along with created sequence in the sequences collection.

- Scheduling Session (SessionRoutes.js):

Users can schedule their dance practice sessions using the calendar in their dashboard. The users can also view, update and delete their practice sessions. The sessions created are stored in the Sessions collection of database with the userID, selected sequence, date, description and duration of the session.

Server Setup (server.js):

The Server is the main part of the system which is the entry point to the backend. This helps to run the whole application, connects it to the database and links all the routes.

- The server connects to the database to run the application

```
const connectDB = require("./config/db");

connectDB()
    .then(() => console.log("MongoDB Connected Successfully!"))
    .catch((err) => {
        console.error("MongoDB Connection Error:", err);
        process.exit(1);
    });
```

- Each route is linked to its specific route handler where all the API logic is defined.

```
const authRoutes = require("./routes/authRoutes");
app.use("/api/auth", authRoutes);

const adminRoutes = require("./routes/adminRoutes");
app.use("/api/admin", adminRoutes);

const danceRoutes = require("./routes/danceRoutes");
app.use("/api/dances", danceRoutes);

const sequenceRoutes = require("./routes/sequenceRoutes");
app.use("/api/sequences", sequenceRoutes);

const sessionRoutes = require("./routes/sessionRoutes");
app.use("/api/sessions", sessionRoutes);
```

- Server static files like HTML, CSS and JS

```
app.use(express.static(path.join(__dirname, "public")));
```

- Starts the server and confirms the server is running

```
const PORT = process.env.PORT || 3000;
app.listen(PORT, () => {
    console.log(`Server running on http://localhost:${PORT}`);
});
```

6. Implementation

7. Testing

8. Discussion

9. Conclusion

10. References

1. Brusilovsky, P. & Millán, E. (2021) 'Adaptive learning systems: From data-driven personalizations to AI-powered recommendations', *ACM Computing Surveys*, 54(3), pp. 1-35.
2. Pane, J. F., Steiner, E. D., Baird, M. D., Hamilton, L. S., & Pane, J. D. (2017). *Informing Progress: Insights on Personalized Learning Implementation and Effects*. RAND Corporation.
3. Mayer, R. E. (2009). *Multimedia Learning* (2nd ed.). Cambridge University Press.
4. Li, Z. (Michael) (2021) 'Creativity and opportunity: how COVID-19 fosters digital dance education', *Digital Creativity*, 32(3), pp. 188–207. doi: 10.1080/14626268.2021.1967406.
5. Ballet with Isabella. (n.d.) *The benefits of online ballet training with Isabella*. Available at: https://balletwithisabella.com/posts/the-benefits-of-online-ballet-training-with-isabella/ (Accessed: 11 April 2025).
6. Xie, H., Chu, H.C., Hwang, G.J. and Wang, C.C., 2019. *Trends and development in technology-enhanced adaptive/personalized learning: A systematic review of journal publications from 2007 to 2017*. *Computers & Education*, 140, p.103599.
7. Chen, C.-M., & Li, Y.-L. (2010). *Personalized e-learning system using Item Response Theory*. Computers & Education, 55(4), 1625–1633.
8. Khalil, M. & Ebner, M. (2016) 'Learning Analytics: Principles and Constraints', *Proceedings of the International Conference on E-Learning & E-Technologies*, pp. 15–22.
9. Deterding, S., Dixon, D., Khaled, R. & Nacke, L. (2011) 'From game design elements to gamefulness: defining "gamification"', *Proceedings of the 15th International Academic MindTrek Conference*, pp. 9–15.
10. Jonassen, D.H. (1991). Objectivism versus constructivism: Do we need a new philosophical paradigm? *Educational Technology Research and Development*, 39(3), 5–14.
11. Preece, J. (2000). *Online Communities: Designing Usability, Supporting Sociability*. Wiley.
12. Wang, F. & Hannafin, M. (2005). "Design-based research and technology-enhanced learning environments," *Educational Technology Research and Development*, 53(4), pp. 5–23.